Red Hat Enterprise Linux 6

導入ガイド

Red Hat Enterprise Linux 6 の導入、設定、管理

エディッション 4

Last Updated: 2017-11-29

Red Hat Enterprise Linux 6 導入ガイド

Red Hat Enterprise Linux 6 の導入、設定、管理
エディッション 4

Jaromír Hradílek
Red Hat エンジニアリングコンテンツサービス
jhradilek@redhat.com

Douglas Silas
Red Hat エンジニアリングコンテンツサービス
silas@redhat.com

Martin Prpič
Red Hat エンジニアリングコンテンツサービス
mprpic@redhat.com

Stephen Wadeley
Red Hat エンジニアリングコンテンツサービス
swadeley@redhat.com

Eva Kopalová
Red Hat エンジニアリングコンテンツサービス
ekopalova@redhat.com

Ella Lackey
Red Hat エンジニアリングコンテンツサービス
dlackey@redhat.com

Tomáš Čapek
Red Hat エンジニアリングコンテンツサービス
tcapek@redhat.com

Petr Kovář
Red Hat エンジニアリングコンテンツサービス
pkovar@redhat.com

Miroslav Svoboda
Red Hat エンジニアリングコンテンツサービス
msvoboda@redhat.com

Petr Bokoč
Red Hat エンジニアリングコンテンツサービス
pbokoc@redhat.com

Florian Nadge
Red Hat エンジニアリングコンテンツサービス
fnadge@redhat.com

John Ha
Red Hat エンジニアリングコンテンツサービス

法律上の通知

Red Hat エンジニアリングコンテンツサービス
Copyright © 2010–2013 Red Hat, Inc.
Michael Hideo
Red Hat エンジニアリングコンテンツサービス
Douglas Silas
Red Hat エンジニアリングコンテンツサービス

This document is licensed by Red Hat under the Creative Commons Attribution-ShareAlike 3.0 Unported License. If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original. If the document is modified, all Red Hat trademarks must be removed.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

概要

本 導入ガイド は Red Hat Enterprise Linux 6 の導入、設定、管理の関連情報について説明しています。本書は、システムに関する基本的な理解をお持ちのシステム管理者を対象としています。

目次

前書き **13**
1. 本書の対象読者 13
2. 本書を読むにあたって 13
3. 謝辞 16

パート I. システムの基本設定 **18**

第1章 キーボード設定 **19**
1.1. キーボードレイアウトの変更 19
1.2. キーボードレイアウト表示器の追加 21
1.3. 一休みの設定 23

第2章 日付と時刻の設定 **25**
2.1. 日付/時刻のプロパティのツール 25
2.1.1. 日付と時刻のプロパティ 25
2.1.2. ネットワーク時刻プロトコルのプロパティ 26
2.1.3. タイムゾーンのプロパティ 27
2.2. コマンドラインからの設定 28
2.2.1. 日付と時刻の設定 29
2.2.2. ネットワーク時刻プロトコルの設定 29

第3章 ユーザーとグループの管理 **32**
3.1. ユーザーとグループの概要 32
3.1.1. ユーザープライベートグループ 32
3.1.2. シャドウパスワード 32
3.2. ユーザー管理ツールの使用 33
3.2.1. ユーザーとグループの表示 33
3.2.2. 新規ユーザーの追加 34
3.2.3. 新規グループの追加 36
3.2.4. ユーザープロパティの変更 36
3.2.5. グループプロパティの変更 37
3.3. コマンドラインツールの使用 38
3.3.1. 新規ユーザーの追加 39
3.3.2. 新規グループの追加 42
3.3.3. グループディレクトリの作成 42
3.4. その他のリソース 43
3.4.1. インストールされているドキュメント 43

第4章 権限の取得 **45**
4.1. SU コマンド 45
4.2. SUDO コマンド 46
4.3. その他のリソース 47
インストールされているドキュメント 47
オンラインのドキュメント 48

パート II. パッケージ管理 **49**

第5章 システム登録およびサブスクリプションの管理 **50**
5.1. RED HAT サブスクリプションマネージャーツールの使用 50
5.1.1. Red Hat サブスクリプションマネージャー GUI の起動 50
5.1.2. サブスクリプションマネージャーのコマンドラインツールの実行 51
5.2. システムの登録および登録解除 51
5.2.1. GUI からの登録 51

5.2.2. コマンドラインからの登録 56
5.2.3. 登録解除 59
5.3. サブスクリプションのアタッチと削除 60
5.3.1. GUI を使用したサブスクリプションのアタッチと削除 60
5.3.2. コマンドラインを使用したサブスクリプションのアタッチと削除 63
5.4. ベンダーサブスクリプションの有効化 65
5.4.1. GUI を使用したサブスクリプションの有効化 65
5.4.2. コマンドラインを使用したサブスクリプションの有効化 66
5.5. SUBSCRIPTION ASSET MANAGER のアクティベーションキーを使用したサブスクリプションのアタッチ方法 67
5.6. システムの詳細設定 67
5.6.1. UI での詳細設定 67
5.6.2. コマンドラインを使用したサービスレベルの設定 68
5.6.3. コマンドラインで希望するオペレーティングシステムのリリースバージョンを設定する 69
5.7. サブスクリプションの有効期限と通知の管理 70

第6章 YUM 75
6.1. パッケージの確認と更新 75
6.1.1. 更新の確認 75
6.1.2. パッケージの更新 76
6.1.3. 設定ファイルの変更の保存 78
6.2. パッケージとパッケージグループ 78
6.2.1. パッケージの検索 78
6.2.2. パッケージの一覧表示 78
6.2.3. パッケージ情報の表示 80
6.2.4. パッケージのインストール 82
6.2.5. パッケージの削除 84
6.2.6. トランザクション履歴の活用 85
6.3. YUM と YUM リポジトリの設定 91
6.3.1. [main] オプションの設定 91
6.3.2. [repository] オプションの設定 95
6.3.3. Yum 変数の使用 97
6.3.4. 現在の設定の表示 98
6.3.5. Yum リポジトリを有効/設定/無効にする方法 98
6.3.6. Yum リポジトリの作成 100
6.4. YUM のプラグイン 101
6.4.1. Yum プラグインを有効/設定/無効にする方法 101
6.4.2. 追加の Yum プラグインのインストール 102
6.4.3. プラグインの詳細 102
6.5. その他のリソース 109

第7章 PACKAGEKIT 110
7.1. ソフトウェア更新を使ったパッケージ更新 110
更新の確認頻度の設定 111
7.2. ソフトウェアの追加/削除の使用 112
7.2.1. ソフトウェアソースを最新の情報への更新 (yum リポジトリ) 112
7.2.2. フィルターを使ったパッケージ検索 113
7.2.3. パッケージ (及び依存関係) のインストールと削除 115
7.2.4. パッケージグループのインストールと削除 116
7.2.5. トランザクションログの表示 117
7.3. PACKAGEKIT アーキテクチャ 118
7.4. その他のリソース 119

パート III. ネットワーキング 121

第8章 NETWORKMANAGER **122**
8.1. NETWORKMANAGER デーモン 122
8.2. NETWORKMANAGER との対話 122
8.2.1. ネットワークへの接続 123
8.2.2. 新規接続の設定と既存接続の編集 124
8.2.3. ネットワークへの自動接続 125
8.2.4. ユーザー接続とシステム接続 126
8.3. 接続の確立 127
8.3.1. 有線 (イーサネット) 接続の確立 127
8.3.2. 無線接続の確立 130
8.3.3. モバイルブロードバンド接続の確立 135
8.3.4. VPN 接続の確立 138
8.3.5. DSL 接続の確立 142
8.3.6. Bond 接続の確立 143
8.3.7. VLAN 接続の確立 147
8.3.8. IP-over-InfiniBand (IPoIB) 接続の確立 149
8.3.9. 接続セッティングの設定 151
8.4. NETWORKMANAGER のアーキテクチャ 159

第9章 ネットワークインターフェース **160**
9.1. ネットワーク設定ファイル 160
9.2. インターフェース設定ファイル 161
9.2.1. イーサネットインターフェース 161
9.2.2. System z 上の Linux 特定の ifcfg オプション 165
9.2.3. System z 上の Linux の必須 ifcfg オプション 166
9.2.4. Ethtool 166
9.2.5. チャンネルボンディングインターフェース 172
9.2.6. ネットワークブリッジ 174
9.2.7. 802.1q VLAN タグのセットアップ 177
9.2.8. エイリアスファイルとクローンファイル 178
9.2.9. ダイヤルアップインターフェース 179
9.2.10. その他のインターフェース 181
9.3. インターフェース制御スクリプト 182
9.4. 静的ルートおよびデフォルトゲートウェイ 184
静的ルート 184
デフォルトゲートウェイ 184
IP コマンド引数形式 185
ネットワーク/ネットマスク指示文の形式 186
9.5. ネットワーク関数ファイル 186
9.6. その他のリソース 187
9.6.1. インストールされているドキュメント 187
9.6.2. 役立つ Web サイト 187

パート IV. インフラストラクチャのサービス **188**

第10章 サービスとデーモン **189**
10.1. デフォルトのランレベルの設定 189
10.2. サービスの設定 190
10.2.1. サービス設定ユーティリティの使用 190
10.2.2. ntsysv ユーティリティの使用 192
10.2.3. chkconfig ユーティリティの使用 194
10.3. サービスの実行 196
10.3.1. サービスステータスの確認 196
10.3.2. サービスの開始 197

10.3.3. サービスの停止 197
10.3.4. サービスの再開 197
10.4. その他のリソース 197
10.4.1. インストールされているドキュメント 197
10.4.2. 関連資料 197

第11章 認証の設定 199
11.1. システム認証の設定 199
11.1.1. 認証設定ツール UI の起動 199
11.1.2. 認証目的での ID ストアの選択 200
11.1.3. 代替認証機能の設定 207
11.1.4. コマンドライン上で 認証の設定 209
11.1.5. カスタムホームディレクトリの使用 213
11.2. SSSDでの認証情報の使用とキャッシング 213
11.2.1. SSSD について 213
11.2.2. sssd.conf ファイルの設定 215
11.2.3. SSSD の起動と停止 216
11.2.4. SSSD およびシステムサービス 217
11.2.5. サービスの設定: NSS 218
11.2.6. サービスの設定: PAM 221
11.2.7. サービスの設定: autofs 223
11.2.8. サービスの設定: sudo 226
11.2.9. サービスの設定: OpenSSH およびキャッシュされたキー 229
11.2.10. SSSD および ID プロバイダー (ドメイン) 233
11.2.11. ドメインの作成: LDAP 237
11.2.12. ドメインの作成: Identity Management (IdM) 242
11.2.13. ドメインの作成: Active Directory 244
11.2.14. ドメインオプション: ユーザー名形式の設定 253
11.2.15. ドメインオプション: オフライン認証の有効化 254
11.2.16. ドメインオプション: パスワード有効期限の設定 255
11.2.17. ドメインオプション: DNS Service Discovery の使用 256
11.2.18. ドメインオプション: 証明書のサブジェクト名での IP アドレスの使用 (LDAP のみ) 258
11.2.19. ドメインの作成: Proxy 259
11.2.20. ドメインの作成: Kerberos 認証 261
11.2.21. ドメインの作成: アクセス制御 265
11.2.22. ドメインの作成: プライマリーサーバーとバックアップサーバー 266
11.2.23. SSSD ユーティリティのインストール 267
11.2.24. ローカルシステムユーザーの作成 267
11.2.25. キックスタート中にユーザーを SSSD キャッシュにシードする方法 268
11.2.26. SSSD キャッシュの管理 269
11.2.27. SSSD で NSCD の使用 271
11.2.28. SSSD のトラブルシューティング 272

第12章 OPENSSH 279
12.1. SSH プロトコル 279
12.1.1. SSH を使用する理由 279
12.1.2. 主要な機能 280
12.1.3. プロトコルのバージョン 280
12.1.4. SSH 接続のイベントシーケンス 280
12.2. OPENSSH の設定 282
12.2.1. 設定ファイル 282
12.2.2. OpenSSH サーバーの起動 284
12.2.3. リモート接続に必要な SSH 285

12.2.4. 鍵ベース認証の使用 285
12.3. OPENSSH クライアント 291
12.3.1. ssh ユーティリティの使用 291
12.3.2. scp ユーティリティの使用 292
12.3.3. sftp ユーティリティの使用 293
12.4. セキュアシェルの高機能 294
12.4.1. X11 転送 294
12.4.2. ポート転送 294
12.5. その他のリソース 295
12.5.1. インストールされているドキュメント 295
12.5.2. 役立つ Web サイト 296

パート V. サーバー 297

第13章 DHCP サーバー 298
13.1. DHCP を使用する理由 298
13.2. DHCP サーバーの設定 298
13.2.1. 設定ファイル 298
13.2.2. リースデータベース 301
13.2.3. サーバーの起動と停止 302
13.2.4. DHCP リレーエージェント 303
13.3. DHCP クライアントの設定 303
13.4. マルチホーム DHCP サーバーの設定 304
13.4.1. ホストの設定 305
13.5. IPV6 の DHCP (DHCPV6) 307
13.6. その他のリソース 308
13.6.1. インストールされているドキュメント 308

第14章 DNS サーバー 309
14.1. DNS の概要 309
14.1.1. ネームサーバーのゾーン 309
14.1.2. ネームサーバーのタイプ 309
14.1.3. ネームサーバーとしての BIND 310
14.2. BIND 310
14.2.1. named サービスの設定 310
14.2.2. ゾーンファイルの編集 319
14.2.3. rndc ユーティリティの使用 326
14.2.4. dig ユーティリティの使用 329
14.2.5. BIND の高度な機能 331
14.2.6. 回避すべき一般的な間違い 332
14.2.7. その他のリソース 333

第15章 WEB サーバー 336
15.1. APACHE HTTP サーバー 336
15.1.1. 新しい機能 336
15.1.2. 注目すべき変更 336
15.1.3. 設定の更新 336
15.1.4. httpd サービスの実行 337
15.1.5. 設定ファイルの編集 338
15.1.6. モジュールでの作業 373
15.1.7. 仮想ホストのセットアップ 374
15.1.8. SSL サーバーのセットアップ 374
15.1.9. その他のリソース 381

第16章 メールサーバー 383
16.1. 電子メールプロトコル 383
16.1.1. メール転送プロトコル 383
16.1.2. メールアクセスプロトコル 384
16.2. 電子メールプログラムの分類 386
16.2.1. メール転送エージェント (Mail Transport Agent) 386
16.2.2. メール配信エージェント (Mail Delivery Agent) 387
16.2.3. メールユーザーエージェント (Mail User Agent) 387
16.3. メール転送エージェント (MTA) 387
16.3.1. Postfix 387
16.3.2. Sendmail 390
16.3.3. Fetchmail 395
16.3.4. MTA の設定 399
16.4. メール配信エージェント(MAIL DELIVERY AGENTS)(MDA) 399
16.4.1. Procmail の設定 400
16.4.2. Procmail のレシピ 401
16.5. メールユーザーエージェント (MAIL USER AGENTS) 406
16.5.1. 通信のセキュリティ保護 406
16.6. その他のリソース 408
16.6.1. インストールされているドキュメント 408
16.6.2. 役立つ Web サイト 408
16.6.3. 関連資料 409

第17章 ディレクトリサーバー 410
17.1. OPENLDAP 410
17.1.1. LDAP の概要 410
17.1.2. OpenLDAP スイートのインストール 412
17.1.3. OpenLDAP サーバーの設定 415
17.1.4. OpenLDAP サーバーの実行 420
17.1.5. OpenLDAP を使用した認証用のシステムの設定 421
17.1.6. その他のリソース 422

第18章 ファイルとプリントサーバー 425
18.1. SAMBA 425
18.1.1. Samba の概要 425
18.1.2. Samba のデーモンと関連サービス 426
18.1.3. Samba 共有への接続 427
18.1.4. Samba サーバーの設定 429
18.1.5. Samba の起動と停止 430
18.1.6. Samba サーバーのタイプと smb.conf ファイル 431
18.1.7. Samba のセキュリティモード 438
18.1.8. Samba アカウント情報データベース 440
18.1.9. Samba のネットワークブラウジング 441
18.1.10. CUPS 印刷サポートを実装した Samba 442
18.1.11. Samba ディストリビューションプログラム 443
18.1.12. その他のリソース 448
18.2. FTP 449
18.2.1. File Transfer Protocol (ファイル転送プロトコル) 449
18.2.2. vsftpd サーバー 450
18.2.3. vsftpd によるファイルのインストール 451
18.2.4. vsftpd の起動と停止 451
18.2.5. vsftpd 設定オプション 452
18.2.6. その他のリソース 462

18.3. プリンターの設定 463
18.3.1. プリンター設定ツールの起動 463
18.3.2. プリンター設定の開始 464
18.3.3. ローカルプリンターの追加 464
18.3.4. AppSocket/HP JetDirect プリンターの追加 465
18.3.5. IPP プリンターの追加 466
18.3.6. LPD/LPR Host or Printer の追加 467
18.3.7. Samba (SMB) プリンターの追加 468
18.3.8. プリンターモデルの選択と完了 470
18.3.9. テストページの印刷 473
18.3.10. 既存プリンターの修正 473
18.3.11. その他のリソース 480

パート VI. 監視と自動化 482

第19章 システムモニタリングツール 483
19.1. システムプロセスの表示 483
19.1.1. ps コマンドの使用 483
19.1.2. top コマンドの使用 484
19.1.3. システムモニターツールの使用 485
19.2. メモリ使用量の表示 487
19.2.1. free コマンドの使用 487
19.2.2. システムモニターツールの使用 487
19.3. CPU 使用率の表示 488
19.3.1. システムモニターツールの使用 488
19.4. ブロックデバイスとファイルシステムの表示 489
19.4.1. lsblk コマンドの使用 489
19.4.2. blkid コマンドの使用 490
19.4.3. findmnt コマンドの使用 491
19.4.4. df コマンドの使用 493
19.4.5. du コマンドの使用 493
19.4.6. システムモニターツールの使用 494
19.5. ハードウェア情報の表示 495
19.5.1. lspci コマンドの使用 495
19.5.2. lsusb コマンドの使用 496
19.5.3. lspcmcia コマンドの使用 497
19.5.4. lscpu コマンドの使用 497
19.6. NET-SNMP を使用したパフォーマンスのモニタリング 498
19.6.1. Net-SNMP のインストール 498
19.6.2. Net-SNMP Daemon の実行 499
19.6.3. Net-SNMP の設定 500
19.6.4. SNMP によるパフォーマンスデータの取得 502
19.6.5. Net-SNMP の拡張 506
19.7. その他のリソース 511
19.7.1. インストールされているドキュメント 511

第20章 ログファイルの表示と管理 513
20.1. RSYSLOG の設定 513
20.1.1. グローバル指示文 513
20.1.2. モジュール 513
20.1.3. ルール 514
20.1.4. rsyslog コマンドラインの設定 525
20.2. ログファイルの場所の特定 526
20.2.1. logrotate の設定 526

20.3. ログファイルの表示 527
20.4. ログファイルの追加 530
20.5. ログファイルのモニタリング 531
20.6. その他のリソース 532
20.6.1. インストールされているドキュメント 532
20.6.2. 役立つ Web サイト 532

第21章 システムタスクの自動化 533
21.1. CRON と ANACRON 533
21.1.1. Cron と Anacron のインストール 533
21.1.2. Crond サービスの実行 533
21.1.3. Anacron ジョブの設定 534
21.1.4. Cron ジョブの設定 536
21.1.5. Cron へのアクセスの制御 538
21.1.6. Cron ジョブのブラックリスト/ホワイトリスト 538
21.2. AT コマンドと BATCH コマンド 539
21.2.1. At と Batch のインストール 539
21.2.2. At サービスの実行 539
21.2.3. At ジョブの設定 540
21.2.4. Batch ジョブの設定 541
21.2.5. 保留中のジョブの表示 542
21.2.6. 追加のコマンドラインオプション 542
21.2.7. at と batch へのアクセスの制御 542
21.3. その他のリソース 543

第22章 自動バグ報告ツール (ABRT) 544
22.1. ABRT のインストールとサービスの起動 545
22.2. グラフィカルユーザーインターフェイスの使用 547
22.3. コマンドラインインターフェイスの使用 554
22.3.1. 問題の表示 554
22.3.2. 問題の報告 556
22.3.3. 問題の削除 557
22.4. ABRT の設定 558
22.4.1. ABRT イベント 558
22.4.2. ABRT の標準インストールでサポートされているイベント 560
22.4.3. ABRT GUI でのイベント設定 561
22.4.4. ABRT 固有の設定 563
22.4.5. カーネルパニックを検出する ABRT の設定 566
22.4.6. 自動ダウンロードおよび Debuginfo パッケージのインストール 566
22.4.7. 自動報告機能の設定 567
22.4.8. プロキシサーバーを使用したアップロードと報告 567
22.5. 集中型のクラッシュ収集の設定 568
22.5.1. 専用システム上で必要な設定手順 568
22.5.2. クライアントシステム上で必要な設定ステップ 569
22.5.3. パッケージ情報の保存 569
22.5.4. ABRT のクラッシュ検出テスト 571

第23章 OPROFILE 572
23.1. ツールの概要 572
23.2. OPROFILE の設定 573
23.2.1. カーネルの指定 573
23.2.2. 監視するイベントのセッティング 574
23.2.3. カーネルとユーザースペースのプロファイルの分離 577
23.3. OPROFILE の開始と停止 578

23.4. データの保存 579
23.5. データの分析 579
23.5.1. opreport の使用 580
23.5.2. 単独実行可能ファイルで opreport の使用 581
23.5.3. モジュールについてより詳細な出力を取得 583
23.5.4. opannotate の使用 584
23.6. /DEV/OPROFILE/ を理解する 584
23.7. 使用法の例 585
23.8. JAVA 対応の OPROFILE サポート 585
23.8.1. Java コードのプロファイル作成 585
23.9. グラフィカルインターフェース 585
23.10. OPROFILE と SYSTEMTAP 589
23.11. その他のリソース 589
23.11.1. インストール済みドキュメント 589
23.11.2. 役立つ Web サイト 589

パート VII. カーネル、モジュール、及びドライバーの設定 590

第24章 手動のカーネルアップグレード 591
24.1. カーネルパッケージの概要 591
24.2. アップグレードへの準備 592
24.3. アップグレードされたカーネルのダウンロード 593
24.4. アップグレードの実行 594
24.5. 初期 RAM ディスクイメージの検証 594
IBM eServer System i 上の初期 RAM ディスクイメージとカーネルの確証 596
24.6. ブートローダーの確証 596
24.6.1. GRUB ブートローダーの設定 597
24.6.2. OS/400 ブートローダーの設定 599
24.6.3. YABOOT ブートローダーの設定 599

第25章 カーネルモジュールでの作業 601
25.1. 現在ロード済みモジュールの一覧表示 601
25.2. モジュール情報の表示 602
25.3. モジュールのロード 605
25.4. モジュールのアンロード 605
25.5. モジュールパラメーターの設定 606
25.6. 永続的なモジュールのロード 608
25.7. カーネルモジュール特有の機能 609
25.7.1. 複数のイーサネットカードの使用 609
25.7.2. チャンネルボンディングの使用 609
25.8. その他のリソース 615
man ページ 615
インストールできる外部のドキュメント 616

第26章 KDUMP クラッシュリカバリーサービス 617
26.1. KDUMP サービスのインストール 617
26.2. KDUMP サービスの設定 617
26.2.1. 初回ブート時の kdump の設定 617
26.2.2. カーネルダンプ設定ユーティリティの使用 618
26.2.3. コマンドライン上で kdump の設定 622
26.2.4. 設定のテスト 626
26.3. コアダンプの分析 626
26.3.1. crash ユーティリティの実行 626
26.3.2. メッセージバッファーの表示 628

26.3.3. バックトレースの表示 628
26.3.4. プロセスステータスの表示 629
26.3.5. 仮想メモリ情報の表示 629
26.3.6. オープンファイルの表示 630
26.3.7. ユーティリティの終了 631
26.4. その他のリソース 631
26.4.1. インストールされているドキュメント 631
26.4.2. 役立つ Web サイト 631

付録A 一貫したネットワークデバイスの命名 632
A.1. 影響を受けるシステム 632
A.2. システムの要件 633
A.3. この機能の有効化と無効化 633
A.4. 管理者のための注記 633

付録B RPM 635
B.1. RPM の設計目標 636
B.2. RPM の使用法 636
B.2.1. RPM パッケージの検索 637
B.2.2. インストールとアップグレード 637
B.2.3. 設定ファイルの変更 640
B.2.4. アンインストール 641
B.2.5. 最新の状態にする 642
B.2.6. クエリ 642
B.2.7. 検証 643
B.3. パッケージの署名確認 644
B.3.1. キーのインポート 645
B.3.2. パッケージの署名の検証 645
B.4. 実用的で一般的な RPM の使用例 645
B.5. その他のリソース 648
B.5.1. インストールされているドキュメント 648
B.5.2. 役立つ Web サイト 648
B.5.3. 関連資料 648

付録C X WINDOW SYSTEM 649
C.1. X サーバー 649
C.2. デスクトップ環境とウィンドウマネージャー 649
C.2.1. デスクトップ環境 650
C.2.2. ウィンドウマネージャー 650
C.3. X サーバーの設定ファイル 651
C.3.1. 設定の構造 651
C.3.2. xorg.conf.d ディレクトリ 652
C.3.3. xorg.conf ファイル 652
C.4. フォント 660
C.4.1. Fontconfig へのフォントの追加 660
C.5. ランレベルと X 661
C.5.1. ランレベル 3 661
C.5.2. ランレベル 5 662
C.6. その他のリソース 663
C.6.1. インストールされているドキュメント 663
C.6.2. 役立つ Web サイト 663

付録D SYSCONFIG ディレクトリ 664
D.1. /ETC/SYSCONFIG/ ディレクトリのファイル 664

D.1.1. /etc/sysconfig/arpwatch 664
D.1.2. /etc/sysconfig/authconfig 664
D.1.3. /etc/sysconfig/autofs 667
D.1.4. /etc/sysconfig/clock 669
D.1.5. /etc/sysconfig/dhcpd 669
D.1.6. /etc/sysconfig/firstboot 669
D.1.7. /etc/sysconfig/i18n 670
D.1.8. /etc/sysconfig/init 670
D.1.9. /etc/sysconfig/ip6tables-config 672
D.1.10. /etc/sysconfig/keyboard 673
D.1.11. /etc/sysconfig/ldap 674
D.1.12. /etc/sysconfig/named 675
D.1.13. /etc/sysconfig/network 675
D.1.14. /etc/sysconfig/ntpd 676
D.1.15. /etc/sysconfig/quagga 676
D.1.16. /etc/sysconfig/radvd 678
D.1.17. /etc/sysconfig/samba 678
D.1.18. /etc/sysconfig/saslauthd 678
D.1.19. /etc/sysconfig/selinux 679
D.1.20. /etc/sysconfig/sendmail 679
D.1.21. /etc/sysconfig/spamassassin 680
D.1.22. /etc/sysconfig/squid 680
D.1.23. /etc/sysconfig/system-config-users 680
D.1.24. /etc/sysconfig/vncservers 681
D.1.25. /etc/sysconfig/xinetd 681
D.2. /ETC/SYSCONFIG/ ディレクトリ内のディレクトリ 682
D.3. その他のリソース 682
D.3.1. インストールされているドキュメント 682

付録E PROC ファイルシステム 684
E.1. 仮想ファイルシステム 684
E.1.1. 仮想ファイルの表示 684
E.1.2. 仮想ファイルの変更 685
E.2. PROC ファイルシステム内の最上位のファイル 685
E.2.1. /proc/buddyinfo 686
E.2.2. /proc/cmdline 686
E.2.3. /proc/cpuinfo 686
E.2.4. /proc/crypto 687
E.2.5. /proc/devices 688
E.2.6. /proc/dma 689
E.2.7. /proc/execdomains 689
E.2.8. /proc/fb 689
E.2.9. /proc/filesystems 689
E.2.10. /proc/interrupts 690
E.2.11. /proc/iomem 691
E.2.12. /proc/ioports 691
E.2.13. /proc/kcore 692
E.2.14. /proc/kmsg 692
E.2.15. /proc/loadavg 692
E.2.16. /proc/locks 692
E.2.17. /proc/mdstat 693
E.2.18. /proc/meminfo 693
E.2.19. /proc/misc 695

E.2.20. /proc/modules 695
E.2.21. /proc/mounts 696
E.2.22. /proc/mtrr 697
E.2.23. /proc/partitions 697
E.2.24. /proc/slabinfo 697
E.2.25. /proc/stat 699
E.2.26. /proc/swaps 700
E.2.27. /proc/sysrq-trigger 700
E.2.28. /proc/uptime 700
E.2.29. /proc/version 700
E.3. /PROC/ 内のディレクトリ 700
E.3.1. プロセスディレクトリ 701
E.3.2. /proc/bus/ 703
E.3.3. /proc/bus/pci 704
E.3.4. /proc/driver/ 704
E.3.5. /proc/fs 705
E.3.6. /proc/irq/ 705
E.3.7. /proc/net/ 705
E.3.8. /proc/scsi/ 706
E.3.9. /proc/sys/ 708
E.3.10. /proc/sysvipc/ 718
E.3.11. /proc/tty/ 719
E.3.12. /proc/PID/ 719
E.4. SYSCTL コマンドの使用 720
E.5. その他のリソース 721
E.5.1. インストールされているドキュメント 721
E.5.2. 役立つ Web サイト 722

付録F 改訂履歴 723

索引 724

前書き

本 『導入ガイド』 は、Red Hat Enterprise Linux 6 システムをご希望のニーズに合わせてカスタマイズする方法について説明しています。システムを設定/カスタマイズするにあたり総合的なタスク指向のガイドをお探しの場合は、本マニュアルを是非ご活用下さい。

本マニュアルでは、以下のような多くの中級レベルのトピックについて説明しています。

- グラフィカルパッケージマネージャーの **PackageKit** とコマンドラインの **Yum** パッケージマネージャーを使用したパッケージのインストールと管理
- ネットワークの設定 — **NetworkManager** を使用したイーサネット接続の確立から、サーバー帯域幅を増加させるためのチャンネルボンディングインターフェースの設定
- **DHCP**、**BIND**、**Apache HTTP サーバー**、**Postfix**、**Sendmail** 及び他のエンタープライズクラスのサーバーとソフトウェアの設定
- **Automatic Bug Reporting Tool (自動バグ報告ツール)** を使用したユーザー領域のクラッシュデータや **kdump** を使ったカーネル領域のクラッシュデータなど、システムに関する情報の収集
- 簡単なカーネルモジュールの使用とカーネルのアップグレード

1. 本書の対象読者

本 『導入ガイド』 では、読者が Red Hat Enterprise Linux オペレーティングシステムについて基本的な理解を持っていると仮定しています。本システムのインストールについては、『Red Hat Enterprise Linux 6 インストールガイド』 を参照して下さい。

2. 本書を読むにあたって

本マニュアルは、大きく次のように分かれています。

パートI「システムの基本設定」

ここでは、キーボード設定、日付と時刻の設定、ユーザーとグループの管理、権限の取得など基本的なシステム管理タスクを取り上げます。

1章 *キーボード設定* では、基本的なキーボード設定について説明しています。キーボードのレイアウト変更、パネルへの **Keyboard Indicator (キーボード表示器)** アプレットの追加、強制的に定期休憩を設定する方法については、本章を参照して下さい。

2章 *日付と時刻の設定* では、システムの日付と時刻の設定について説明しています。日時設定を変更する場合や、リモートの NTP (Network Time Protocol: ネットワーク時刻プロトコル) サーバーを使って時計を同期するようにシステムを設定したい場合は、本章をお読み下さい。

3章 *ユーザーとグループの管理* では、グラフィカルユーザーインターフェースとコマンドラインでのユーザーとグループの管理について取り上げています。システム上でのユーザーとグループの管理、パスワードエージングの有効方法については、本章をお読み下さい。

4章 *権限の取得* では、管理者権限を取得する方法が説明されています。**su** および **sudo** コマンドの使用法に関しては、この章をお読みください。

パートII「パッケージ管理」

ここでは、製品サブスクリプションとエンタイトルメントに焦点を置きながら、グラフィカルパッケージ管理ツールである **Yum** と **PackageKit** スイートを使用した Red Hat Enterprise Linux でのソフトウェアパッケージの管理方法について説明します。

5章*システム登録およびサブスクリプションの管理* は、Red Hat Enterprise Linux におけるサブスクリプション管理と利用可能な Red Hat サブスクリプションマネージャツールの概要を説明しています。システムの登録/登録解除、マシンのアクティベート、製品サブスクリプションとエンタイトルメントの処理方法について知りたい場合は、本章をお読み下さい。

6章*Yum* では、**Yum** パッケージマネージャーについて説明しています。コマンドラインでパッケージを検索/インストール/更新/アンインストールする方法については、本章を参照して下さい。

7章*PackageKit* は、グラフィカルパッケージ管理ツールの **PackageKit** スイートについて記載しています。グラフィカルユーザーインターフェースを使ってパッケージを検索/インストール/更新/アンインストールする方法を知りたい場合は、本章をお読み下さい。

パートIII「ネットワーキング」

ここでは、Red Hat Enterprise Linux でのネットワーク設定方法について紹介します。

8章*NetworkManager* では、**NetworkManager** と呼ばれる、ネットワークデバイスと接続を利用可能時に常にアクティブに保つ動的ネットワーク制御及び設定システムに焦点を置いています。本章では、**`NetworkManager`** デーモンの実行方法と通知スペースの該当するアプレットを使用したやりとりの方法について説明しています。

9章*ネットワークインターフェース* は、**`/etc/sysconfig/network-scripts/`** ディレクトリにある様々なインターフェース設定ファイル、インターフェース制御スクリプト、ネットワーク機能ファイルについて取り上げます。ネットワークインターフェースを設定するためのこうしたファイルの使用方法については、本章を参照して下さい。

パートIV「インフラストラクチャのサービス」

ここでは、サービスとデーモンの設定、認証設定、リモートログインの有効方法について解説します。

10章*サービスとデーモン* では、ランレベルの概念、そのデフォルトを設定する方法について説明しています。また、各ランレベルで実行するサービスの設定、サービスを開始/停止/再起動する方法についても取り上げています。ご使用のシステムでサービスを管理する方法を知りたい場合は、本章をお読み下さい。

11章*認証の設定* は、LDAP (Lightweight Directory Access Protocol: ライトウェイトディレクトリアクセスプロトコル)、NIS (Network Information Service: ネットワーク情報サービス) 及び Winbind のユーザーアカウントデータベースから取得したユーザー情報の設定方法を説明して、SSSD (System Security Services Daemon: システムセキュリティサービスデーモン) について紹介します。ご使用のシステムで認証設定が必要な場合は、本章をお読み下さい。

12章*OpenSSH* では、SSH プロトコルによるリモートログインを有効にする方法について説明しています。**`sshd`** サービスの設定、さらには **`ssh`**、**`scp`**、**`sftp`** クライアントユーティリティに関する基本的な使用方法も取り上げています。マシンにリモートアクセスが必要な場合は、本章をお読み下さい。

パートV「サーバー」

ここでは、Web サーバーの設定方法やネットワーク上でのファイルとディレクトリの共有方法など、サーバーに関連する様々なトピックについて説明します。

13章*DHCP サーバー* では、DHCP (Dynamic Host Configuration Protocol: 動的ホスト構成プロトコル) サーバーとクライアントのインストールについて説明しています。ご使用のシステムに DHCP を設定する必要がある場合は、本章をお読み下さい。

14章*DNS サーバー* では、DNS (Domain Name System: ドメインネームシステム) について紹介し、**BIND** DNS サーバーをインストール/設定/実行/管理する方法について説明しています。ご使用のシステムに DNS サーバーを設定する必要がある場合は、本章をお読み下さい。

15章*Web サーバー* では、**Apache HTTP サーバー 2.2** と呼ばれる Apache Software Foundation が開発した、堅牢でフル装備のオープンソース Web サーバーについて重点を置き解説しています。ご使用のシステムに Web サーバーを設定したい場合は、本章をお読み下さい。

16章*メールサーバー* では、現在使用している最新の電子メールプロトコルや **Postfix**、**Sendmail**、**Fetchmail**、**Procmail** など電子メールの送受信を行うプログラムについて説明しています。ご使用のシステムにメールサーバーを設定する場合は、本章をお読み下さい。

17章*ディレクトリサーバー* では、LDAPv2 及び LDAPv3 プロトコルのオープンソース実装である **OpenLDAP 2.4** をインストール/設定する方法について取り上げます。本章をお読みいただくと、ご使用のシステムにディレクトリサーバーを設定することができます。

18章*ファイルとプリントサーバー* では、SMB (Server Message Block: サーバーメッセージブロック) プロトコルのオープンソース実装である **Samba** と Red Hat Enterprise Linux に同梱されているプライマリ FTP サーバーである **vsftpd** をインストール/設定する方法を紹介しています。加えて、プリンターを設定する **Printer Configuration (プリンター設定)** ツールの使用方法についても説明しています。ご使用のシステムにファイル/プリントサーバーを設定したい場合は、本章をお読み下さい。

パートVI「監視と自動化」

ここでは、システム管理者がシステムパフォーマンスのモニター、システムタスクの自動化、バグの報告を行うための各種ツールについて解説します。

19章*システムモニタリングツール* では、システムに関する重要な情報を取得するために使用するアプリケーションとコマンドを説明しています。不可欠なシステム情報の取得方法については、本章をお読み下さい。

20章*ログファイルの表示と管理* では、**rsyslog** デーモンの設定の他、ログファイルを検索/表示/モニターする方法について説明しています。ログファイルの使用方法については、本章を参照して下さい。

21章*システムタスクの自動化* は、**cron**、**at** 及び **batch** ユーティリティの概要を記載しています。自動化タスクを実行するためのこうしたユーティリティの使用方法については、本章をお読み下さい。

22章*自動バグ報告ツール (ABRT)* では、クラッシュデータを収集し、関連する問題トラッカーにレポートを送信する **ABRT** と呼ばれるシステムサービスとツールセットについて重点的に説明しています。ご使用のシステムで **ABRT** を使用する方法については、本章をお読み下さい。

23章*OProfile* では、オーバーヘッドが低いシステム全体のパフォーマンスをモニターするツールである **OProfile** を取り上げます。ご使用のシステムでの **OProfile** の使用方法については、本章をお読み下さい。

パートVII「カーネル、モジュール、及びドライバーの設定」

ここでは、管理者がカーネルをカスタマイズする場合に役立つ様々なツールを取り上げます。

24章*手動のカーネルアップグレード* は、**yum** の代わりに **rpm** コマンドを使用してカーネルパッケージを手動で更新する方法に関する重要な情報を記載しています。**Yum** パッケージマネージャーでカーネルパッケージを更新できない場合は、本章をお読み下さい。

25章*カーネルモジュールでの作業* では、カーネルモジュール及びその依存関係を表示/クエリ/ロード/アンロードする方法の他、モジュールパラメーターの設定方法について説明しています。さらに、複数のイーサネットカードの使用やチャンネルボンディングの使用など、カーネルモジュール特有の機能についても取り上げています。カーネルモジュールを扱う場合は、本章をお読み下さい。

26章*kdump クラッシュリカバリーサービス* では、Red Hat Enterprise Linux で **kdump** サービスを設定/テスト/使用する方法について記載しています。また、それにより生じるコアダンプを **crash** デバッグユーティリティを使用して分析する方法の概要も示しています。ご使用のシステムで **kdump** を有効にしたい場合は、本章をお読み下さい。

付録A *一貫したネットワークデバイスの命名*

本付録は、ネットワークインターフェース用の一貫性のあるネットワークデバイスの命名、より簡単にインターフェースを検索/識別できるようにシステム上のネットワークインターフェースの名前を変更する機能について取り上げています。本機能を有効/無効にする方法など詳細を知りたい場合は、本付録をお読み下さい。

付録B *RPM*

本付録は、Red Hat Enterprise Linux で採用されているオープンパッケージングシステムである RPM (RPM パッケージマネージャー) と **rpm** ユーティリティの使用について主に説明しています。 **yum** の代わりに **rpm** を使用する必要がある場合は、本付録をお読み下さい。

付録C *X Window System*

本付録では、Red Hat Enterprise Linux で採用されているグラフィカル環境の X Window System の設定について取り上げています。ご使用の X Window System の設定を調節する必要がある場合は、本付録をお読み下さい。

付録D *sysconfig ディレクトリ*

本付録では、**/etc/sysconfig/** ディレクトリにある一部のファイル及びディレクトリについて概説しています。これらのファイルとディレクトリの機能やコンテンツの詳細が知りたい場合は、本付録をお読み下さい。

付録E *proc ファイルシステム*

本付録では、仮想ファイルシステムの概念、**proc** ファイルシステム (**/proc/** ディレクトリ) 内にある最上位のファイルとディレクトリについて説明しています。このファイルシステムの詳細については、本付録を参照して下さい。

3. 謝辞

本書の一部は、最初に http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5/html/Deployment_Guide/index.html にある 『Deployment Guide (導入ガイド)』, copyright © 2007 Red Hat, Inc. に記載されました。

「Net-SNMP を使用したパフォーマンスのモニタリング」 は、Michael Solberg 氏が執筆した記事を基にしています。

本書を執筆するにあたり、以下の方々から多大なご協力をいただきました: Adam Tkáč、Andrew

Fitzsimon、Andrius Benokraitis、Brian Cleary Edward Bailey、Garrett LeSage、Jeffrey Fearn、Joe Orton、Joshua Wulf、Karsten Wade、Lucy Ringland、Marcela Mašláňová、Mark Johnson、Michael Behm、Miroslav Lichvár、Radek Vokál、Rahul Kavalapara、Rahul Sundaram、Sandra Moore、Zbyšek Mráz、Jan Včelák、Peter Hutterer、James Antill 他、多数 (敬称略)。

パート I. システムの基本設定

ここでは、キーボード設定、日付と時刻の設定、ユーザーとグループの管理、権限の取得など基本的なシステム管理タスクを取り上げます。

第1章 キーボード設定

本章では、キーボードのレイアウトを変更する方法と **Keyboard Indicator (キーボード表示器)** アプレットをパネルに追加する方法について説明します。また、強制的に休憩を設定するオプション、それによるメリットとデメリットについても取り上げます。

1.1. キーボードレイアウトの変更

インストールプログラムにより、ご使用のシステム用にキーボードのレイアウトが設定されています。しかし、デフォルト設定が必ずしも現在のニーズに合うとは限りません。インストール後に違うキーボードのレイアウトを設定したい場合は、**Keyboard Preferences (キーボードの設定)** ツールを使用します。

キーボードのレイアウト設定 を開くには、パネルより **System (システム)** → **Preferences (設定)** → **Keyboard (キーボード)** の順に選択し、**Layouts (レイアウト)** タブをクリックします。

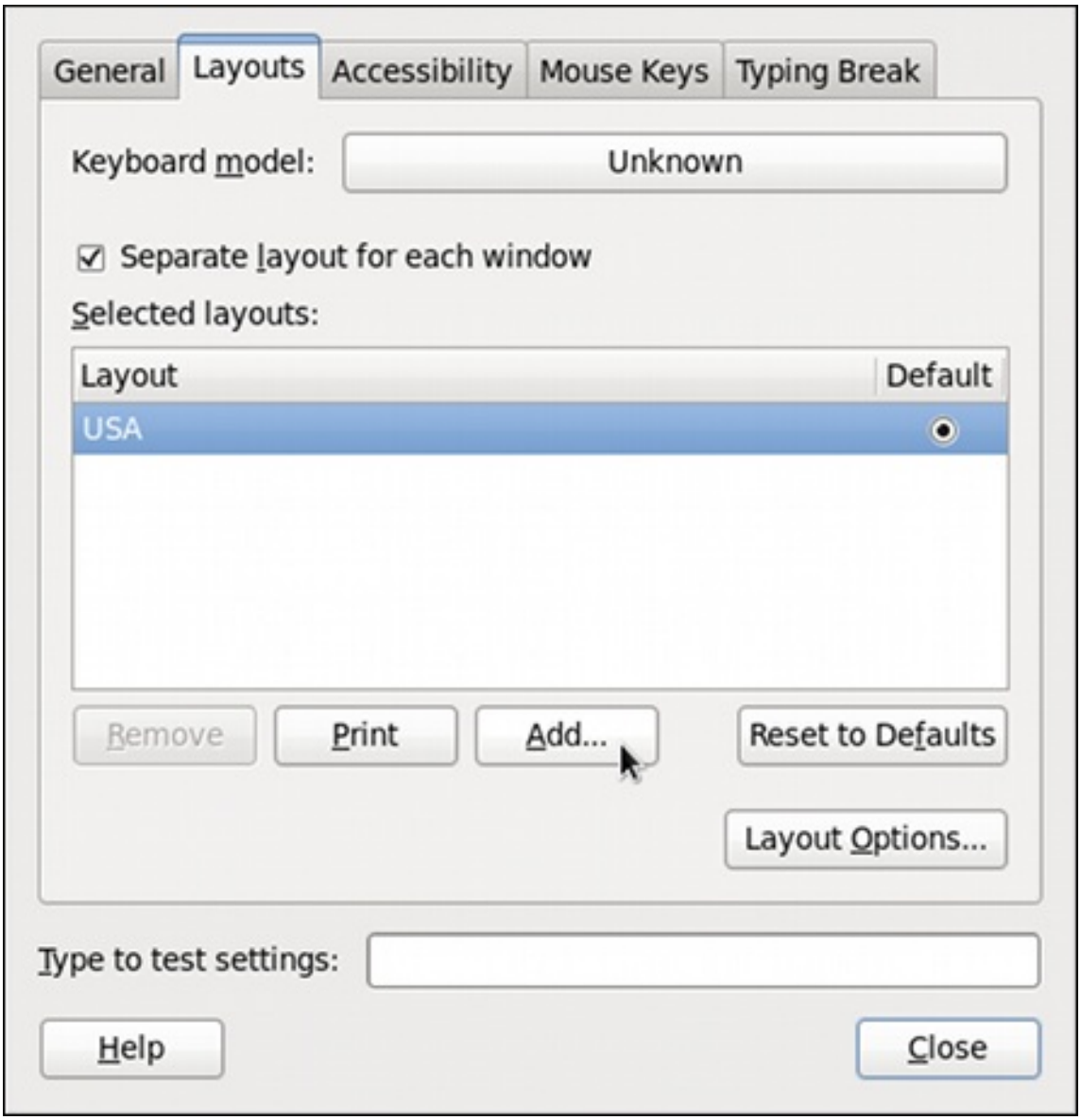


図1.1 キーボードのレイアウト設定

使用できるレイアウトの一覧が表示されます。新規レイアウトを追加するには、一覧の下の **追加** ボタンをクリックします。追加したいレイアウトを選択する画面が表示されます。

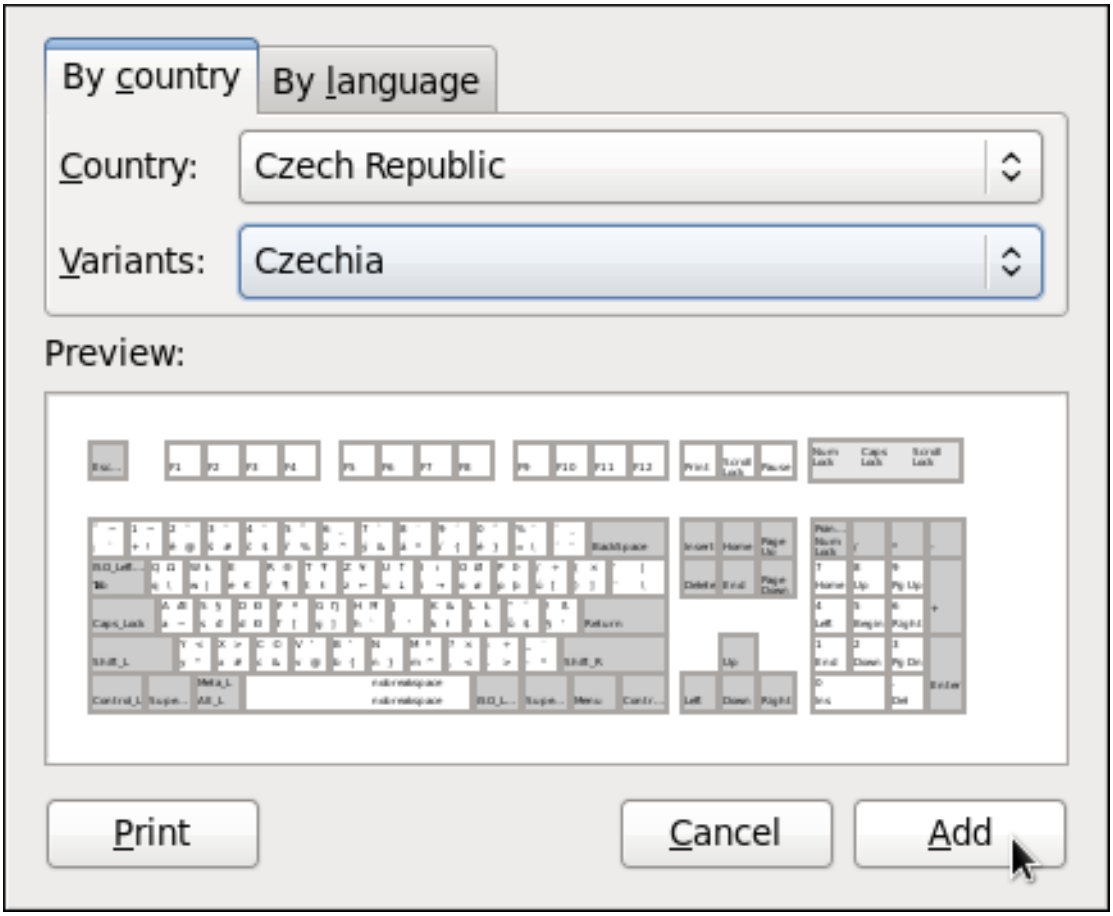


図1.2 レイアウトの選択

現時点では、キーボードのレイアウトを選択する方法は 2 つあります。関連する国から見つけるか (**By country (国別)** タブ)、言語から選択するか (**By language (言語別)** タブ) のどちらかです。いずれの場合も、最初に **国** または **言語** のプルダウンメニューから希望する国か言語を選択します。次に、**Variants (系列)** メニューから系列を指定します。選択したら、レイアウトのプレビューは直ちに変更されます。選択を決定するには、**追加** をクリックします。

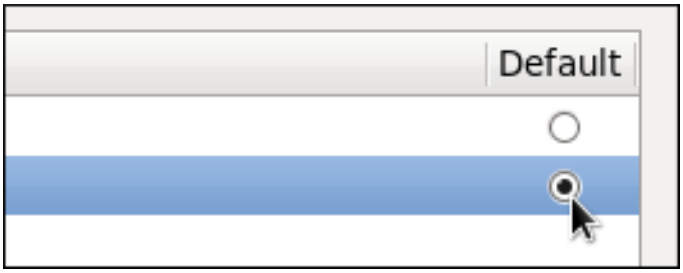


図1.3 デフォルトレイアウトの選択

選択したレイアウトが一覧に表示されるはずです。それをデフォルト設定にするには、名前の横にあるラジオボタンを選択して下さい。変更は直ちに反映されます。ウィンドウの最下部にはテキストを入力できるフィールドがあるため、そこで安心して設定を試すことができます。選択したレイアウトでよければ、**閉じる** をクリックしてウィンドウを閉じます。

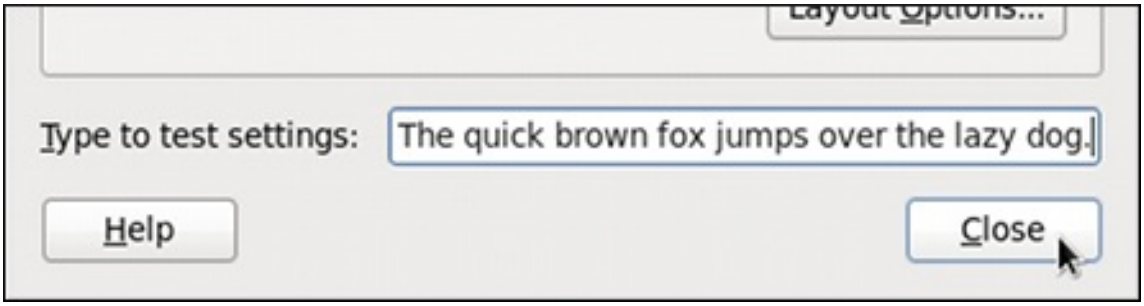
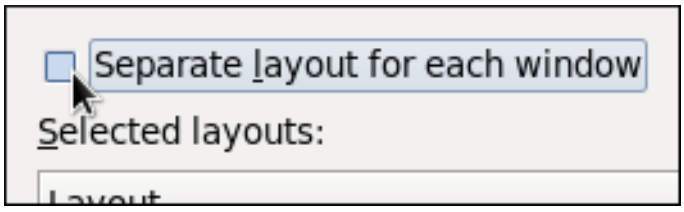


図1.4 レイアウトのテスト

注記

デフォルトでは、キーボードのレイアウトを変更するとアクティブなウィンドウのみで変更が反映されるようになっています。つまり、レイアウトを変更した後に別のウィンドウに切り替えると、この別のウィンドウでは前のレイアウトが適用されるため、混乱する場合があります。この機能をオフにするには、**ウィンドウ毎にグループ化する** チェックボックスからチェックマークを外して下さい。



ただこれには、図1.3「デフォルトレイアウトの選択」 で示されたようにラジオボタンを選択することで、デフォルトのレイアウトを選択できなくなるという欠点があります。選択したレイアウトをデフォルトにするには、一覧の冒頭へドラッグするだけで可能です。

1.2. キーボードレイアウト表示器の追加

現在使用しているキーボードのレイアウトを確認したい場合や、シングルクリックで異なるレイアウトに切り替わるようにするには、**キーボード表示器** アプレットをパネルに追加します。そのためには、メインパネルの空いているスペースで右クリックし、プルダウンメニューから **Add to Panel... (パネルへ追加...)** オプションを選択します。

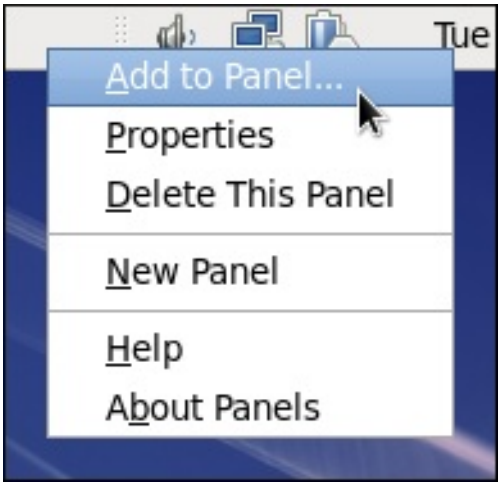


図1.5 新規アプレットの追加

利用可能なアプレットの一覧が表示されます。一覧をスクロールし (またはウィンドウの最上部にある検索フィールドに 「キーボード」 と入力し始める)、**キーボード表示器** を選択して、**追加** ボタンをクリックします。

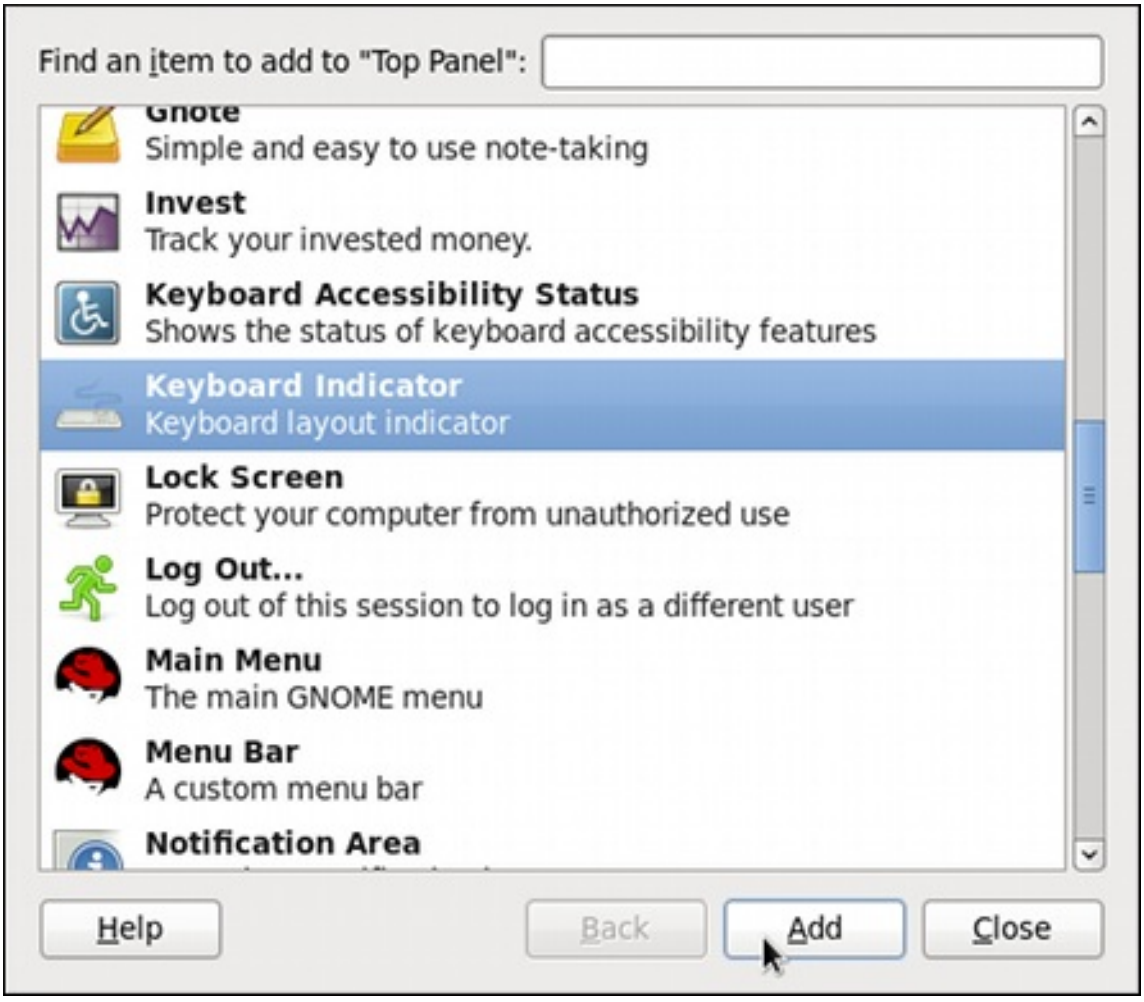


図1.6 キーボード表示器 の選択

アプレットは直ちに表示され、現在のレイアウトと関連する短縮された国名が表示されます。実際の系列名を表示するには、アプレットのアイコン上にポインターを移動します。

図1.7 キーボード表示器 アプレット

1.3. 一休みの設定

長時間タイプし続けることは疲れるだけでなく、手根管症候群など深刻な健康問題のリスクを高める恐れもあります。これを防ぐ方法として、強制的に入力を休憩するようにシステムを設定することができます。設定方法は、パネルから **システム** → **設定** → **キーボード** の順に選択し、**一休み** タブをクリックし、**一休みに入ったら強制的に画面をロックする** のチェックボックスにチェックマークを入れます。

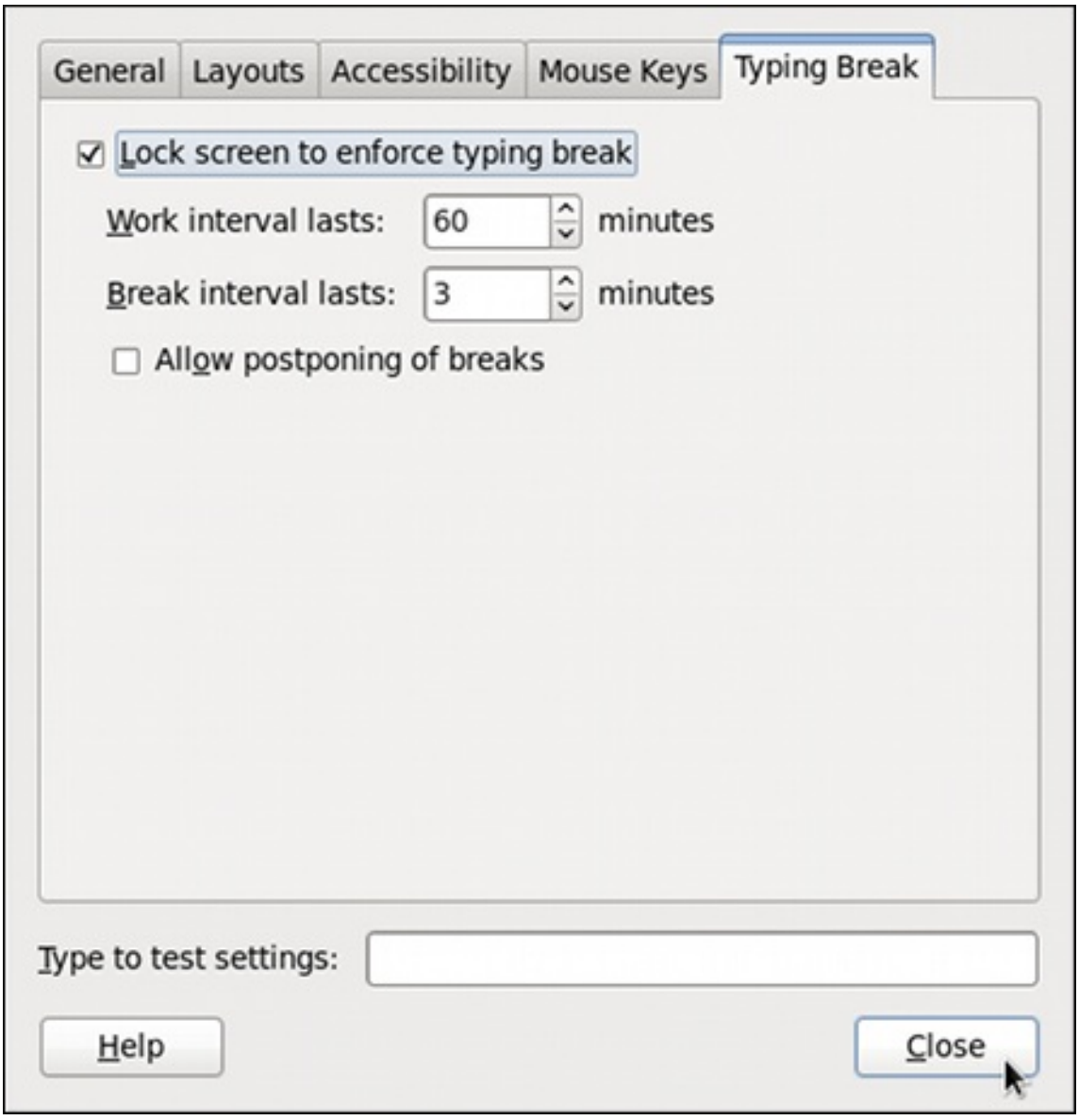


図1.8 一休みの設定

一休みの機能が強制実行されるまでの時間を調整するには、**一休みの警告を出すまでの時間** ラベルの横にある上向き/下向きボタンをクリックします。休憩時間の長さを変更するには、**一休みする時間** の設定を同じように変更します。最後に、作業を終わらせたい場合など休憩を先送りしたい場合には、**一休みの延長を許可する** チェックボックスにチェックマークを入れます。変更は直ちに反映されます。

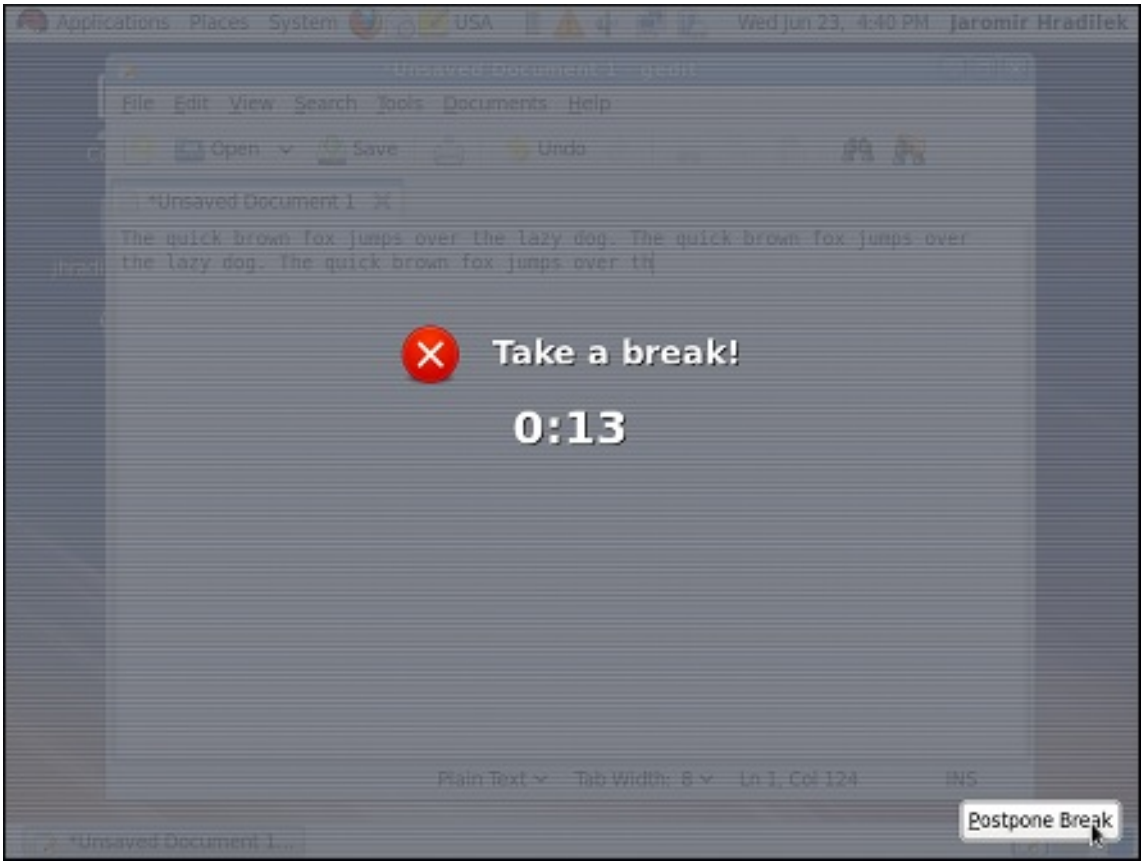


図1.9 休憩中

設定した制限時間に達すると、休憩するよう画面に通知が表示され、残りの休憩時間を表示した時計が現れます。 一休みの延長許可を有効にした場合は、**Postpone Break (中断を延長する)** ボタンが画面の右下隅に表示されます。

第2章 日付と時刻の設定

本章では Red Hat Enterprise Linux での手動、及び NTP (Network Time Protocol ネットワーク時刻プロトコル) を使用した、システムの日付と時刻の設定、さらには適切なタイムゾーンの設定について取り上げます。**Date/Time Properties (日付/時刻のプロパティ)** ツールを使用して日付と時刻を設定する方法、コマンドラインでそれを行う方法の 2 点について説明します。

2.1. 日付/時刻のプロパティのツール

日付/時刻のプロパティ ツールを使用すると、ユーザーはシステムの日付と時刻の変更、システムが使用するタイムゾーンの設定、タイムサーバーとシステムクロックが同期するようネットワーク時刻プロトコルデーモンの設定を行うことができます。このアプリケーションを使用するには、**X Window System** (このトピックの詳細については 付録C *X Window System* を参照) を実行している必要がある点に注意して下さい。

このツールを起動するには、パネルより **System (システム)** → **Administration (管理)** → **Date & Time (日付と時刻)** の順に選択するか、シェルプロンプト (**xterm** や **GNOME 端末**) で **`system-config-date`** のコマンドを入力します。すでに認証されてない限りは、スーパーユーザーのパスワードを入力するよう求められます。

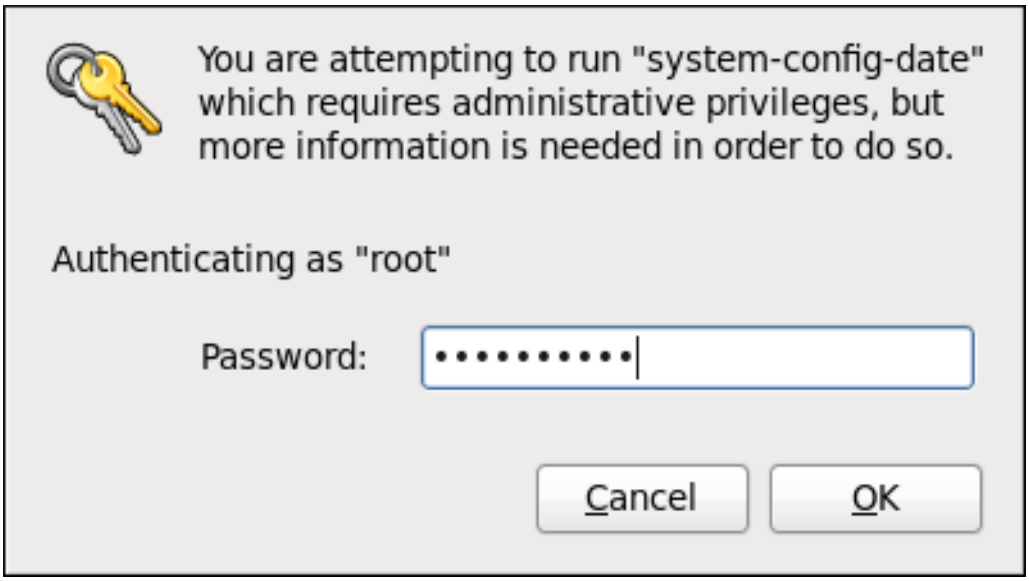


図2.1 認証クエリ

2.1.1. 日付と時刻のプロパティ

図2.2「日付と時刻のプロパティ」 のとおり、**日付/時刻のプロパティ** ツールにはタブが 2 つあります。デフォルトでは、現在の日時設定があるタブが表示されます。

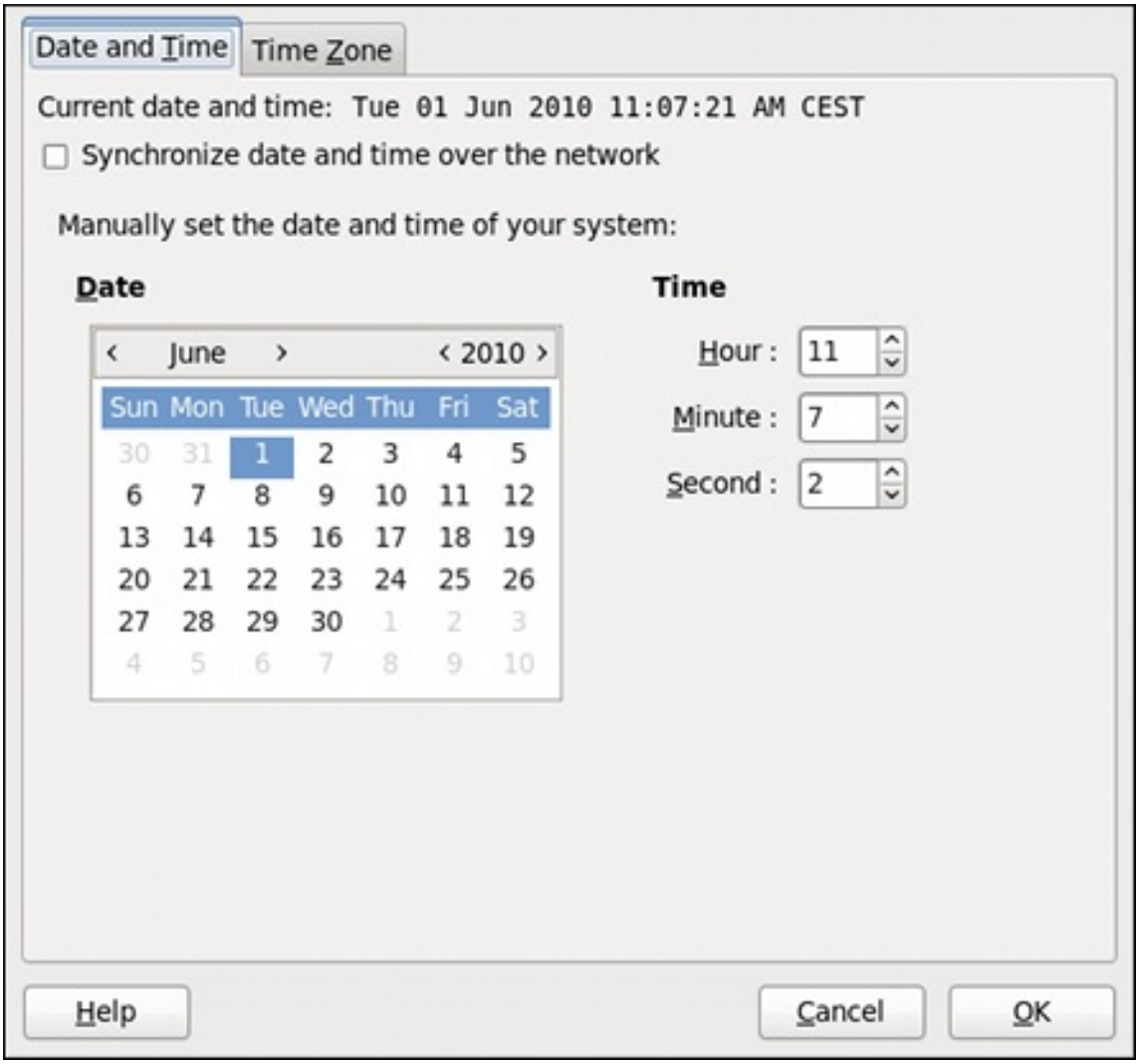


図2.2 日付と時刻のプロパティ

手動でシステムを設定するには、以下の手順に従って下さい:

1. **現在の日付を変更する。** 月と年の左右にある矢印を使って、それぞれ変更します。次に、カレンダーの中で日付を選択します。

2. **現在の時刻を変更する。** **時**、**分**、**秒** の横にある上下の矢印ボタンを使用するか、数字を直接変更します。

OK ボタンをクリックして変更を適用し、アプリケーションを終了します。

2.1.2. ネットワーク時刻プロトコルのプロパティ

自動設定にしたい場合は、**Synchronize date and time over the network (ネットワーク上で日付と時刻を同期化します)** というチェックボックスにチェックマークを入れて下さい。そうすると、利用可能な NTP サーバーの一覧が 図2.3「ネットワーク時刻プロトコルのプロパティ」 のように表示されます。

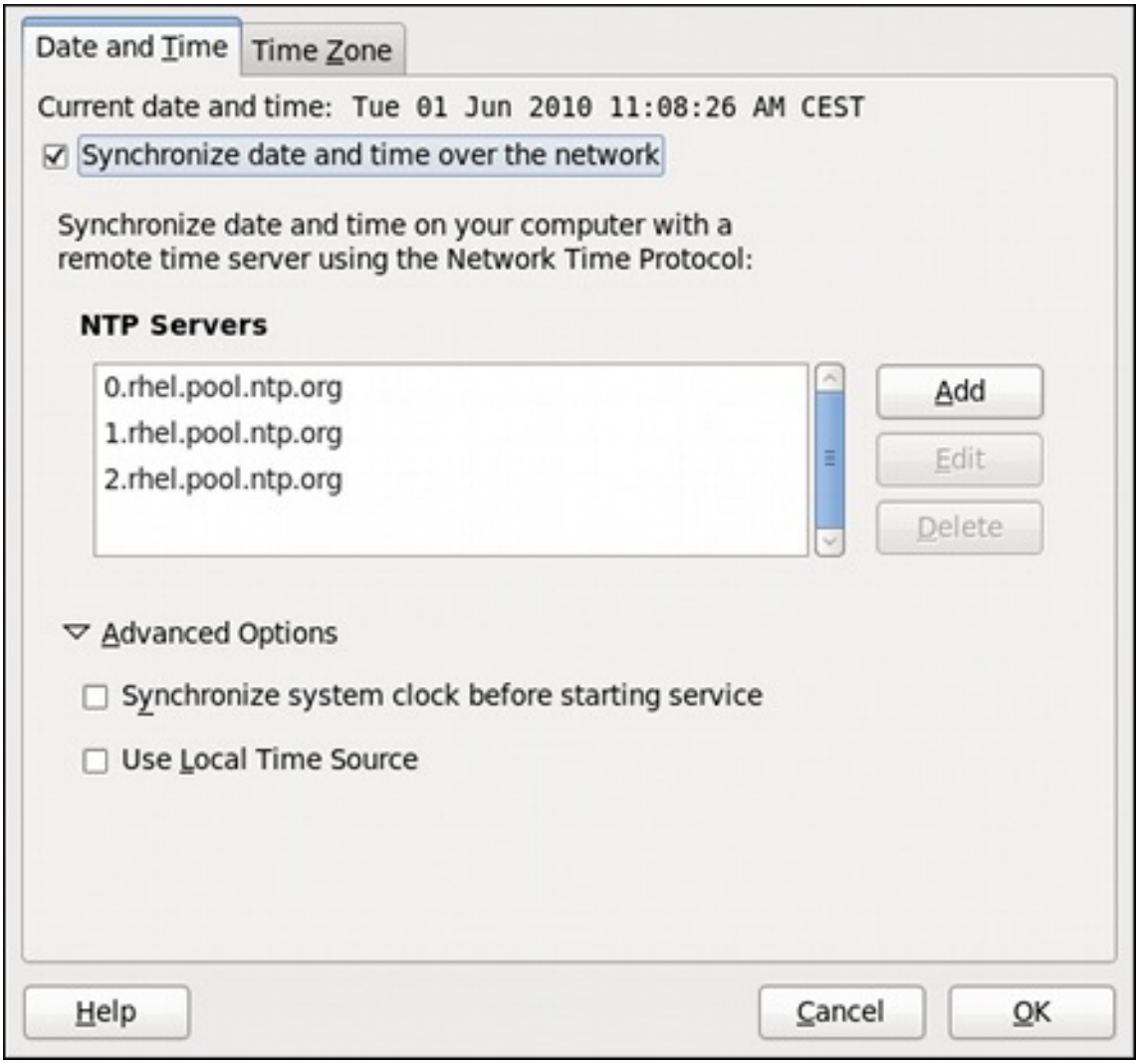
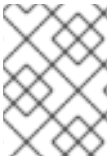


図2.3 ネットワーク時刻プロトコルのプロパティ

ここで、事前定義済みのサーバーから 1 つ選択し、**編集** ボタンをクリックして編集するか、**追加** をクリックして新規サーバー名を追加することも可能です。**Advanced Options (高度なオプション)** では、サービスを開始する前にシステムクロックを同期したいか、ローカルタイムソースを使用したいかどうかを選択することもできます。



注記

ウィンドウの最下部にある **OK** ボタンをクリックして変更を確定するまで、システムは NTP サーバーとの同期を開始しません。

OK ボタンをクリックして、日付と時刻の設定への変更を適用して、アプリケーションを終了します。

2.1.3. タイムゾーンのプロパティ

システムのタイムゾーンを設定するには、 図2.4「タイムゾーンのプロパティ」 のように **Time Zone (タイムゾーン)** タブをクリックします。

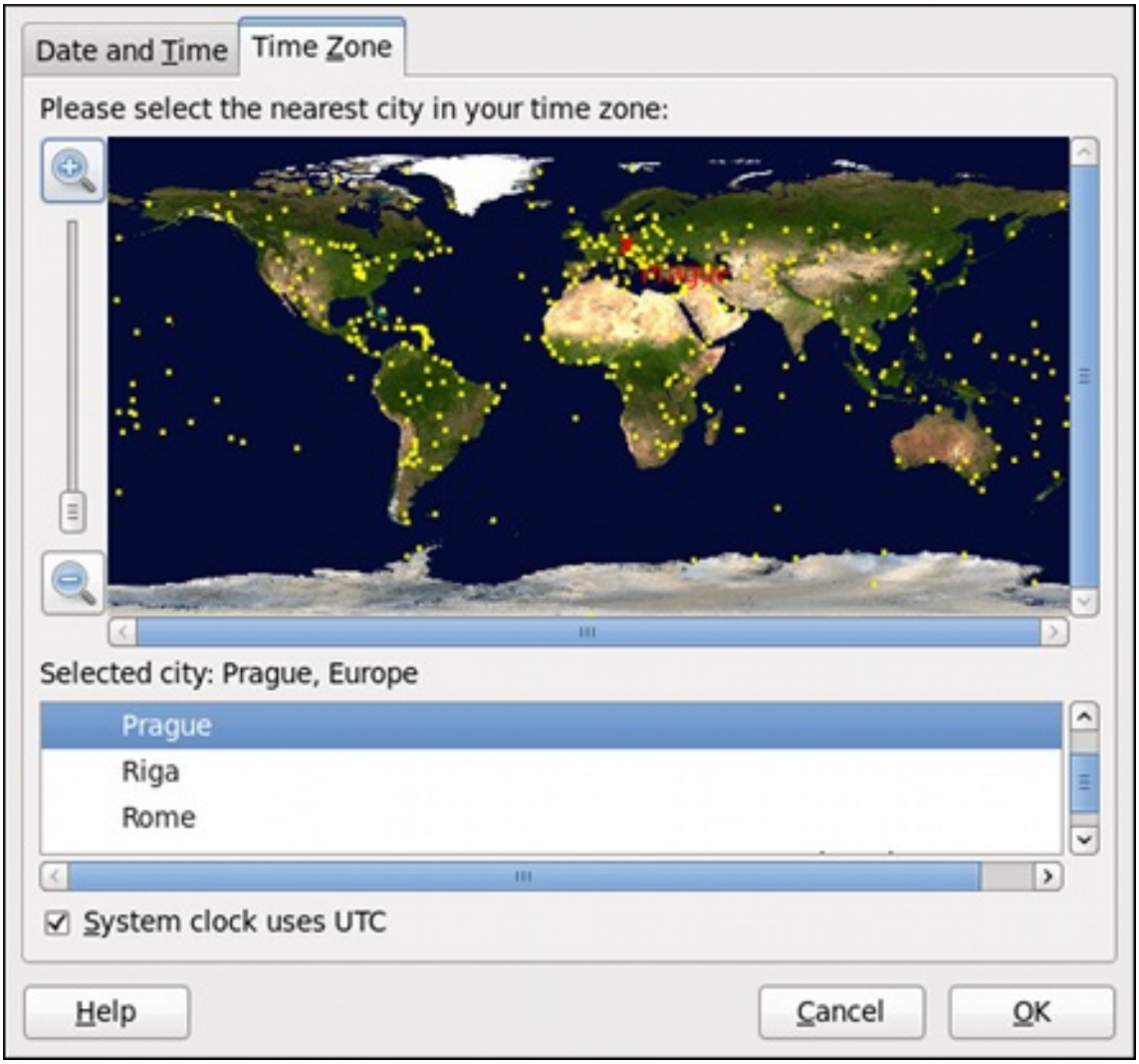


図2.4 タイムゾーンのプロパティ

タイムゾーンを選択する一般的な方法としては、以下の 2 つがあります。

1. **インタラクティブマップを使用する。** マップの横にある “拡大”/“縮小” ボタンをクリックします。あるいは、マップ上でクリックして、選択した地域を拡大します。次に、希望するタイムゾーンに合う都市を選択します。赤い **X** が表示され、マップの下にある一覧内のタイムゾーンの選択肢が変更します。

2. **マップの下にある一覧を使用する。** 選択しやすいように特定の大陸内で都市と国がグループ化されています。科学界のニーズに合うように、地理的ではないタイムゾーンも追加されている点に注意して下さい。

ご使用のシステムクロックが UTC を使用するよう設定するには、**System clock uses UTC (システムクロックで UTC を使用)** オプションを選択します。UTC は *Universal Time, Coordinated (協定世界時)* の略で、GMT(*グリニッジ標準時*) とも呼ばれます。他のタイムゾーンは UTC 時間にプラス/マイナスすることで割り出されます。

OK をクリックし変更を適用して、プログラムを終了します。

2.2. コマンドラインからの設定

ご使用のシステムに **日付/時刻のプロパティ** ツールがインストールされていない場合や **X Window Server** が実行していない場合は、コマンドラインでシステムの日時を変更する必要があります。本項で説明する操作を行うには、スーパーユーザーとしてログインする必要がある点にご注意下さい:

```
~]$ su -
Password:
```

2.2.1. 日付と時刻の設定

スーパーユーザーは、**date** コマンドを使ってシステムの日時を手動で設定できます。

1. **現在の日付を変更する** ためには、シェルプロンプトで次の形式でコマンドを入力します。*YYYY* は 4 桁の年、*MM* は 2 桁の月、*DD* は 2 桁の日付に置き換えて下さい:

   ```
   ~]# date +%D -s YYYY-MM-DD
   ```

   例えば 2010 年 6 月 2 日と設定するには、以下のように入力します:

   ```
   ~]# date +%D -s 2010-06-02
   ```

2. **現在の時刻を変更する** ためには、次のコマンドを使用します。*HH* は時間、*MM* は分、*SS* は秒を表し、すべて 2 桁で入力します:

   ```
   ~]# date +%T -s HH:MM:SS
   ```

   システムクロックが UTC を使用するように設定するには、次のオプションを追加します:

   ```
   ~]# date +%T -s HH:MM:SS -u
   ```

   例えば UTC を使用してシステムクロックを 11:26 PM に設定するには、以下のように入力します:

   ```
   ~]# date +%T -s 23:26:00 -u
   ```

現在の設定を確認するには、引数なしで **date** と入力して下さい:

例2.1 現在日時の表示

```
~]$ date
Wed Jun  2 11:58:48 CEST 2010
```

2.2.2. ネットワーク時刻プロトコルの設定

前述した手動設定とは対照的に、ネットワーク時刻プロトコル (NTP) でリモートサーバーとシステムクロックを同期することもできます。1 回だけの同期の場合は、**ntpdate** コマンドを使用します。

1. 最初に、選択した NTP サーバーがアクセス可能か確認します:

   ```
   ~]# ntpdate -q server_address
   ```

   例:

   ```
   ~]# ntpdate -q 0.rhel.pool.ntp.org
   ```

2. 適切なサーバーが見つかったら、サーバーアドレスを 1 つ以上 **ntpdate** コマンドの後に付けて実行します:

```
~]# ntpdate server_address...
```

例えば以下のようにします:

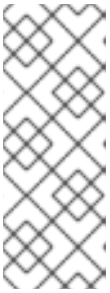
```
~]# ntpdate 0.rhel.pool.ntp.org 1.rhel.pool.ntp.org
```

エラーメッセージが表示されない限り、システムの時刻は設定されているはずです。「日付と時刻の設定」 のように追加の引数を付けずに **date** と入力すると、現在の設定を確認できます。

3. ほとんどの場合、上記のステップで十分です。常に正確な時刻を使用するシステムサービスが 1 つ以上必要な場合にのみ、ブート時の **ntpdate** の実行を有効にします:

```
~]# chkconfig ntpdate on
```

システムサービスとその設定の詳細については、10章 *サービスとデーモン* を参照して下さい。



注記

/var/log/boot.log システムログに関連するエラーメッセージが出るなど、ブート時にタイムサーバーとの同期が失敗し続ける場合は、次の行を **/etc/sysconfig/network** に追加してみて下さい:

```
NETWORKWAIT=1
```

ただし、ブート時に時刻を自動的に同期するためのより便利な方法は **ntpd** デーモンを設定することです。

1. **vi** や **nano** などのテキストエディターで NTP 設定ファイル **/etc/ntp.conf** を開きます。該当ファイルがない場合は、新しく作成して下さい:

```
~]# nano /etc/ntp.conf
```

2. 次に、パブリック NTP サーバーの一覧を追加/編集します。Red Hat Enterprise Linux 6 を使用している場合は、次の行がファイルにすでに含まれているはずですが、ニーズに合うように自由に追加/変更して下さい:

```
server 0.rhel.pool.ntp.org
server 1.rhel.pool.ntp.org
server 2.rhel.pool.ntp.org
```

注記

初期同期を迅速にするには、サーバーの各行の最後に **iburst** 指示文を追加します:

```
server 0.rhel.pool.ntp.org iburst
server 1.rhel.pool.ntp.org iburst
server 2.rhel.pool.ntp.org iburst
```

3. サーバーの一覧での作業が完了したら、同じファイルで適切なパーミッションを設定し、ローカルホストのみに無制限アクセスを付与します:

```
restrict default kod nomodify notrap nopeer noquery
restrict -6 default kod nomodify notrap nopeer noquery
restrict 127.0.0.1
restrict -6 ::1
```

4. すべての変更を保存し、エディターを終了してから NTP デーモンを再起動します:

```
~]# service ntpd restart
```

5. **ntpd** デーモンがブート時に開始していることを確認して下さい:

```
~]# chkconfig ntpd on
```

第3章 ユーザーとグループの管理

ユーザーとグループの制御は、Red Hat Enterprise Linux システムと管理の中核となる要素です。本章は、グラフィカルユーザーインターフェース及びコマンドライン上でユーザーとグループを追加/管理/削除する方法について説明し、パスワードエージングの有効化やグループディレクトリの作成などの高度なトピックを扱います。

3.1. ユーザーとグループの概要

ユーザーは、人 (実在のユーザーに関連するアカウントの意味) か、使用する特定のアプリケーション用に存在するアカウントであるのに対して、グループは共通目的のためにユーザーを統合して組織を論理的に表現したものです。グループ内のユーザーは、そのグループが所有するファイルの読み取り/書き込み/実行を行うことができます。

各ユーザーは、*ユーザー ID* (UID) と呼ばれる一意の数値型の識別番号に関連付けられています。同様に、各グループは *グループ ID* (GID) に関連付けられています。ファイルを作成するユーザーは、そのファイルの所有者でありグループ所有者でもあります。ファイルには所有者、グループ、その他に対して読み取り/書き込み/実行のパーミッションが別々に割り当てられます。ファイルの所有者は **root** のみ変更することができます。アクセスパーミッションは **root** ユーザーとファイルの所有者とも変更できます。

さらに、Red Hat Enterprise Linux はファイルとディレクトリに対する *アクセス制御リスト* (ACL) をサポートしています。これにより、所有者以外の特定のユーザーにパーミッションを設定することができます。この機能の詳細については、『Storage Administration Guide (ストレージ管理ガイド)』 の 『Access Control Lists (アクセス制御リスト)』 の章を参照して下さい。

3.1.1. ユーザープライベートグループ

Red Hat Enterprise Linux では、*UPG* (*ユーザープライベートグループ*) スキームを使用するため、UNIX グループを簡単に管理することができます。ユーザープライベートグループは、新規ユーザーがシステムに追加される度に作成されます。ユーザープライベートグループは作成されたユーザーと同じ名前を持ち、そのユーザーがそのユーザープライベートグループの唯一のメンバーです。

ユーザープライベートグループを使用すると、新規作成されたファイルやディレクトリに対し安全にデフォルトのパーミッションを設定することができます。また、ユーザー及び **そのユーザーのグループ** 双方とも、ファイルやディレクトリを修正できます。

新規作成されたファイルやディレクトリに適用するパーミッションを決定する設定は、*umask* と呼ばれる **/etc/bashrc** ファイルで設定されます。従来 UNIX システムでは、**umask** は **022** に設定されており、ファイルやディレクトリを作成したユーザーのみが修正できるようになっています。このスキームでは、**作成者のグループのメンバーを含め** 他のどのユーザーも修正を行うことはできません。一方、UPG スキームではどのユーザーもそれぞれ独自のプライベートグループを持っているため、この 「グループ保護」 は必要ありません。

3.1.2. シャドウパスワード

マルチユーザー環境では、shadow-utils パッケージにより提供される *シャドウパスワード* を使用することが非常に重要です。これを使用することで、システムの認証ファイルのセキュリティを強化できます。このため、インストールプログラムによりシャドウパスワードはデフォルト設定で有効になっています。

以下は、UNIX ベースシステムでパスワードを格納する従来の方法と比べた場合のシャドウパスワードの利点です。

- シャドウパスワードは、暗号化されたパスワードハッシュをあらゆるユーザーから読み取り可能な **/etc/passwd** ファイルから **root** ユーザーのみが読み取り可能な **/etc/shadow** に移動することで、システムセキュリティを向上します。
- シャドウパスワードは、パスワードエージングに関する情報を保存します。
- シャドウパスワードを使用すると、**/etc/login.defs** ファイルはセキュリティポリシーを強制できます。

shadow-utils パッケージにより提供される大半のユーティリティは、シャドウパスワードが有効かどうかに関わらず適切に動作します。ただし、パスワードエージングの情報は **/etc/shadow** ファイルにのみ格納されているため、パスワードエージングの情報を作成/修正するコマンドはいずれも機能しません。以下は、最初にシャドウパスワードを有効にしなければ機能しないユーティリティとコマンドの一覧です。

- **chage** ユーティリティ
- **gpasswd** ユーティリティ
- **-e** または **-f** オプションを付けた **usermod** コマンド
- **-e** または **-f** オプションを付けた **useradd** コマンド

3.2. ユーザー管理ツールの使用

ユーザー管理 のアプリケーションを使用すると、グラフィカルユーザーインターフェース上でローカルユーザー及びグループを表示/修正/追加/削除することができます。**ユーザー管理** を起動するには、パネルから **システム** → **管理** → **ユーザーとグループ** の順にクリックします。別の方法として、シェルプロンプトで **system-config-users** と入力することも可能です。スーパーユーザー権限を持っていない場合は、アプリケーションは **root** として認証するようプロンプトします。

3.2.1. ユーザーとグループの表示

ユーザー管理 の主要ウィンドウは、2 つのタブに分かれています。**ユーザー** タブは、ローカルユーザーのユーザー ID、プライマリグループ、ホームディレクトリ、ログインシェル、氏名などの追加情報を含んだローカルユーザーの一覧を表示します。**グループ** タブは、ローカルグループのグループ ID とグループメンバーに関する情報を含むローカルグループの一覧を表示します。

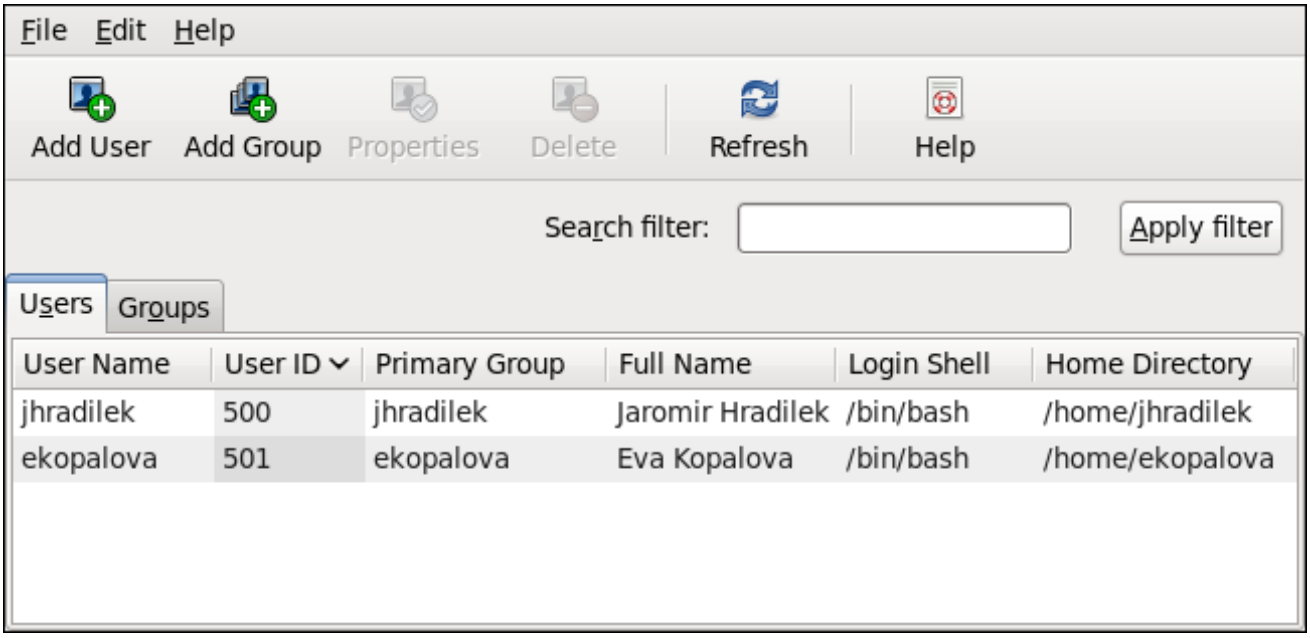


図3.1 ユーザーとグループの表示

特定のユーザーやグループを検索したい場合は、**フィルターの探索** フィールドに検索したい名前の最初の数文字を入力します。そして、**Enter** を押すか **フィルターの適用** ボタンをクリックします。また、いずれかのカラムのヘッダーをクリックするとそのカラムに従って項目を並び替えることもできます。

Red Hat Enterprise Linux は、システムユーザーとグループ用に 500 より小さい数字のユーザー ID とグループ ID を保有しています。デフォルトでは、**ユーザー管理** はシステムユーザーを表示しません。すべてのユーザーとグループを表示するには、**編集** → **設定** の順に選択して、**個人設定** のダイアログボックスを開きます。そして、**システムユーザーとグループを非表示** のチェックマークを外します。

3.2.2. 新規ユーザーの追加

新規ユーザーを追加するには、**ユーザーを追加** ボタンをクリックします。図3.2「新規ユーザーの追加」 のようなウィンドウが表示されます。

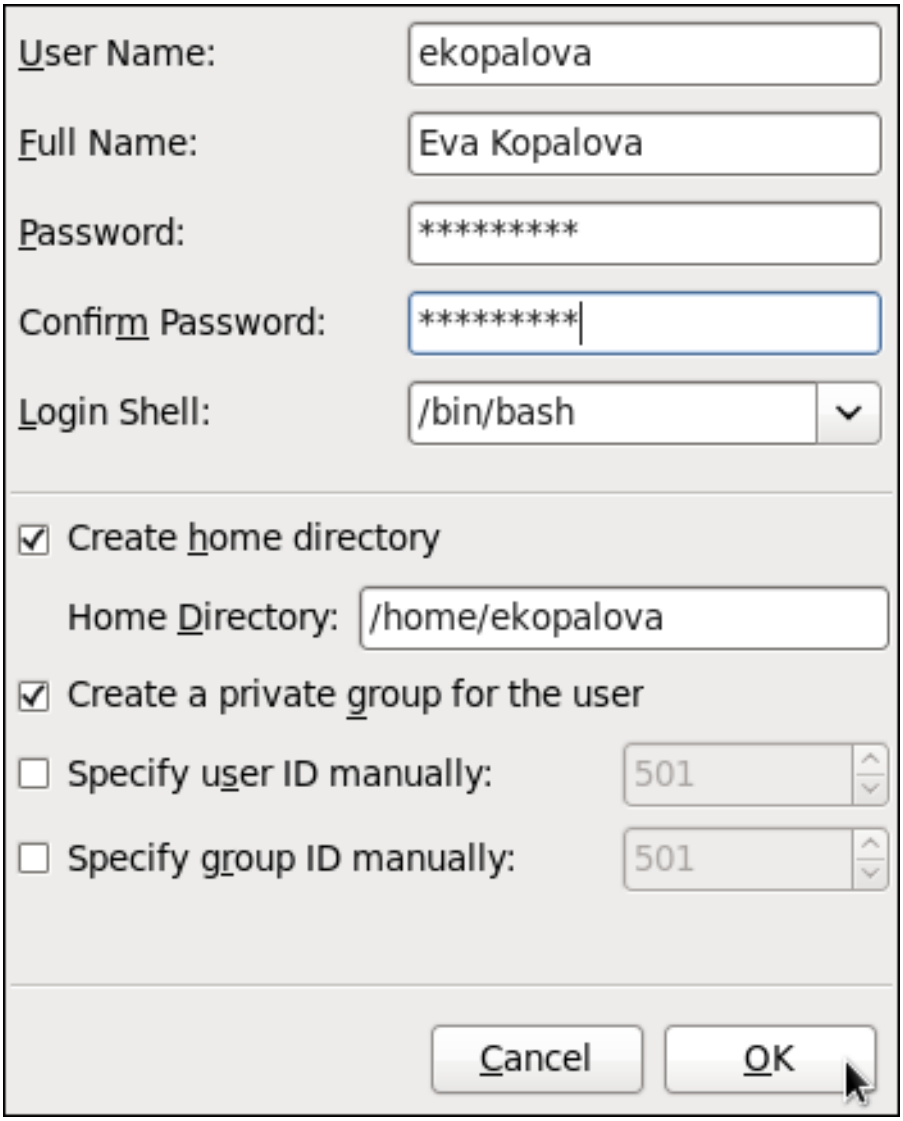


図3.2 新規ユーザーの追加

新規ユーザーの追加 ダイアログボックスを使用して、新規作成するユーザーに関する情報を入力できます。ユーザーを作成するには、該当フィールドにユーザー名と氏名を入力して、**パスワード** と **パスワードの確認** フィールドにユーザーのパスワードを入力します。パスワードは 6 文字以上でなくてはなりません。

注記

できるだけ長いパスワードを使用することをお勧めします。長いパスワードを使用することで、侵入者がパスワードを推測してパーミッションなくアカウントにアクセスしにくくできます。また、辞書に載っている単語をベースとするパスワードは避け、文字や数字、特殊文字を組み合わせることが推奨されます。

ログインシェル プルダウンの一覧から、ユーザー用のログインシェルを選択できます。どのシェルを選択すべきか不明な場合は、デフォルト値である **/bin/bash** を使用して下さい。

デフォルトでは、**ユーザー管理** アプリケーションは新規ユーザー用のホームディレクトリを **/home/*username*/** 内に作成します。ホームディレクトリを作成しないようにするには、**ホームディレクトリの作成** チェックボックスのチェックを外すか、**ホームディレクトリ** テキストボックスの内容を編集してディレクトリを変更します。ホームディレクトリが作成されると、デフォルトの設定ファイルが **/etc/skel/** ディレクトリからそこへコピーされます。

Red Hat Enterprise Linux は、ユーザープライベートグループ (UPG) スキームを使用します。新規ユーザーを作成する度に、デフォルトによりそのユーザーと同じ名前を持つ一意のグループが作成されます。このグループを作成したくない場合は、**ユーザー用にプライベートグループを作成** からチェックマークを外して下さい。

ユーザー用のユーザー ID を指定するには、**ユーザー ID を手動で指定** を選択します。そのオプションが選択されていない場合は、500 より大きい数字の使用可能なユーザー ID が新規ユーザーに割り当てられます。Red Hat Enterprise Linux は、システムユーザー用に 500 より小さい数字のユーザー ID を保有しているため、1 から 499 の数字のユーザー ID を手動で割り当てることは推奨されません。

OK ボタンをクリックすると、新規ユーザーが作成されます。パスワードの有効期限などさらに高度なユーザープロパティを設定するには、ユーザーの追加後にそのプロパティを編集します。

3.2.3. 新規グループの追加

新規ユーザーグループを追加するには、ツールバーから **グループを追加** を選択します。図3.3「新規グループ」 に似たウィンドウが表示されますので、新規グループの名前を入力します。そのグループ用のグループ ID を指定するには、**グループ ID を手動で指定** にチェックマークを入れて、GID を選択します。Red Hat Enterprise Linux はシステムグループ用に 500 より小さい数字のグループ ID を保有している点に注意して下さい。

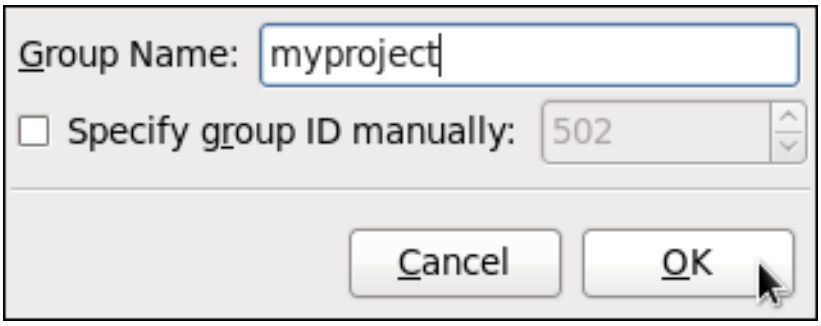


図3.3 新規グループ

OK をクリックするとグループが作成され、グループ一覧に表示されます。

3.2.4. ユーザープロパティの変更

既存ユーザーのプロパティを表示するには、**ユーザー** タブをクリックし、ユーザーの一覧から該当するユーザーをクリックし、メニューから **プロパティ** をクリック (またはプルダウンメニューから **ファイル** → **プロパティ** の順に選択) します。図3.4「ユーザープロパティ」 に似たウィンドウが表示されます。

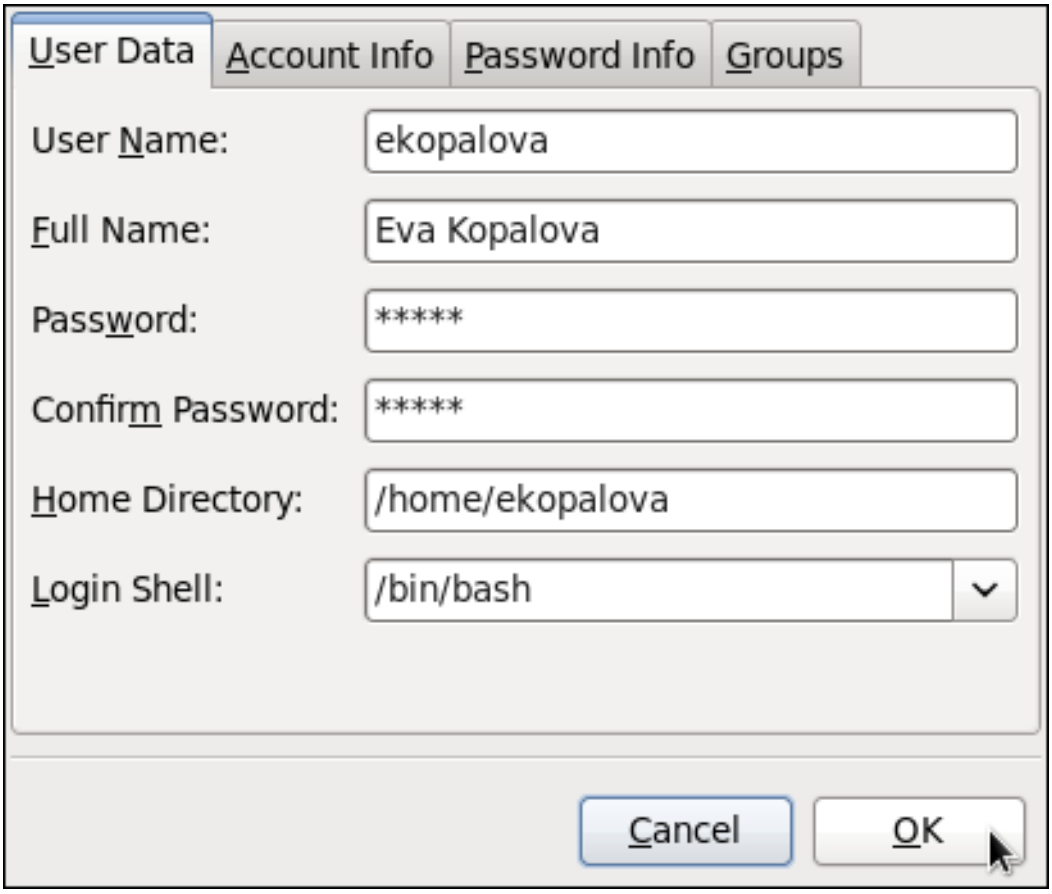


図3.4 ユーザープロパティ

ユーザープロパティ ウィンドウは複数のタブページに分かれています。

- **ユーザーデータ** — ユーザーの追加時に設定する基本情報を示しています。このタブを使用すると、ユーザーの氏名、パスワード、ホームディレクトリ、ログインシェルを変更できます。
- **アカウント情報** — 特定の日にアカウントを失効させたい場合は **アカウント失効を有効にする** を選択します。表示されたフィールドに日付を入力します。ユーザーアカウントをロックしてユーザーがシステムにログインできないようにするには、**ローカルパスワードがロックされています** にチェックマークを入れて下さい。
- **パスワード情報** — ユーザーのパスワードが最後に変更された日を表示します。一定の日数が経過した後にユーザーがパスワードを変更するよう強制的に設定するには、**パスワード失効を有効にする** を選択して、**変更が要求されるまでの日数** フィールドに希望する値を入力します。ユーザーのパスワードが失効するまでの日数、ユーザーがパスワードを変更するよう警告されるまでの日数、アカウントが無効になるまでの日数はすべて変更できます。
- **グループ** — ユーザーのプライマリグループ、さらにはユーザーをメンバーにしたい他のグループも表示/設定することができます。

3.2.5. グループプロパティの変更

既存グループのプロパティを表示するには、グループ一覧から該当するグループを選択し、メニューから **プロパティ** をクリックします (またはプルダウンメニューから **ファイル** → **プロパティ** の順に選択します)。図3.5「グループのプロパティ」 に似たウィンドウが表示されます。

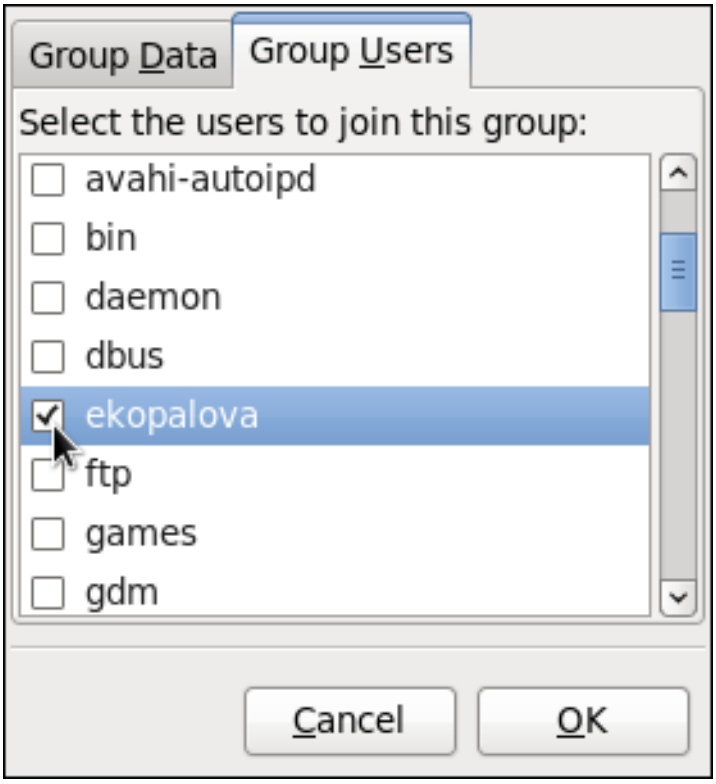


図3.5 グループのプロパティ

グループユーザー タブはグループのメンバーであるユーザーを表示します。このタブを使って、グループに対してユーザーの追加/削除を行います。**OK** をクリックし、変更を保存します。

3.3. コマンドラインツールの使用

Red Hat Enterprise Linux でユーザーを管理する最も簡単な方法は、「ユーザー管理ツールの使用」 で説明のとおり **ユーザー管理** アプリケーションを使用することです。しかし、コマンドラインツールを好む場合や X Window System がインストールされていない場合には、表3.1「ユーザーとグループを管理するためのコマンドラインユーティリティ」 に一覧表示されているコマンドラインユーティリティを使用することができます。

表3.1 ユーザーとグループを管理するためのコマンドラインユーティリティ

ユーティリティ	詳細
useradd, **usermod**, **userdel**	ユーザーアカウントを追加/修正/削除するための標準ユーティリティです。
groupadd, **groupmod**, **groupdel**	グループを追加/修正/削除するための標準ユーティリティです。
gpasswd	**/etc/group** 設定ファイルを管理するための標準ユーティリティです。
pwck, **grpck**	パスワード、グループ、関連シャドウファイルを検証するためのユーティリティです。

ユーティリティ	詳細
pwconv, **pwunconv**	通常のパスワードをシャドウパスワードに変換するため、または逆にシャドウパスワードから通常のパスワードに変換するためのユーティリティです。

3.3.1. 新規ユーザーの追加

システムにユーザーを追加するには、**root** としてシェルプロンプトで以下を入力します:

```
useradd [options] username
```

options は 表3.2「useradd のコマンドラインオプション」 で説明のとおり、コマンドラインのオプションです。

デフォルトでは、**useradd** コマンドはロックされたユーザーアカウントを作成します。アカウントのロックを解除するには、**root** として以下のコマンドを実行して、パスワードを割り当てます:

```
passwd username
```

オプションとして、パスワードエージングポリシーを設定することもできます。パスワードエージングを有効にする方法については、『Red Hat Enterprise Linux 6 Security Guide』 を参照してください。

表3.2 useradd のコマンドラインオプション

オプション	詳細
-c '*comment*'	*comment* はあらゆる文字列と置換できます。このオプションは、通常ユーザーの氏名を指定するために使用されます。
-d *home_directory*	デフォルトの **/home/*username*/** の代わりに使用するホームディレクトリです。
-e *date*	YYYY-MM-DD の形式でアカウントを無効にする日付です。
-f *days*	パスワードが失効してからアカウントが無効になるまでの日数です。**0** を指定すると、パスワードが失効した直後にアカウントは無効になります。**-1** を指定すると、パスワードが失効した後でもアカウントは無効になりません。
-g *group_name*	ユーザーのデフォルトグループ用のグループ名またはグループ番号です。グループはここで指定される以前から存在していなければなりません。
-G *group_list*	ユーザーがメンバーとなる追加 (デフォルト以外) のグループ名またはグループ番号の一覧で、カンマで区切られています。グループはここで指定される以前から存在していなければなりません。
-m	ホームディレクトリが存在しない場合は作成します。

オプション	詳細
-M	ホームディレクトリを作成しません。
-N	ユーザー用のユーザープライベートグループを作成しません。
-p *password*	**crypt** で暗号化されたパスワードです。
-r	UID が 500 より小さく、ホームディレクトリがないシステムアカウントを作成します。
-s	ユーザーのログインシェルです。デフォルトでは **/bin/bash** に設定されています。
-u *uid*	ユーザーのユーザー ID です。一意の番号で 499 より大きい数でなければなりません。

プロセスの説明

以下のステップは、シャドウパスワードが有効なシステム上で **useradd juan** コマンドを実行したときに発生する内容を解説したものです:

1. **/etc/passwd** に **juan** の新しい行が作成されます:

```
juan:x:501:501::/home/juan:/bin/bash
```

この行には以下の特徴があります。

- ユーザー名 **juan** で始まります。
- パスワードフィールドには **x** があり、システムがシャドウパスワードを使用していることを示しています。
- 499 より大きい数字の UID が作成されます。Red Hat Enterprise Linux では、500 より小さい数字の UID はシステム使用のために保有されています。これらはユーザー用には割り当てないことをお勧めします。
- 499 より大きい数字の GID が作成されます。Red Hat Enterprise Linux では、500 より小さい数字の GID はシステム使用のために保有されています。これらはユーザー用には割り当てないことをお勧めします。
- オプションの *GECOS* 情報は空白のままです。GECOS フィールドは、氏名や電話番号などユーザーの追加情報を提供するために使用されます。
- **juan** のホームディレクトリは **/home/juan/** に設定されています。
- デフォルトのシェルは **/bin/bash** に設定されています。

2. **juan** 用の新しい行が **/etc/shadow** に作成されます:

```
juan:!!:14798:0:99999:7:::
```

この行には以下の特徴があります。

- ユーザー名 **juan** で始まります。
- 2 つの感嘆符 (**!!**) が **/etc/shadow** ファイルのパスワードフィールドに表示され、アカウントがロックされます。

注記

暗号化されたパスワードが **-p** フラグを使用して渡される場合、**/etc/shadow** ファイル内のユーザー用の新しい行に置かれます。

- パスワードは有効期限なしに設定されています。

3. **juan** というグループ用の新しい行が **/etc/group** に作成されます:

```
juan:x:501:
```

ユーザーと同じ名前のグループは、*ユーザープライベートグループ* と呼ばれます。ユーザープライベートグループの詳細については 「ユーザープライベートグループ」 を参照して下さい。

/etc/group に作成された行には、以下の特徴があります。

- グループ名 **juan** で始まります。
- パスワードフィールドに **x** が表示され、システムがシャドウグループパスワードを使用していることを示しています。
- GID は **/etc/passwd** 内に表示されているユーザー **juan** の GID と一致します。

4. **juan** というグループ用の新しい行が **/etc/gshadow** に作成されます:

```
juan:!::
```

この行には以下の特徴があります。

- グループ名 **juan** で始まります。
- 1 つの感嘆符 (**!**) が **/etc/gshadow** ファイルのパスワードフィールドに表示され、グループがロックされます。
- その他のフィールドはすべて空白です。

5. ユーザー **juan** 用のディレクトリが **/home/** ディレクトリ内に作成されます:

```
~]# ls -l /home
total 4
drwx------. 4 juan juan 4096 Mar  3 18:23 juan
```

このディレクトリは、ユーザー **juan** とグループ **juan** が所有しています。ユーザー **juan** に ***のみ*** *読み取り*、*書き込み* 及び *実行* の権限が与えられています。その他のパーミッションはすべて拒否されます。

6. **/etc/skel/** ディレクトリ (デフォルトのユーザー設定を含む) 内のファイルが、新しい **/home/juan/** ディレクトリにコピーされます:

```
~]# ls -la /home/juan
total 28
drwx------. 4 juan juan 4096 Mar  3 18:23 .
drwxr-xr-x. 5 root root 4096 Mar  3 18:23 ..
-rw-r--r--. 1 juan juan   18 Jun 22  2010 .bash_logout
-rw-r--r--. 1 juan juan  176 Jun 22  2010 .bash_profile
-rw-r--r--. 1 juan juan  124 Jun 22  2010 .bashrc
drwxr-xr-x. 2 juan juan 4096 Jul 14  2010 .gnome2
drwxr-xr-x. 4 juan juan 4096 Nov 23 15:09 .mozilla
```

この時点で、**juan** と呼ばれるロックされたアカウントがシステム上に存在します。管理者は、これをアクティベートするには **passwd** コマンドを使用してアカウントにパスワードを割り当てる必要があります。パスワードエージングのガイドラインを設定するというオプションもあります。

3.3.2. 新規グループの追加

システムに新規グループを追加するには、**root** としてシェルプロンプトで以下を入力します:

```
groupadd [options] group_name
```

options は、表3.3「groupadd のコマンドラインオプション」 で説明のとおりコマンドラインオプションです。

表3.3 groupadd のコマンドラインオプション

オプション	詳細
-f, **--force**	**-g** *gid* と共に使用され *gid* がすでに存在している場合、**groupadd** はグループ用に別の一意の *gid* を選択します。
-g *gid*	グループのグループ ID です。一意の番号で 499 より大きい数でなければなりません。
-K, **--key** *key=value*	**/etc/login.defs** のデフォルトを上書きします。
-o, **--non-unique**	グループの複製を許可します。
-p, **--password** *password*	新規グループ用にこの暗号化されたパスワードを使用します。
-r	GID が 500 より小さいシステムグループを作成します。

3.3.3. グループディレクトリの作成

システム管理者は、通常それぞれの主要なプロジェクトごとにグループを作成し、そのプロジェクトのファイルにアクセスする必要がある場合に人をグループに割り当てる傾向があります。こうした従来型のスキームの場合、誰かがファイルを作成すると、ファイルはその人が属するプライマリグループに関連付けられるため、ファイル管理は困難です。1 人が複数のプロジェクトに関わっている場合、正しいファイルを正しいグループに関連付けることは難しくなります。一方で、UPG スキームを使用すると、グループは *setgid* ビットセットを持つディレクトリ内に作成されたファイルに自動的に割り当てら

れます。setgid ビットにより、共通のディレクトリを共有するグループプロジェクトを非常に簡単に管理することができます。これは、ユーザーがディレクトリ内で作成するすべてのファイルは、ディレクトリを所有するグループが所有するためです。

例えば、あるグループが **/opt/myproject/** ディレクトリ内のファイルで作業する必要があるとします。グループの中にはこのディレクトリのコンテンツの修正を信頼して任せられる人もいますが、全員ではありません。

1. **root** としてシェルプロンプトで以下を入力して、**/opt/myproject/** ディレクトリを作成します:

```
mkdir /opt/myproject
```

2. システムに **myproject** グループを追加します:

```
groupadd myproject
```

3. **/opt/myproject/** ディレクトリのコンテンツと **myproject** グループを関連付けます:

```
chown root:myproject /opt/myproject
```

4. ユーザーがディレクトリ内のファイルを作成できるよう、setgid ビットを設定します:

```
chmod 2775 /opt/myproject
```

この時点で、ユーザーが新規ファイルの書き込みをする度に管理者はファイルのパーミッションを変更する必要なく、**myproject** グループの全メンバーは **/opt/myproject/** ディレクトリ内のファイルを作成/編集することができます。パーミッションが正しく設定されていることを確認するには、以下のコマンドを実行します:

```
~]# ls -l /opt
total 4
drwxrwsr-x. 3 root myproject 4096 Mar  3 18:31 myproject
```

3.4. その他のリソース

ユーザーとグループの管理に関する詳細情報については、以下のリソースを参照して下さい。

3.4.1. インストールされているドキュメント

ユーザーとグループを管理するための各種ユーティリティの詳細については、以下の man ページを参照して下さい:

- **chage**(1) — パスワードエージングのポリシーとアカウントの有効期限を修正するコマンドです。
- **gpasswd**(1) — **/etc/group** ファイルを管理するコマンドです。
- **groupadd**(8) — グループを追加するコマンドです。
- **grpck**(8) — **/etc/group** ファイルを検証するコマンドです。

- **groupdel**(8) — グループを削除するコマンドです。
- **groupmod**(8) — グループのメンバーシップを修正するコマンドです。
- **pwck**(8) — **/etc/passwd** 及び **/etc/shadow** ファイルを検証するコマンドです。
- **pwconv**(8) — 通常のパスワードをシャドウパスワードに変換するツールです。
- **pwunconv**(8) — シャドウパスワードを通常のパスワードに変換するツールです。
- **useradd**(8) — ユーザーを追加するコマンドです。
- **userdel**(8) — ユーザーを削除するコマンドです。
- **usermod**(8) — ユーザーを修正するコマンドです。

関連する設定ファイルの詳細については、以下をご覧下さい:

- **group**(5) — システムのグループ情報を含むファイルです。
- **passwd**(5) — システムのユーザー情報を含むファイルです。
- **shadow**(5) — システムのパスワードとアカウントの有効期限に関する情報を含むファイルです。

第4章 権限の取得

システム管理者は (ユーザーも時には) 管理者アクセスでタスクを実行する必要があります。システムに **root** でアクセスすることは危険を伴う場合やシステムおよびデータの大幅な破損につながる可能性もあります。この章では、**su** や **sudo** といった setuid プログラムを使用して管理者権限を取得する方法を説明します。これらのプログラムを使うと、特定のユーザーが高レベルの制御およびシステムセキュリティーを維持しつつ、通常は root ユーザーしかできないタスクを実行することができます。

管理者制御や潜在的な危険、権限者アクセスの不適切な使用によるデータ破損の回避方法についての詳細は、『Red Hat Enterprise Linux 6 セキュリティガイド』 を参照してください。

4.1. SU コマンド

ユーザーは、**su**を実行すると root パスワードを求められ、認証後に root shell プロンプトが表示されます。

su コマンドでログインした後は、そのユーザーは root ユーザーとなり、システムへの絶対管理アクセスを持つことになります[1]。さらに、root になるとユーザーは **su** コマンドを使って、パスワードを求められることなくシステム上の他のユーザーに変わることができます。

このプログラムは非常に強力なので、組織内の管理者はこのコマンドにアクセスできる人を制限してください。

簡単な制限方法は、*wheel* と呼ばれる特別な管理グループにユーザーを追加することです。これを実行するには、以下のコマンドを root で入力します。

```
usermod -G wheel <username>
```

このコマンドで *<username>* の部分を **wheel** グループに追加したいユーザーに置き換えます。

また、**User Manager** を使って以下のようにグループのメンバーを修正することもできます。この手順を実行するには、管理者権限が必要なことに注意してください。

1. パネル上の **システム** メニューをクリックして **管理** から **ユーザーとグループ** をクリックして User Manager を表示させます。別の方法では、shell プロンプトで **system-config-users** のコマンドを入力します。
2. **ユーザー** タブをクリックして、ユーザーリストの中から必要なユーザーを選択します。
3. ツールバーの **設定** をクリックして、ユーザー設定のダイアログボックスを表示させます (または **ファイル** メニューで **設定** を選択します) 。
4. **グループ** タブをクリックし、wheel グループのチェックボックスにチェックマークを付けて **OK** をクリックします。

User Manager の詳細に関しては、「ユーザー管理ツールの使用」 を参照してください。

wheel グループにユーザーを追加した後は、この追加した特定のユーザーのみに **su** コマンドの使用を許可することが推奨されます。それには、**su**: **/etc/pam.d/su** で PAM 設定ファイルを編集する必要があります。このファイルをテキストエディターで開き、以下の行からコメント (#) を削除します。

```
#auth            required        pam_wheel.so use_uid
```

この変更で、**wheel** の管理グループメンバーのみが **su** コマンドを使って別のユーザーに変わることができるようになります。

注記

root ユーザーはデフォルトで **wheel** グループの一部となっています。

4.2. SUDO コマンド

sudo コマンドを使うことで、ユーザーに管理アクセスを付与する別のアプローチができます。信頼できるユーザーが **sudo** で管理コマンドに優先する場合、このユーザーは **ユーザー自身の** パスワードを要求されます。ユーザーが認証され、コマンドが許可されると、管理コマンドは root ユーザーであるかのように実行されます。

sudo コマンドの基本的なフォーマットは、以下の通りです。

```
sudo <command>
```

上記の例では、*<command>* の部分は、通常 root ユーザーのみが使用する **mount** といったコマンドに置き換えます。

sudo コマンドでは、ハイレベルの柔軟性が可能になります。例えば、**/etc/sudoers** 設定ファイルに記載されているユーザーのみが **sudo** コマンドを使うことができ、root シェルではなく、**ユーザーの** シェルでコマンドが実行されます。つまり、『Red Hat Enterprise Linux 6 セキュリティガイド』 にあるように、root シェルは完全に無効化できるのです。

sudo を使った正常な認証はすべて **/var/log/messages** ファイルに記録され、この発行者のユーザー名で発行されたコマンドは、**/var/log/secure** ファイルに記録されます。新たなログが必要な場合は、以下の行を **/etc/pam.d/system-auth** ファイルに追加して、**pam_tty_audit** モジュールで特定ユーザーの TTY 監査を有効にします。

```
session required pam_tty_audit.so disable=<pattern> enable=<pattern>
```

pattern で表示されるのは、コンマで区切ったユーザーのリストで、グロブをオプションで使います。例えば、以下の設定は、root ユーザーのTTY 監査を有効にし、その他のユーザーについては無効にします。

```
session required pam_tty_audit.so disable=* enable=root
```

sudo コマンドのもう一つの利点は、異なるユーザーのニーズに応じて特定のコマンドへのアクセスを管理者が許可できることです。

sudo 設定ファイルである **/etc/sudoers** の編集を希望する管理者は、**visudo** コマンドを使用することが推奨されます。

完全な管理権限を誰かに付与する場合は、**visudo** を入力し、ユーザー権限特定セクションに以下と同様の行を追加します。

```
juan ALL=(ALL) ALL
```

この例では、ユーザーの **juan** はどのホストからでも **sudo** を実行して、どんなコマンドも実行できます。

以下の例では、**sudo** を設定する際に可能な粒度を示しています。

```
%users localhost=/sbin/shutdown -h now
```

この例が示しているのは、コンソールからであれば、どのユーザーでも **/sbin/shutdown -h now** コマンドを発行でできるということです。

sudoers の man ページにはこのファイルのオプションの詳細なリストが記載されています。

重要

sudo コマンドの使用時には、潜在的なリスクがいくつかあることを覚えておく必要があります。このリスクは、上記のように **visudo** を使用して **/etc/sudoers** 設定ファイルを編集することで回避できます。**/etc/sudoers** ファイルをデフォルトの状態にしておくと、**wheel** グループのユーザー全員に無制限の root アクセスを与えることになります。

- **sudo** はデフォルトで、sudo コマンド使用者のパスワードをタイムアウトの 5 分間、保存します。この間にコマンドを続けて使うと、ユーザーはパスワードを要求されません。このため、ユーザーがログイン状態のままワークステーションを離れたりロックしない状態にしておくと、攻撃者に悪用されかねません。この動作は、以下の行を **/etc/sudoers** に追加することで変更できます。

  ```
  Defaults    timestamp_timeout=<value>
  ```

  <value> には、指定するタイムアウトの分数を入れます。*<value>* を 0 にすると **sudo** は毎回パスワードを要求します。

- sudo 使用者のアカウントが侵害されると、攻撃者は **sudo** を使って管理権限のある新たな shell を開くことができます。

  ```
  sudo /bin/bash
  ```

  この方法や同様の方法で root として新たな shell を開くと、**/etc/sudoers** ファイルで指定されたタイムアウト時間を無視し、新たに開かれたセッションが閉じられるまで攻撃者に **sudo** パスワード入力を要求することがなく、理論上は攻撃者に時間無制限の管理アクセスを与えることになります。

4.3. その他のリソース

ユーザーに管理権限を与えるプログラムは潜在的なセキュリティーリスクである一方、セキュリティーそのものは本ガイドの対象外となります。このため、セキュリティーや権限のアクセスに関する情報に関しては、以下に挙げるリソースを参照してください。

インストールされているドキュメント

- **su**(1) - **su** の man ページにはこのコマンドで利用可能なオプションに関する情報があります。
- **sudo**(8) - **sudo** の man ページには、このコマンドに関する詳細な説明と、**sudo** の動作のカスタマイズで利用可能なオプションのリストがあります。
- **pam**(8) - man ページでは、 Linux 向け Pluggable Authentication Modules の使用方法が説明されています。

オンラインのドキュメント

- Red Hat Enterprise Linux 6 セキュリティガイド - 『セキュリティガイド』 setuid プログラムに関する潜在的なセキュリティー問題をより深く掘り下げるとともに、これらのリスクを緩和するテクニックを提供します。
- Red Hat Enterprise Linux 6 Managing Single Sign-On and Smart Cards - このガイドは主に、**Pluggable Authentication Modules (PAM)** とその設定、使用方法の詳細な説明を提供します。

---

[1] このアクセスは、SELinux の制限が有効な場合は、この制限対象となります。

パート II. パッケージ管理

Red Hat Enterprise Linux のシステム上にあるすべてのソフトウェアは RPM パッケージとして分割されています。これらはインストール、アップグレード、及び削除をすることができます。このセクションでは、プロダクトのサブスクリプションとエンタイトルメントに焦点を置き、**Yum** とグラフィカルパッケージ管理ツールの **PackageKit** スイートを使用した Red Hat Enterprise Linux に於けるパッケージ管理法を説明しています。

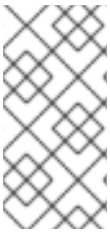
第5章 システム登録およびサブスクリプションの管理

資産管理を効果的に行うには、ソフトウェアインベントリ (ソフトウェアがインストールされている製品タイプとシステム数の両方) を処理するメカニズムが必要です。サブスクリプションサービスには、このメカニズムが実装されており、組織全体のサブスクリプションのグローバルな割り当てと、単一のシステムに割り当てられた固有のサブスクリプションとの両方に対する透過性を提供します。

Red Hat サブスクリプションマネージャーは **yum** と連動し、サブスクリプション管理とコンテンツ配信を結び付けます。サブスクリプションマネージャーが扱うのはサブスクリプションシステム関連のみです。**yum** または他のパッケージ管理ツールは実際のコンテンツ配信を扱います。**yum** の使用方法については、6章*Yum* で説明します。

5.1. RED HAT サブスクリプションマネージャーツールの使用

Red Hat サブスクリプションマネージャー と呼ばれるローカルシステム上の GUI および CLI ツールが登録とサブスクリプションの両方を管理します。



注記

システムに加える変更の性質上、Red Hat サブスクリプションマネージャーのツールは常に **root** で実行する必要があります。しかし、Red Hat サブスクリプションマネージャーは、サブスクリプションサービスのユーザーアカウントとしてサブスクリプションサービスに接続します。

5.1.1. Red Hat サブスクリプションマネージャー GUI の起動

Red Hat サブスクリプションマネージャーは、トップパネルの **システム => 管理** メニューに、管理ツールの一つとして表示されます。

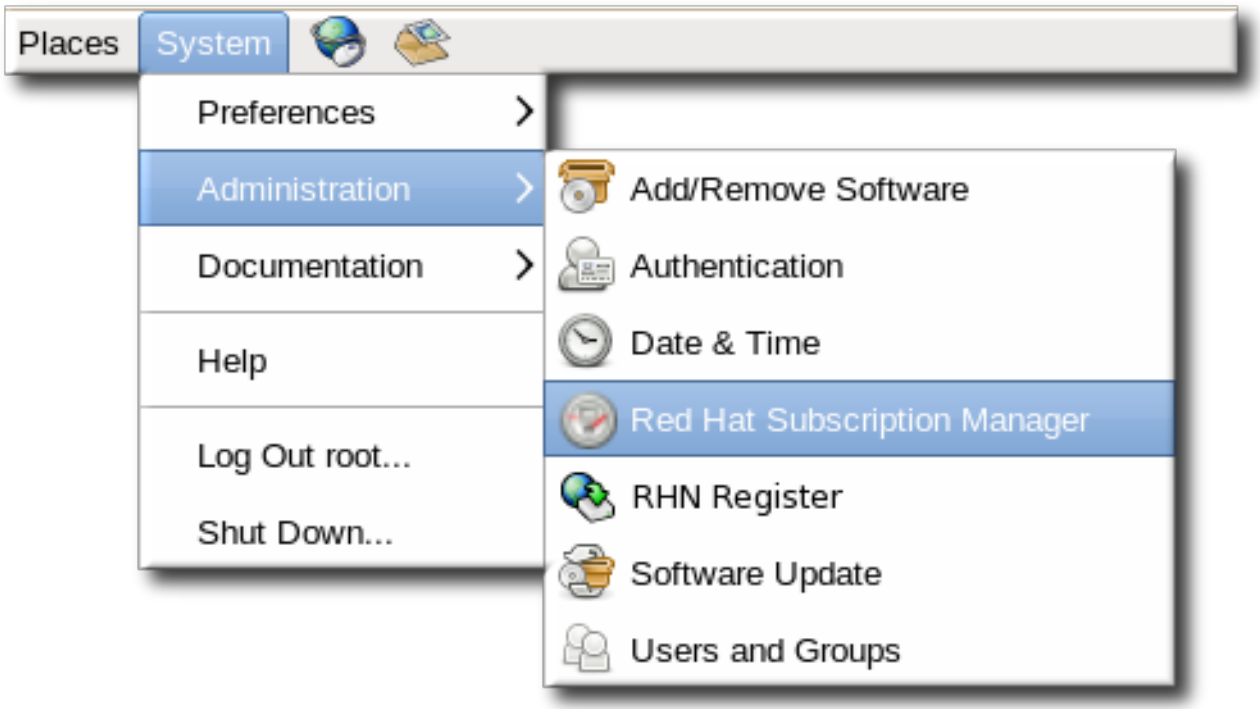


図5.1 Red Hat サブスクリプションマネージャーメニューオプション

また、単一のコマンドを使用して、コマンドラインから Red Hat サブスクリプションマネージャー GUI を開くこともできます:

```
[root@server1 ~]# subscription-manager-gui
```

5.1.2. サブスクリプションマネージャーのコマンドラインツールの実行

Red Hat サブスクリプションマネージャー UI を使って実行できる操作はすべて、**subscription-manager** ツールを使用しても実行できます。このツールは以下のフォーマットになります。

```
[root@server1 ~]# subscription-manager command [options]
```

各コマンドには、それぞれ独自の **オプション** セットを使用します。詳しくは、**subscription-manager** のヘルプと man ページを参照して下さい。

表5.1 一般的な subscription-manager コマンド

コマンド	詳細
register	新規システムをサブスクリプションサービスに対し登録または特定します。
unregister	マシンの登録解除を行います。サブスクリプションは解除され、サブスクリプションサービスからマシンが削除されます。
subscribe	マシンに特定のサブスクリプションをアタッチします。
redeem	ハードウェアおよび BIOS 情報に基づいて、ベンダーから購入した事前指定済みのサブスクリプションにマシンを自動的にアタッチします。
unsubscribe	マシンから特定のサブスクリプションまたはすべてのサブスクリプションを削除します。
list	マシンとの互換性があるすべてのサブスクリプションを一覧表示します。マシンに実際にアタッチされているサブスクリプションまたはマシンが利用可能な未使用のサブスクリプションのどちらかになります。

5.2. システムの登録および登録解除

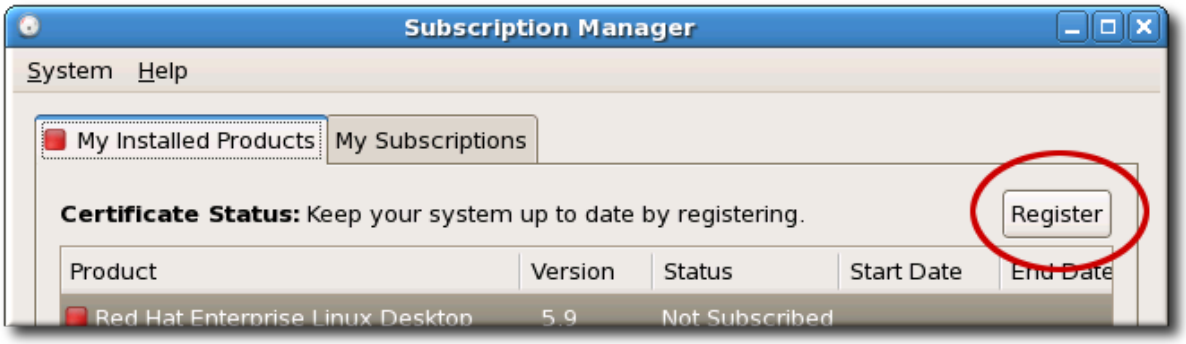
システムは、初回起動のプロセス中またはキックスタート設定の一部として (いずれも 『インストールガイド』 を参照)、サブスクリプションサービスに登録することができます。システムは、設定完了後に登録することも可能です。また、登録済みのエンタイトルメントサービス内での管理対象外となる場合には、サブスクリプションサービスインベントリから削除 (登録解除) することもできます。

5.2.1. GUI からの登録

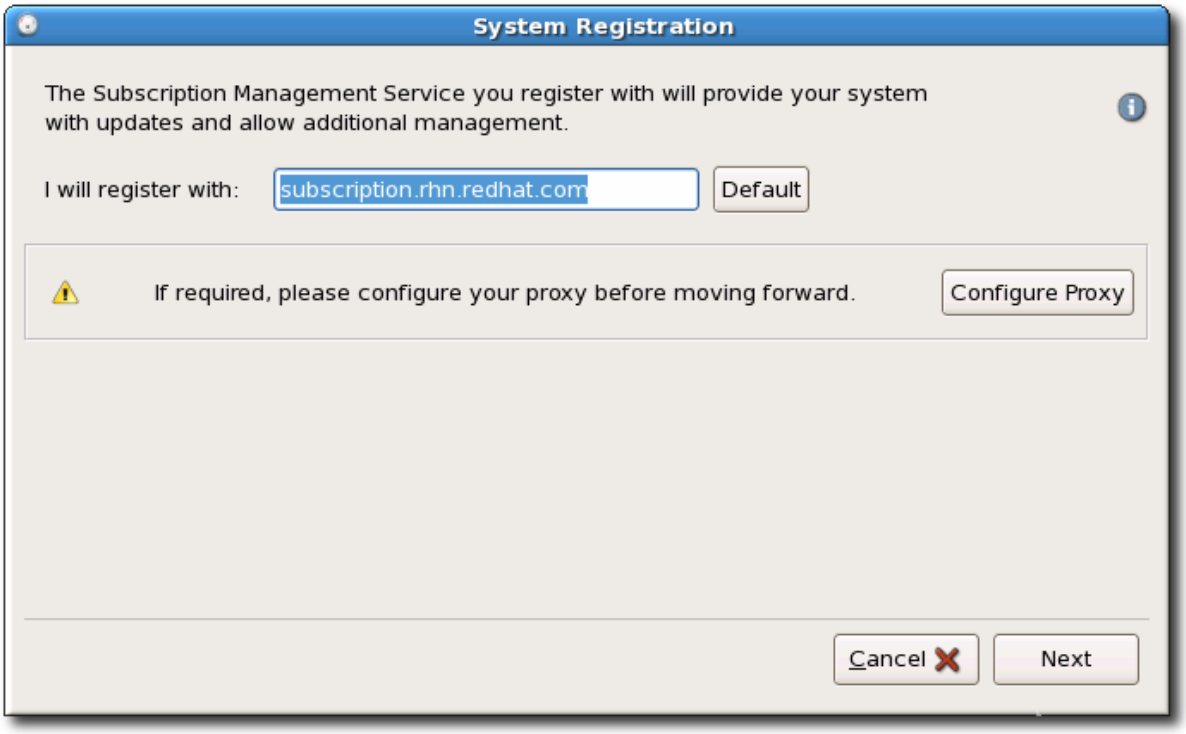
1. サブスクリプションマネージャーを起動します。

```
[root@server ~]# subscription-manager-gui
```

2. システムが未登録の場合は、**インストール済みの製品** タブの右上隅に **登録** ボタンが表示されます。



3. 登録に使用するサブスクリプションサーバーを特定するには、サービスのホスト名を入力します。 デフォルトのサービスはカスタマーポータルによるサブスクリプション管理です。 ホスト名は **`subscription.rhn.redhat.com`** になります。 Subscription Asset Manager など、 別のサブスクリプションサービスを使用する場合は、ローカルサーバーのホスト名を入力します。



証明書ベースのサブスクリプションを認識して使用する各種のサブスクリプションサービスがあります。初回起動時にこれらのサービスにシステムを登録できます。

- カスタマーポータルによるサブスクリプション管理、Red Hat でホストしているサービス (デフォルト) です。
- Subscription Asset Manager、オンプレミスのサブスクリプションサーバーで、 コンテンツ配信はプロキシでカスタマーポータルのサービスに戻します。
- CloudForms System Engine、オンプレミスのサービスでサブスクリプションサービスとコンテンツ配信の両方を処理します。

4. **ログインするサブスクリプションサービス** のユーザー認証情報を入力します。

使用するユーザー認証情報はサブスクリプションサービスによって異なります。カスタマーポータルに登録する場合は、管理者または企業アカウントの Red Hat Network 認証情報を使用します。

ただし、 Subscription Asset Manager や CloudForms System engine の場合には、使用するユーザーアカウントはオンプレミスのサービス内で作成され、カスタマポータルのユーザーアカウントとは異なります。

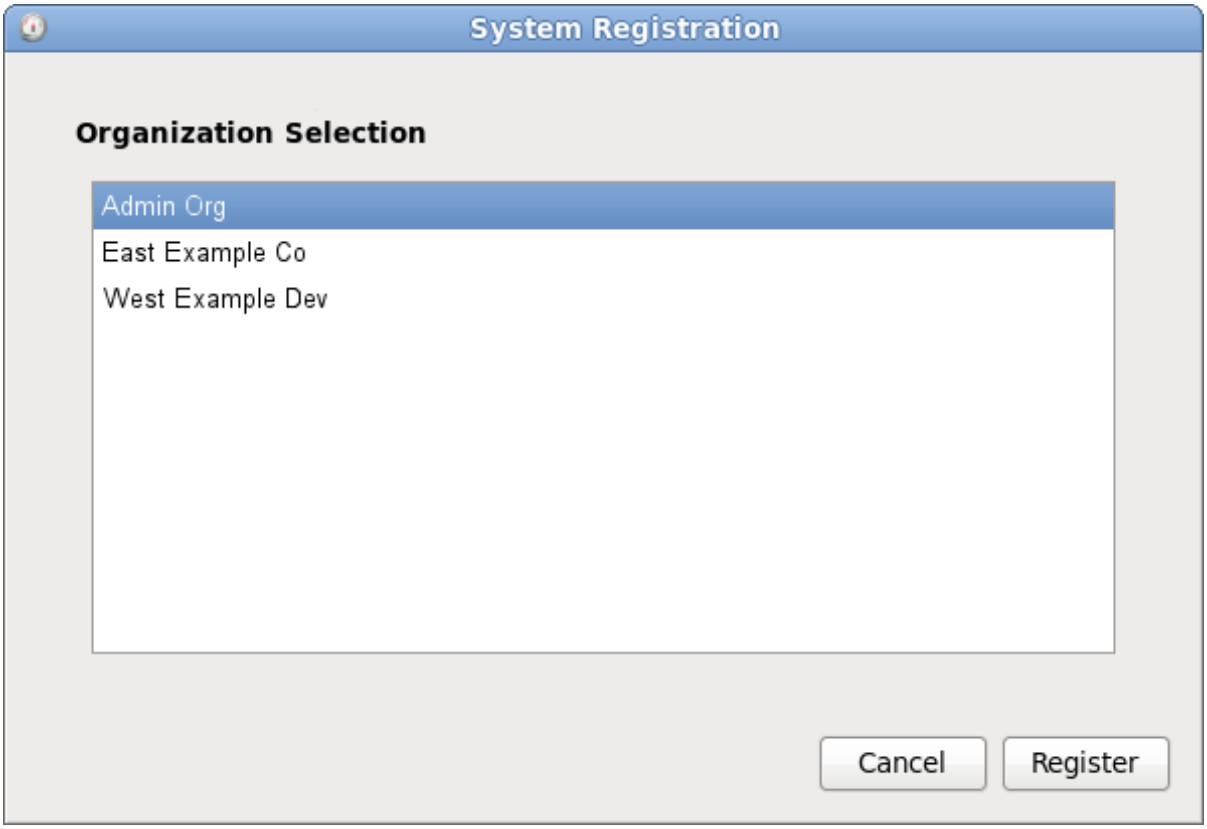
5. オプションで、**登録の後に手動でサブスクリプションを割当て** チェックボックスを選択することもできます。

デフォルトでは、登録プロセスでシステムに最適なサブスクリプションが自動的にアタッチされます。「サブスクリプションのアタッチと削除」 にあるように、この機能をオフにしてサブスクリプションを手動で選択できるようにすることも可能です。

6. 登録が始まると、サブスクリプションマネージャーはシステムを登録する組織および環境 (組織内のサブドメイン) をスキャンします。

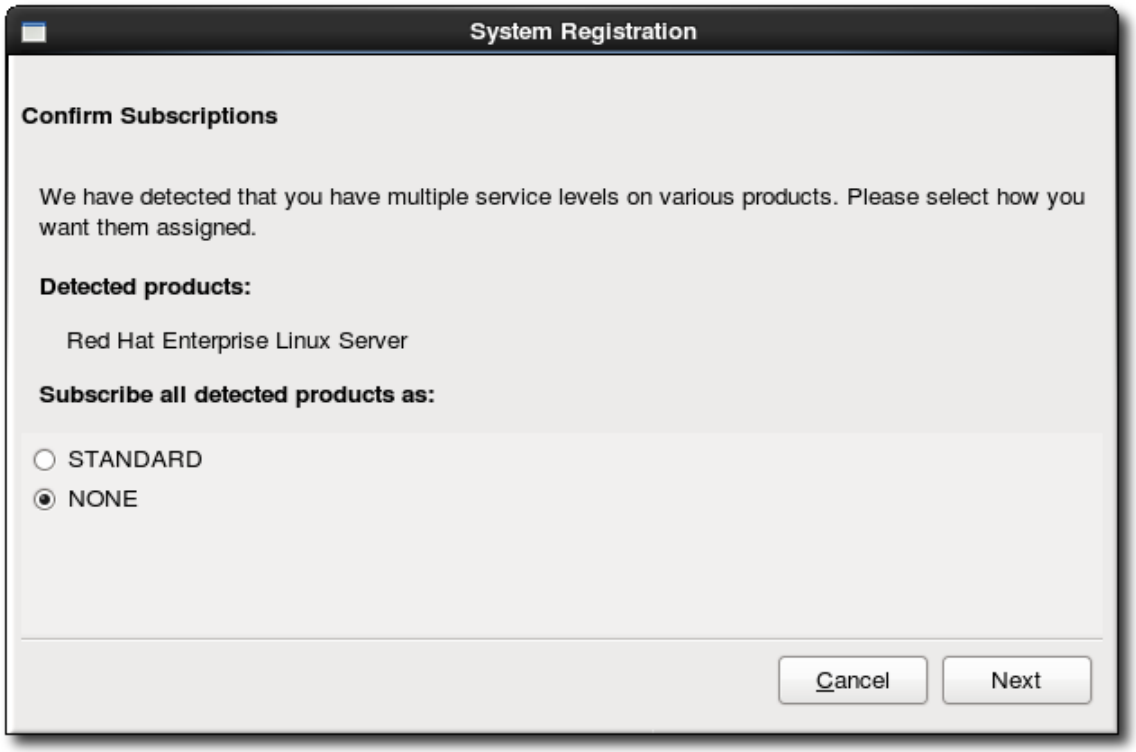
カスタマーポータルのサブスクリプション管理を使用する IT 環境には組織が 1 つしかないため、 これ以上の設定は必要ありません。Subscription Asset Manager などのローカルのサブスクリプションサービスを使用する IT インフラストラクチャーの場合、 複数の組織が設定されていることがあります。 また、 それらの組織内にさらに複数の環境が設定されている場合もあります。

複数の組織が検出された場合は、サブスクリプションマネージャーは参加する組織を 1 つ選択するように促します。

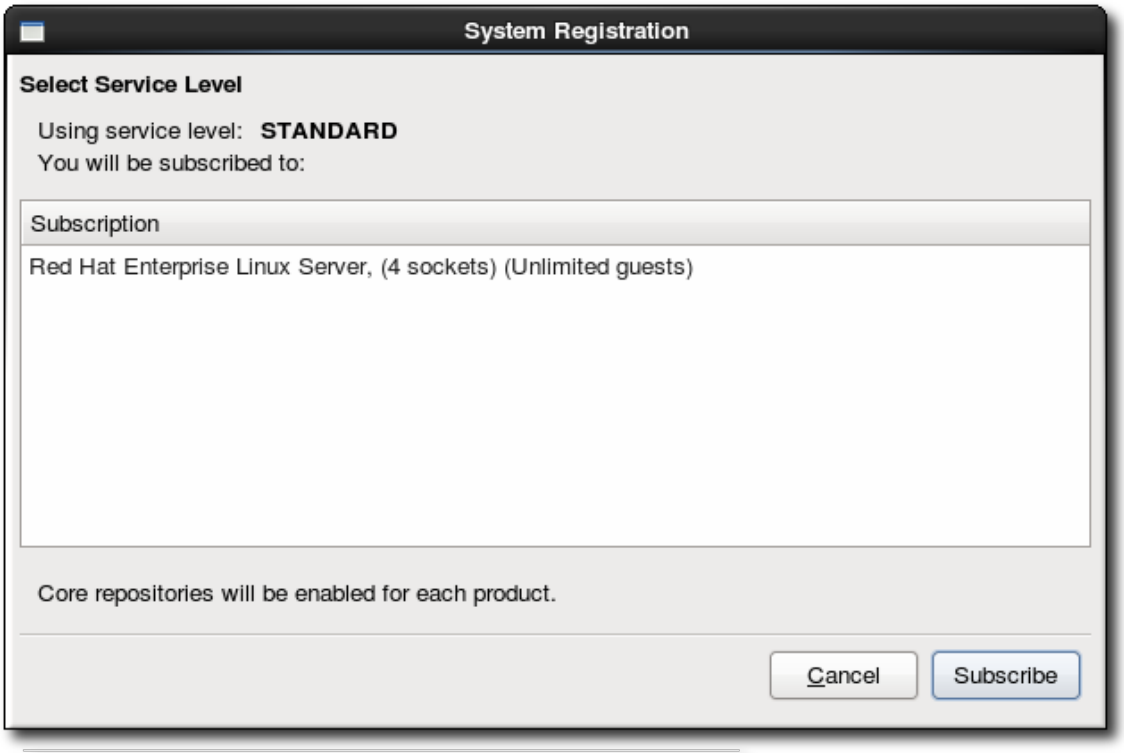


7. デフォルト設定では、サブスクリプションは自動的に選択され、システムにアタッチされます。システムにアタッチするサブスクリプションを確認して、確定します。

    1. プロンプトが表示された場合、検出されたサブスクリプションに使用するサービスレベルを選択します。



    2. サブスクリプションマネージャーは、選択されたサブスクリプションを表示します。ウィザードの **サブスクライブ** ボタンをクリックして、このサブスクリプションを確定します。



5.2.2. コマンドラインからの登録

マシンを登録する最も簡単な方法は、**register** コマンドで、認証に必要なユーザーアカウント情報をカスタマイズポータルのサブスクリプション管理にわたすことです。システムが正しく認証されると、新しく割り当てられたシステムインベントリ ID とそれを登録したユーザーのアカウント名がエコーバックされます。

register のオプション一覧は、表5.2「register オプション」 に記載されています。

例5.1 カスタマーポータルへのシステム登録

```
[root@server1 ~]# subscription-manager register --username admin-example
--password secret

The system has been registered with id: 7d133d55-876f-4f47-83eb-
0ee931cb0a97
```

例5.2 登録中の自動サブスクライブ

register コマンドには **--autosubscribe** というオプションがあります。これを使用すると、ワンステップでシステムをサブスクリプションサービスに登録し、そのシステムのアーキテクチャーに最適なサブスクリプションを即時にアタッチすることができます。

```
[root@server1 ~]# subscription-manager register --username admin-example
--password secret --autosubscribe
```

これは、サブスクリプションマネージャー UI のデフォルト設定で登録する場合と同じ動作です。

例5.3 Subscription Asset Manager を使用したシステム登録

Subscription Asset Manager または CloudForms System Engine では、1 つのアカウントに **組織** と呼ばれる複数の独立した下位区分があります。システムの参加先となる組織 (基本的には、主要アカウント内の独立したグループまたはユニット) を指定する必要があります。これには、ユーザー名とパスワードの他に **--org** オプションを使用します。ここで指定するユーザーには、その組織にシステムを追加するためのアクセスパーミッションも必要となります。

カスタマーポータルのサブスクリプション管理以外のサブスクリプションサービスに登録するには、追加のオプションをいくつか使用して、システム登録先の環境および組織区分を特定する必要があります。

- **サブスクリプションサービス内のユーザーアカウント** のユーザー名およびパスワード
- **--serverurl** でサブスクリプションサービスのホスト名を指定
- **--baseurl** でコンテンツ配信サービスのホスト名を指定 (CloudForms System Engine のみ)
- **--org** でシステム登録する組織名を指定
- **--environment** でシステムを追加する組織内の環境 (グループ) 名を指定。デフォルトの環境はすべての組織に設定されているため、これはオプションです。

  システムを環境に追加できるのは、登録中のみです。

```
[root@server1 ~]# subscription-manager register --username=admin-example
--password=secret --org="IT Department" --environment="dev" --
serverurl=sam-server.example.com

The system has been registered with id: 7d133d55-876f-4f47-83eb-
0ee931cb0a97
```

注記

システムが複数組織環境内にある場合、組織が指定されていない状態で **register** コマンドを実行すると Remote Server エラーが返されます。

表5.2 register オプション

オプション	詳細	必須
--username=**name**	コンテンツサーバーのユーザーアカウント名を指定します。	必須
--password=**password**	ユーザーアカウントのパスワードを指定します。	必須

オプション	詳細	必須
--serverurl=**hostname**	使用するサブスクリプションサービスのホスト名を指定します。デフォルトは、カスタマーポータルのサブスクリプション管理、subscription.rhn.redhat.com です。このオプションを使用しない場合、システムはカスタマーポータルのサブスクリプション管理に登録されます。	Subscription Asset Manager または CloudForms System Engine で必須
--baseurl=**URL**	更新を受信するためのコンテンツ配信サーバーのホスト名を指定します。カスタマーポータルのサブスクリプション管理と Subscription Asset Manager のいずれも Red Hat がホストするコンテンツ配信サービスを使用します。URL は https://cdn.redhat.com です。CloudForms System Engine は独自のコンテンツをホストしているため、System Engine に登録するシステムはこの URL を使用する必要があります。	CloudForms System Engine で必須
--org=**name**	システムの参加先となる組織を指定します。	ホスト型環境以外では必須
--environment=**name**	1 つの組織内の 1 つの環境にシステムを登録します。	オプション
--name=**machine_name**	登録するシステム名を設定します。デフォルトではホスト名と同じです。	オプション
--autosubscribe	互換性がある最適なサブスクリプションを自動的にアタッチします。ワンステップでシステムを設定できるため、自動設定の操作に適しています。	オプション
--activationkey=**key**	登録プロセスの一環として、既存のサブスクリプションをアタッチします。サブスクリプションは、ベンダーまたはシステム管理者が Subscription Asset Manager を使用して事前に割り当てています。	オプション

オプション	詳細	必須
--servicelevel=None|Standard|Premium	マシン上でサブスクリプションに使用するサービスレベルを設定します。これは **--autosubscribe** オプションのみで使用されます。	オプション
--release=NUMBER	システムのサブスクリプションで使用するオペレーティングシステムのマイナーリリースを設定します。製品と更新は、この特定のマイナーリリースバージョンに限定されます。**--autosubscribe** オプションのみで使用されます。	オプション
--force	システムが既に登録済みの場合でも登録します。通常、マシンが登録済みの場合には、登録の操作は失敗します。	オプション

5.2.3. 登録解除

マシンの登録解除に必要な操作は、**unregister** コマンドの実行のみです。これにより、サブスクリプションサービスから対象となるシステムのエントリーが削除され、サブスクリプションもすべて削除されます。また、そのシステムの ID とサブスクリプション証明書もローカルで削除されます。

コマンドラインでのシステム登録解除には、**unregister** コマンドのみが必要となります。

例5.4 システムの登録解除

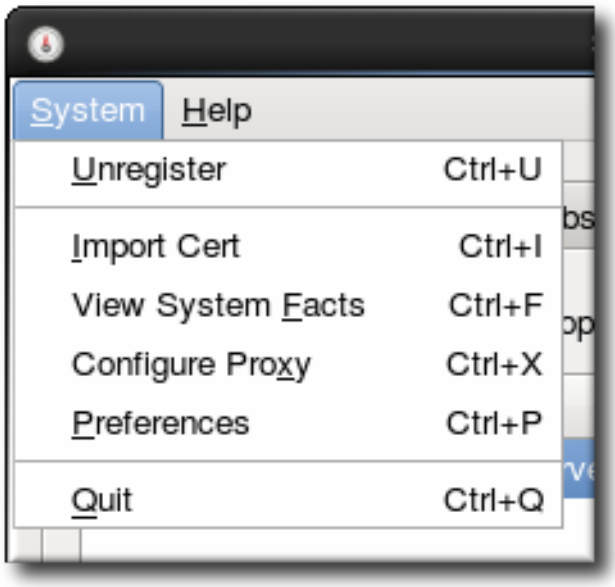
```
[root@server1 ~]# subscription-manager unregister
```

サブスクリプションマネージャー GUI からの登録解除方法

1. サブスクリプションマネージャー UI を開きます。

   ```
   [root@server ~]# subscription-manager-gui
   ```

2. **システム** メニューを開いて、**登録解除** を選択します。



3. システムの登録解除を確認します。

5.3. サブスクリプションのアタッチと削除

システムに **サブスクリプション** を割り当てることで、そのシステムはサブスクリプション内の Red Hat 製品をインストール、更新することができるようになります。サブスクリプションとは、1 回に購入された全製品の全バリエーションの一覧であり、製品とサブスクリプションが利用できる回数を定義します。この数量は、概して利用可能なユーザーライセンス数です。これらのライセンスの 1 つがシステムに割り当てられると、そのサブスクリプションはそのシステムに **アタッチ** されます。

5.3.1. GUI を使用したサブスクリプションのアタッチと削除

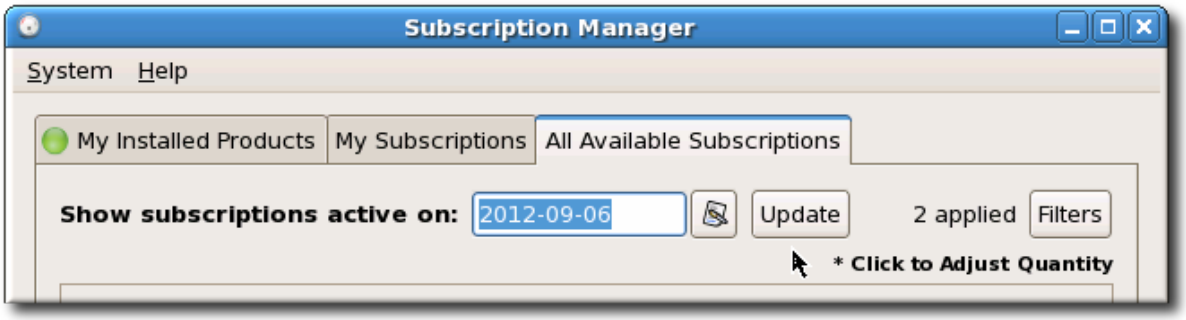
5.3.1.1. サブスクリプションのアタッチ

1. サブスクリプションマネージャーを起動します。

```
[root@server ~]# subscription-manager-gui
```

2. **すべての利用可能なサブスクリプション** タブを開きます。

3. オプションとして、日付の範囲を設定してから **フィルター** ボタンをクリックし、利用可能なサブスクリプションを検索します。

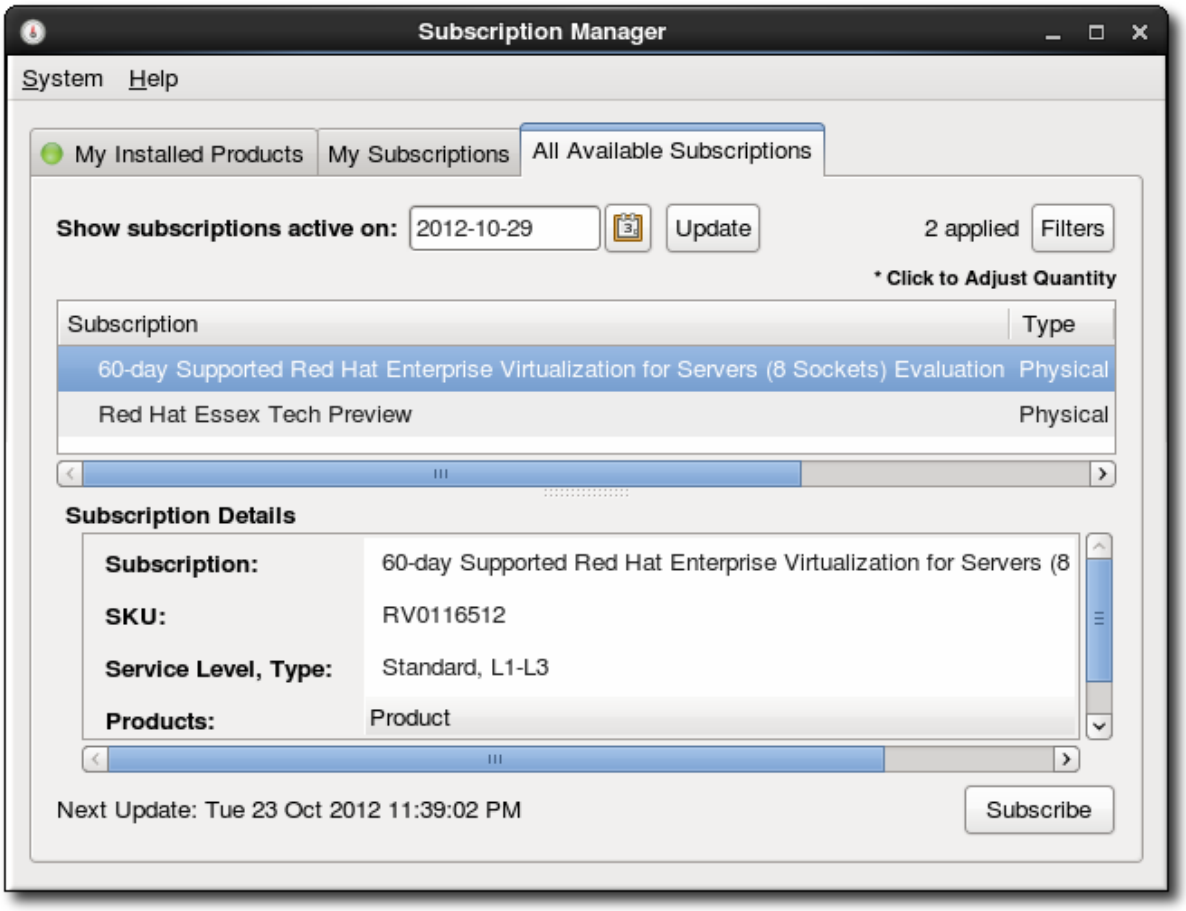


サブスクリプションは、アクティブな日付や名前でフィルタリングできます。また、チェックボックスを使用すると、より詳細なフィルタリングが可能です。

- **match my system (自分のシステムにマッチ)** を選択すると、システムアーキテクチャーに適合するサブスクリプションのみが表示されます。
- **match my installed products (自分のインストール済み製品にマッチ)** を選択すると、システム上に現在インストール済みの製品と機能するサブスクリプションが表示されます。
- **既存のサブスクリプションとの重複をしない** を選択すると、製品が重複しているサブスクリプションが除外されます。サブスクリプションがすでに特定の製品のシステムにアタッチされている場合や、複数のサブスクリプションが同じ製品を提供している場合は、サブスクリプションサービスがそれらのサブスクリプションをフィルタリングして、最適なサブスクリプションのみが表示されます。
- **次のテキストを含む** は、サブスクリプションやプール内で製品名などの文字列を検索します。

日付とフィルターの設定後、**更新** ボタンをクリックして適用します。

4. 利用可能なサブスクリプションを 1 つ選択します。



5. **Subscribe (サブスクライブ)** ボタンをクリックします。

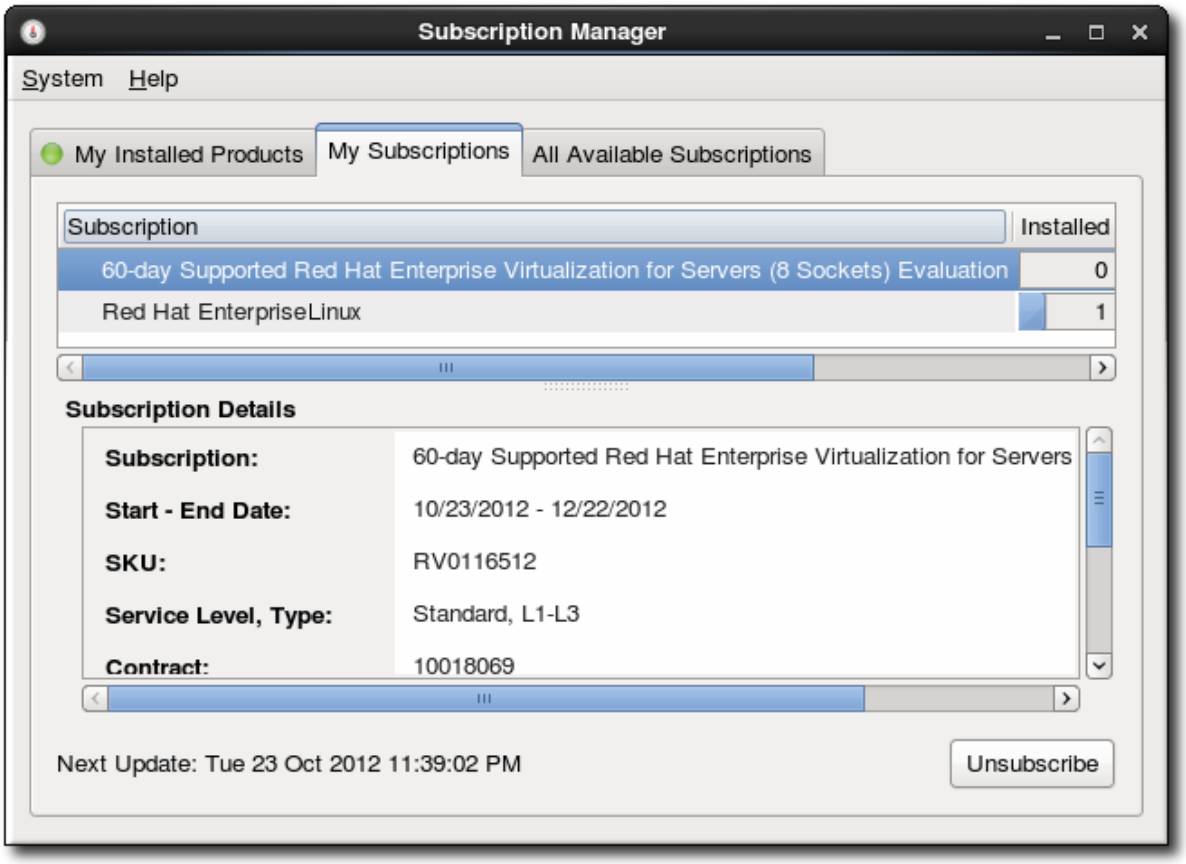
5.3.1.2. サブスクリプションの削除

1. サブスクリプションマネージャーを起動します。

```
[root@server ~]# subscription-manager-gui
```

2. **自分のサブスクリプション** タブを開きます。

   システムに現在アタッチされているすべてのアクティブなサブスクリプションが一覧表示されます (そのサブスクリプションで利用できる製品は、インストール済みの場合と未インストールの場合があります)。



3. 削除するサブスクリプションを選択します。

4. ウィンドウ右下にある **サブスクリプション解除** ボタンをクリックします。

5.3.2. コマンドラインを使用したサブスクリプションのアタッチと削除

5.3.2.1. サブスクリプションのアタッチ

サブスクリプションをシステムにアタッチするには、**--pool** オプションを使用して、個別の製品またはサブスクリプションを指定する必要があります。

```
[root@server1 ~]# subscription-manager subscribe --pool=XYZ01234567
```

subscribe コマンドのオプション一覧は、表5.3「subscribe オプション」 に記載されています。

購入した製品のサブスクリプションプール ID を指定する必要があります。このプール ID は製品サブスクリプション情報に表示され、**list** コマンドを実行すると確認できます。

```
[root@server1 ~]# subscription-manager list --available

+-------------------------------------------+
    Available Subscriptions
+-------------------------------------------+
ProductName:            RHEL for Physical Servers
ProductId:              MKT-rhel-server
PoolId:                 ff8080812bc382e3012bc3845ca000cb
Quantity:               10
Expires:                2011-09-20
```

別の方法では、**--auto** オプション (**register** コマンドの **--autosubscribe** オプションに類似) を使用して、サブスクリプションサービスが特定した最適なサブスクリプションをシステムにアタッチすることもできます。

```
[root@server1 ~]# subscription-manager subscribe --auto
```

表5.3 subscribe オプション

オプション	詳細	必須
--pool=**pool-id**	システムにアタッチするサブスクリプション ID を指定します。	**--auto** の使用時以外は必須
--auto	システムを 1 つ以上の最適なサブスクリプションに自動的にアタッチします。	オプション
--quantity=**number**	複数カウントのサブスクリプションをシステムにアタッチします。これは、カウント制限を定義するサブスクリプションを対象に使用されます (例: 2 ソケットサーバー用のサブスクリプションを 2 つ使用して、1 台の 4 ソケットマシンに適用)。	オプション
--servicelevel=None|Standard|Premium	マシン上でサブスクリプションに使用するサービスレベルを設定します。これが使われるのは、**--auto** オプションと一緒の時のみです。	オプション

5.3.2.2. コマンドラインを使用したサブスクリプションの削除

一つのシステムを複数のサブスクリプションおよび製品にアタッチすることができます。同様に、一つまたはすべてのサブスクリプションをシステムから削除することもできます。

unsubscribe コマンドを **--all** オプションで実行すると、現在システムにアタッチされているすべての製品サブスクリプションおよびサブスクリプションプールが削除されます。

```
[root@server1 ~]# subscription-manager unsubscribe --all
```

一つの製品サブスクリプションを削除することもできます。各製品には、識別のための X.509 証明書がインストールされています。削除する製品サブスクリプションは、**unsubscribe** コマンドの X.509 証明書の ID 番号を参照して特定します。

1. 製品サブスクリプションを削除する場合は、製品証明書のシリアル番号を取得してください。シリアル番号は **subscription#.pem** ファイル (例: **3927295555856979 07.pem**) から、または以下のように **list** コマンドを使って取得できます。

   ```
   [root@server1 ~]# subscription-manager list --consumed
   ```

```
+-------------------------------------------+
    Consumed Product Subscriptions
+-------------------------------------------+


ProductName:          High availability (cluster suite)
ContractNumber:       0
SerialNumber:         11287514358600162
Active:               True
Begins:               2010-09-18
Expires:              2011-11-18
```

2. subscription-manager ツールを **--serial** オプションで実行し、証明書を特定します。

```
[root@server1 ~]# subscription-manager unsubscribe --
serial=11287514358600162
```

5.4. ベンダーサブスクリプションの有効化

システムで利用可能な既存のサブスクリプションを使用して、そのシステムを設定することができます。サードパーティベンダーから購入した一部のシステムには、Red Hat 製品のサブスクリプションがそのマシン購入に含まれています。

Red Hat サブスクリプションマネージャーは、システムハードウェアと BIOS に関する情報をシステム情報にプルし、ハードウェアベンダーを認識します。ベンダーと BIOS 情報が特定の設定に一致する場合は、サブスクリプションを **有効** にできます。これにより、サブスクリプションをシステムに自動的にアタッチすることができます。

5.4.1. GUI を使用したサブスクリプションの有効化

注記

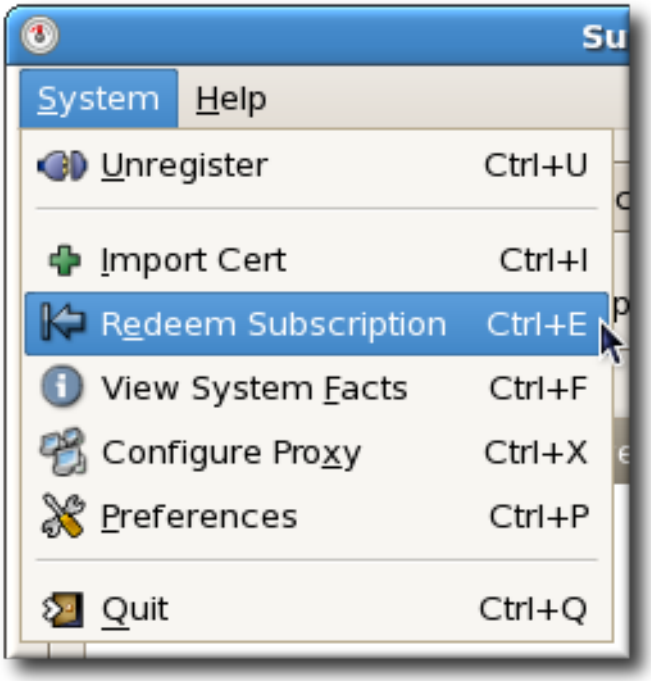
有効にするサブスクリプションがマシンにない場合、 **Redeem (有効化)** メニュー項目は表示されません。

1. サブスクリプションマネージャーを起動します。

```
[root@server ~]# subscription-manager-gui
```

2. 必要に応じて 「GUI からの登録」 にあるとおり、システムを登録します。

3. ウィンドウの左上隅にある **システム** メニューを開き、**Redeem (有効化)** 項目をクリックします。



4. 有効化が完了した時点に送信する通知の宛先となる電子メールアドレスを、ダイアログウィンドウ内に入力します。有効化のプロセスは、ベンダーに連絡して事前定義されたサブスクリプションの情報を受信するのに数分かかる場合があるため、通知メッセージはサブスクリプションマネージャーのダイアログウィンドウではなく、電子メールで送られます。

5. **Redeem（有効化）** ボタンをクリックします。

確認の電子メールが届くまで、最大 10 分程度かかる場合があります。

5.4.2. コマンドラインを使用したサブスクリプションの有効化

注記

サブスクリプションサービスがシステムとそのサブスクリプションを適切に特定するには、マシンを **最初に** 登録する必要があります。

マシンのサブスクリプションを有効化をするには、**redeem** コマンドで、プロセスが完了した時点で送信される有効化の通知メールの宛先となる電子メールアドレスを指定して実行します。

```
# subscription-manager redeem --email=jsmith@example.com
```

5.5. SUBSCRIPTION ASSET MANAGER のアクティベーションキーを使用したサブスクリプションのアタッチ方法

ローカルの Subscription Asset Manager は、システムが使用するサブスクリプションを事前設定できます。この事前定義されたサブスクリプションのセットは、アクティベーションキーで識別されます。このアクティベーションキーは、ローカルシステムでサブスクリプションをアタッチする際に使用します。

Subscription Asset Manager のアクティベーションキーは、新規システム登録プロセスの一環で使用できます。

```
# subscription-manager register --username=jsmith --password=secret --
org="IT Dept" --activationkey=abcd1234
```

組織が複数ある場合はシステムの組織を指定する必要があり、この情報はアクティベーションキーでは定義されていません。

5.6. システムの詳細設定

サブスクリプションの自動アタッチと自動修復 (更新) は、システムにアタッチするサブスクリプションを選択します。これは、現在インストール済みの製品、ハードウェア、アーキテクチャーなどの各種基準に基づいて行われます。サブスクリプションマネージャーが使用する詳細設定で以下の 2 つをさらに設定することが可能です。

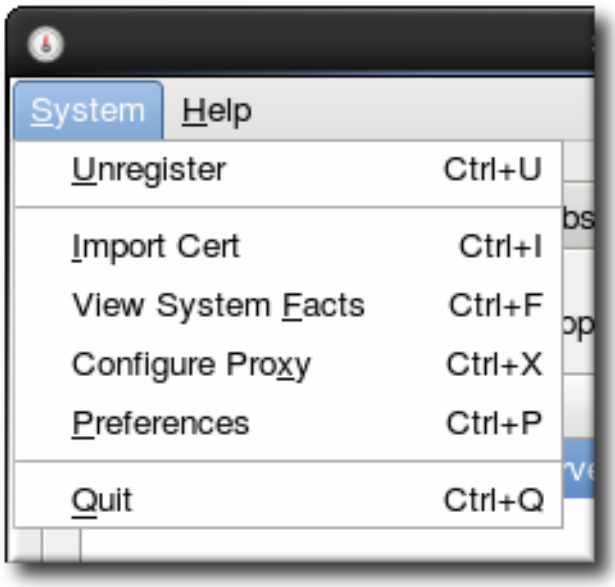
- サブスクリプションのサービスレベル
- 使用するオペレーティングシステムのマイナーバージョン (X.Y)

これは自動修復で特に有用です。自動修復は、すべてのインストール済み製品と現在のサブスクリプションがアクティブとなっているように毎日実行されます。

5.6.1. UI での詳細設定

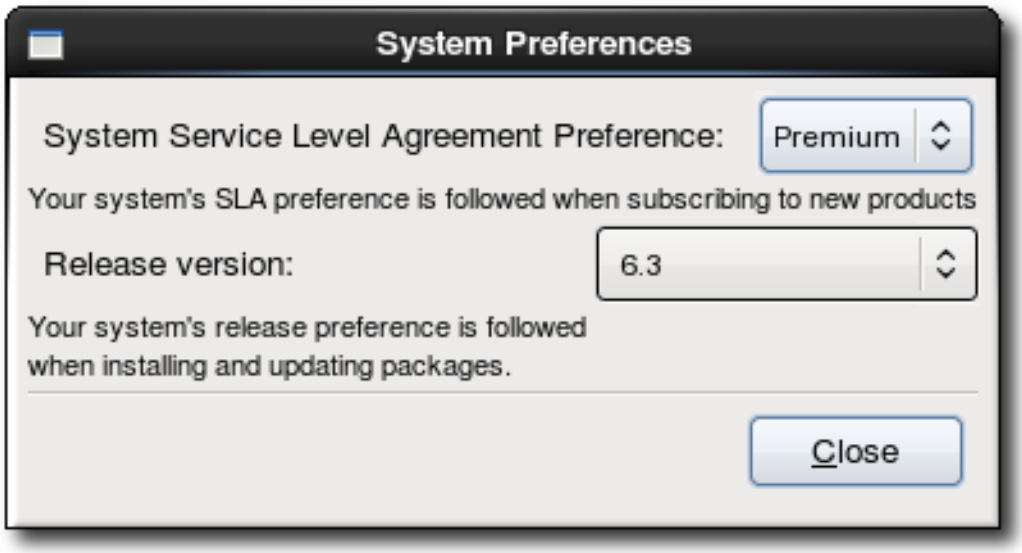
サービスレベルおよびオペレーティングシステムのリリースバージョンの詳細設定はどちらも、サブスクリプションマネージャーの **システムの設定** のダイアログボックスで行います。

1. サブスクリプションマネージャーを開きます。
2. **システム** メニューを開きます。
3. **システムの設定** メニュー項目を選択します。



4. ドロップダウンメニューから希望するサービスレベル同意書の設定を選択します。アクティブなサブスクリプションすべてに基づいて、Red Hat アカウントで利用可能なサービスレベルのみが表示されます。

5. **リリースバージョン** ドロップダウンメニューでオペレーティングシステムのリリース設定を選択します。表示されるバージョンは、アカウントにアクティブなサブスクリプションがある Red Hat Enterprise Linux バージョンのみです。



6. 設定が保存され、今後のサブスクリプションの操作に適用されます。**閉じる** をクリックしてダイアログを閉じます。

5.6.2. コマンドラインを使用したサービスレベルの設定

一般的なサービスレベルは、**`service-level --set`** コマンドを使用して設定できます。

> **例5.5 サービスレベル詳細の設定**
>
> まず、**`service-level`** コマンドを **`--list`** オプションで実行して、システムで利用可能なサービスレベルを表示します。

```
[root@server ~]# subscription-manager service-level --list
+-----------------------------------------+
          Available Service Levels
+-----------------------------------------+
Standard
None
Premium
Self-Support
```

次に、システムで希望するレベルを設定します。

```
[root@server ~]# subscription-manager service-level --set=self-support
Service level set to: self-support
```

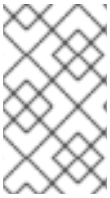
ローカルシステムの現在の設定は、**--show** オプションで表示できます。

```
[root#server ~]# subscription-manager service-level --show
Current service level: self-support
```

サービスレベルの設定は、サブスクリプション動作の実行中に定義できます (システム登録や登録後のサブスクリプションのアタッチなど)。これにより、システム設定の上書きが可能です。**register** と **subscribe** のコマンドでは、**--servicelevel** オプションで動作の詳細を設定できます。

例5.6 プレミアムサービスレベルでのサブスクリプションの自動アタッチ

```
[root#server ~]# subscription-manager subscribe --auto --servicelevel
Premium
Service level set to: Premium
Installed Product Current Status:
ProductName:            RHEL 6 for Workstations
Status:                 Subscribed
```



注記

--servicelevel オプションには、**--autosubscribe** オプション (登録用) または **--auto** オプション (サブスクライブ用) が必要です。特定のプールのアタッチ時やサブスクリプションのインポート時には使用できません。

5.6.3. コマンドラインで希望するオペレーティングシステムのリリースバージョンを設定する

IT 環境の多くは、一定のセキュリティーレベルか他の基準に合致していると認定されている必要があります。その場合、主要なアップグレードの計画と管理は注意して行う必要があります — 管理者は単に **yum update** を実行して、あるバージョンから次のバージョンへと移動することはできません。

リリースバージョンの詳細を設定すると、システムのアクセスは、自動的に最新バージョンのリポジトリを使用するのではなく、オペレーティングシステムのバージョンに関連したコンテンツリポジトリに制限されます。

例えば、希望するオペレーティングシステムバージョンが 6.3 の場合、他のリポジトリが利用可能でも、システムのすべてのインストール済み製品およびアタッチされたサブスクリプションには 6.3 のコンテンツリポジトリが選ばれます。

例5.7 登録中にオペレーティングシステムのリリースを設定する

リリースバージョンの設定は、システムの登録時に **register** コマンドを **--release** オプションで実行することで設定できます。これによりこのリリース設定は、システム登録時に選択され自動的にアタッチされたサブスクリプションに適用されます。

--autosubscribe オプションは、自動的にアタッチするサブスクリプションの選択に使用される基準の 1 つであるため、詳細設定時に必要となります。

```
[root#server ~]# subscription-manager register --autosubscribe --
release=6.4 --username=admin@example.com...
```

注記

リリース設定はサービスレベルの設定と違い、登録時にのみ使用するか、詳細設定として設定できます。**subscribe** コマンドでは指定できません。

例5.8 オペレーティングシステムのリリース詳細設定

release コマンドは、アタッチされたサブスクリプションだけでなく、組織で利用可能な購入済みサブスクリプションに基づいて、利用可能なオペレーティングシステムのリリースを表示します。

```
[root#server ~]# subscription-manager release --list
+-------------------------------------------+
          Available Releases
+-------------------------------------------+
6.2
6.3
```

そして、**--set** で詳細設定を利用可能なリリースバージョンの 1 つに設定します。

```
[root#server ~]# subscription-manager release --set=6.3
Release version set to: 6.3
```

5.7. サブスクリプションの有効期限と通知の管理

サブスクリプションは、**有効期間** と呼ばれる一定期間中にアクティブとなります。サブスクリプションを購入すると、そのコントラクトの開始日と終了日が設定されます。

システムには、複数のサブスクリプションをアタッチすることが可能です。製品にはそれ自体のサブスクリプションが必要です。さらに、一部の製品では完全なサブスクリプションとするために、複数のサブスクリプションが必要になる場合があります。例えば、16 ソケットのマシンでは、ソケット数に対応するため 4 つの 4 ソケットのオペレーティングシステムのサブスクリプションが必要になります。

自分のインストール済みソフトウェア タブには、システム全体のサブスクリプションステータスが表示されます。また、有効な製品サブスクリプションが無効になる (有効期限が切れる) 最初の日付も表示されます。

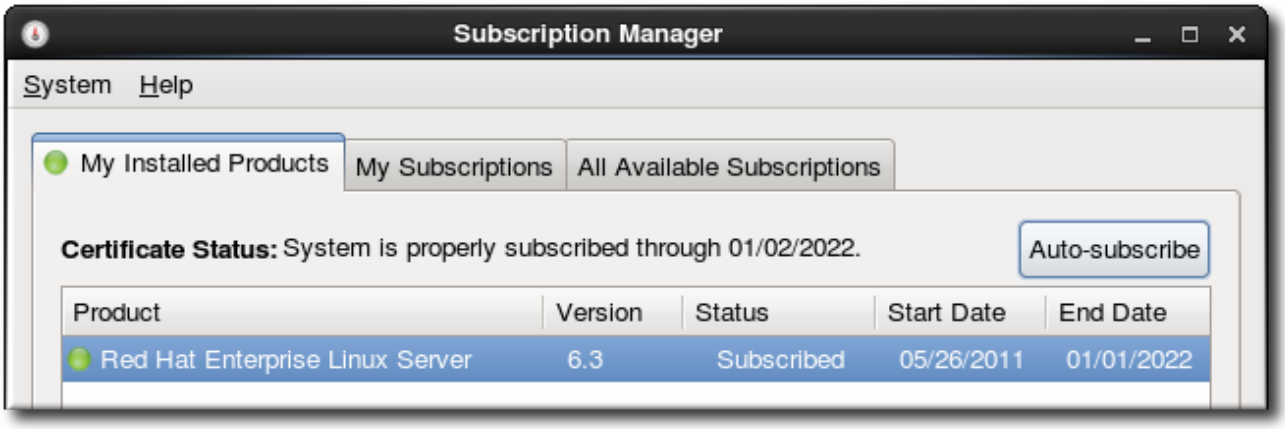


図5.2 有効期限

Red Hat サブスクリプションマネージャーは、システムにインストールされた製品の有効な証明書に対するあらゆる変更を示すログと UI メッセージを提供します。サブスクリプションマネージャー GUI では、システムサブスクリプションのステータスは色分けされています。全製品が完全にサブスクライブされている場合は **緑色**、 一部の製品がサブスクライブされていない可能性はあるが更新がまだ有効な場合は **黄色**、更新が無効な場合は **赤色** で表示されます。

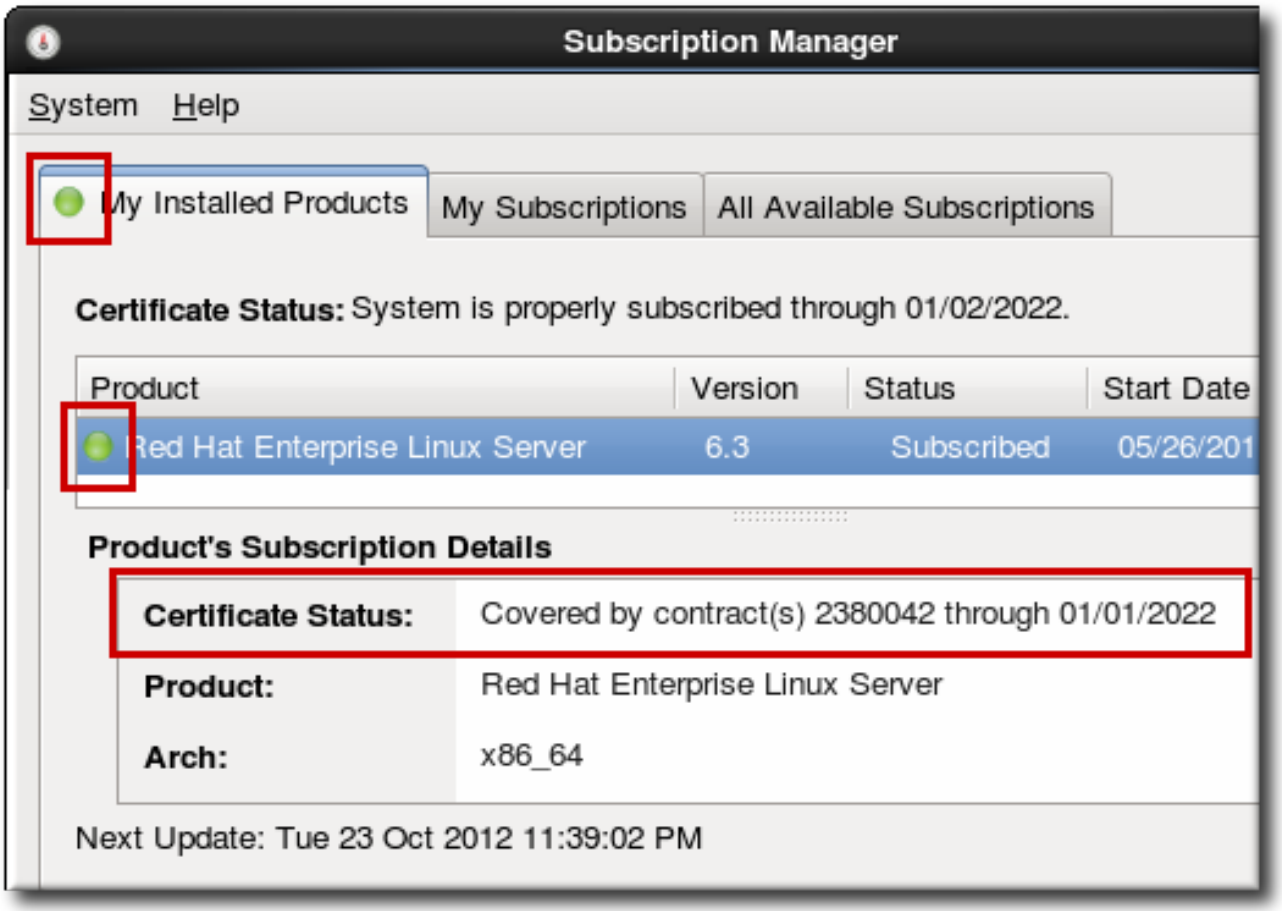


図5.3 色分けされたステータス表示

マシンのステータスは、コマンドラインツールでも表示されます。緑、黄、赤の色分けは、テキストのステータスメッセージではそれぞれ、**subscribed**、**partially subscribed**、**expired/not subscribed** と表示されます。

```
[root@server ~]# subscription-manager list
+-------------------------------------------+
    Installed Product Status
+-------------------------------------------+

ProductName:            Red Hat Enterprise Linux Server
Status: Not Subscribed
Expires:
SerialNumber:
ContractNumber:
AccountNumber:
```

サブスクリプションの変更に関する警告がある時には、上部のメニューバーに燃料メーター状の小さなアイコンが表示されます。

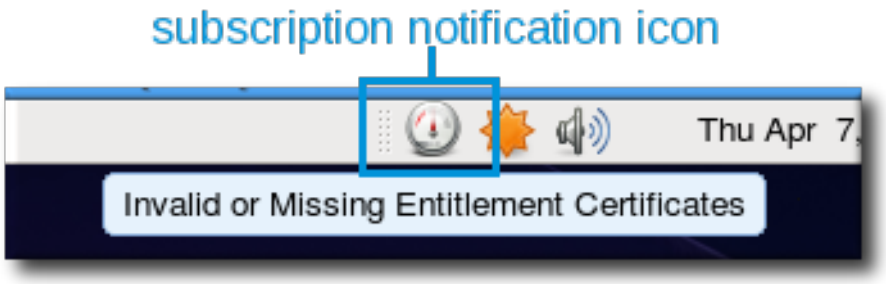


図5.4 サブスクリプション通知のアイコン

インストール済み製品のサブスクリプションの有効期限が近づくと、サブスクリプションマネージャーデーモンは警告を発します。このメッセージは、システムに有効な証明書のない製品がある場合に表示されるメッセージに似ています。この警告は、製品に対応するサブスクリプションがアタッチされていないか、サブスクリプションの有効期限が切れた後も製品がインストールされていることを意味します。サブスクリプション通知のウィンドウにある **自分のサブスクリプションを管理** のボタンをクリックすると、Red Hat サブスクリプションマネージャー GUI が開き、サブスクリプションの確認と更新ができます。

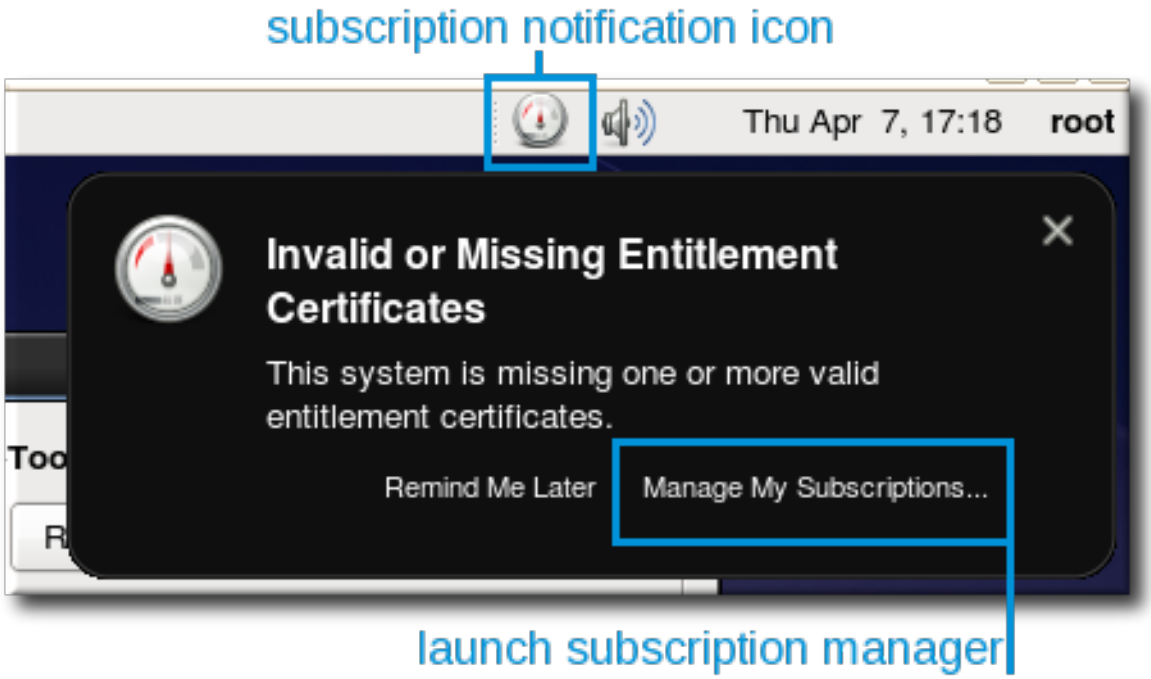


図5.5 サブスクリプション警告のメッセージ

サブスクリプションマネージャー UI を開くと (通知エリアから開いたか、通常の方法で開いたかに関わらず)、製品に有効な証明書が欠けているかどうかを示すアイコンが左上隅に表示されます。無効になった製品に適合するサブスクリプションをアタッチする最も簡単な方法は、**自動サブスクライブ** ボタンをクリックすることです。

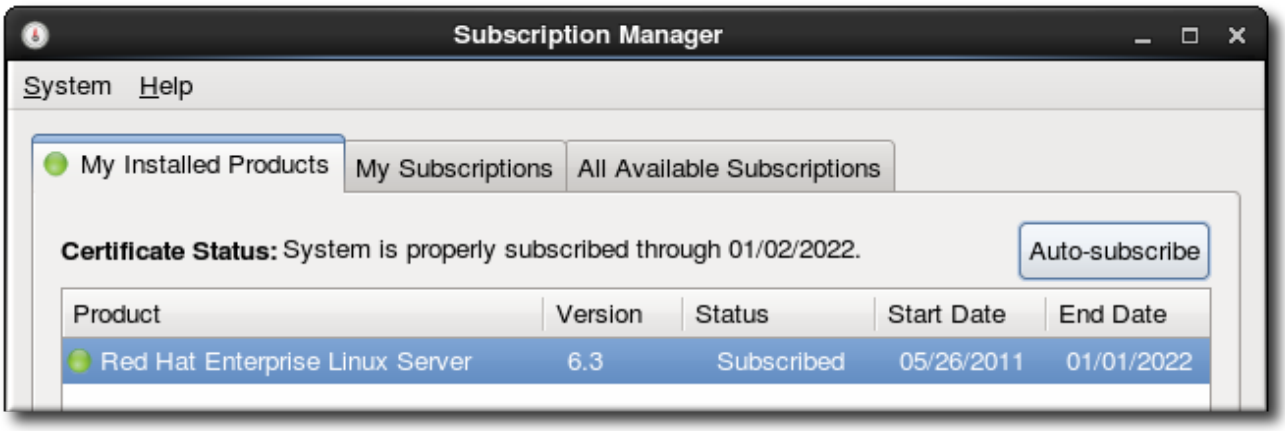


図5.6 自動サブスクライブボタン

システムのサブスクライブ のダイアログは、有効な証明書がない特定の製品にあてはまる利用可能なサブスクリプションの対象一覧を表示します (サブスクリプションが利用可能と仮定)。

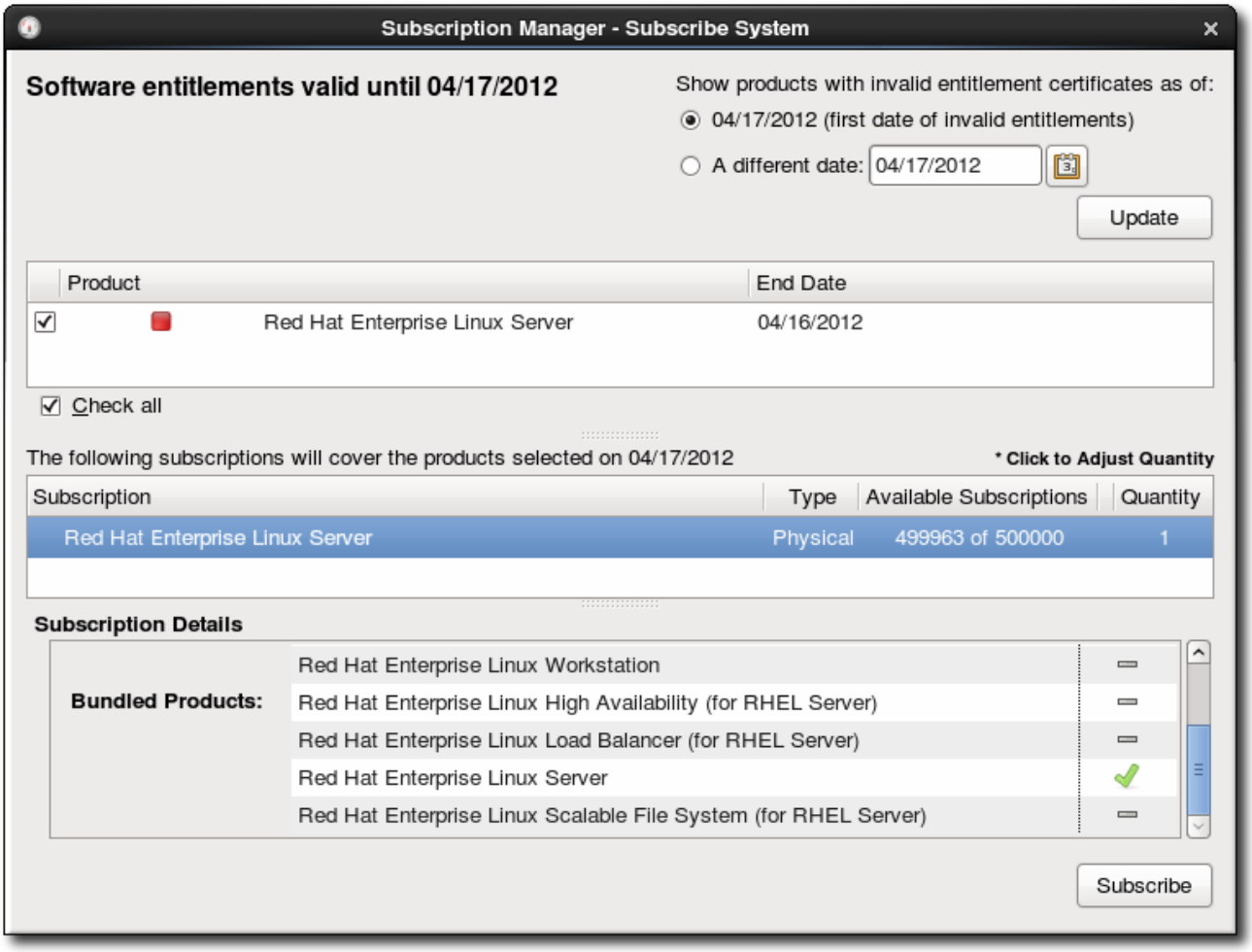
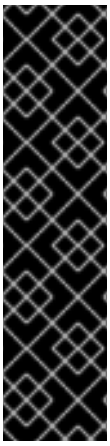


図5.7 システムのサブスクライブ

第6章 YUM

Yum は Red Hat パッケージマネージャーです。Yum は、パッケージ情報に関するクエリ、リポジトリからのパッケージのフェッチ、パッケージのインストールとアンインストール、さらには利用可能な最新バージョンへのシステム全体の更新を行うことができます。Yum は、更新/インストール/削除を実行しているパッケージで依存関係の自動解決を行います。そのため、利用可能なすべての依存パッケージを自動的に決定/フェッチ/インストールすることができます。

Yum は、新しい追加のリポジトリまたは *パッケージソース* を使って設定することができ、機能を向上して拡張する多くのプラグインも提供します。また、Yum は **RPM** が実行できる多くの同じタスクを行うことができます。さらに、多くのコマンドラインオプションも似ています。Yum を使うことで、1 つのマシンまたはマシンのグループ上でのパッケージ管理を簡単かつシンプルに行うことができます。



重要

Yum は、GPG (Gnu Privacy Guard: 別名 GnuPG) 署名付きのパッケージの GPG 署名照合をすべてのパッケージリポジトリ (パッケージソース) または個々のリポジトリに有効にすることで、セキュアなパッケージ管理を実現します。署名照合が有効な場合、Yum はリポジトリに対して GPG 署名がない不正確なキーを持つパッケージのインストールを拒否します。これにより、ご使用のシステムにダウンロードしてインストールする **RPM** パッケージが Red Hat などの信頼されたソースからのものであり、移行中に変更されていないことが信用できます。Yum を使用して署名照合を有効にする方法については、「Yum と Yum リポジトリの設定」 を参照して下さい。また、GPG 署名付きの **RPM** パッケージ全般の使用及び検証方法に関しては 「パッケージの署名確認」 を参照して下さい。

Yum を使用すると、他のマシンへダウンロード/インストールするための **RPM** パッケージのリポジトリを簡単に設定することができます。

Yum を使用するとシステム管理タスクを最速で実行できることが多いため、習得する価値は十分にあります。また、Yum は **PackageKit** グラフィカルパッケージ管理ツールが実現できる以上の機能を提供します。**PackageKit** の使用に関する詳細は、*7章PackageKit* を参照して下さい。

注記

yum を使用して、システムにパッケージをインストール/更新/削除するにはスーパーユーザー権限が必要です。本章のすべての例では、**su** または **sudo** コマンドを使用することでスーパーユーザー権限をすでに持っていると仮定しています。

6.1. パッケージの確認と更新

6.1.1. 更新の確認

使用しているシステムに利用可能な更新があるインストール済みのパッケージを確認するには、以下のコマンドを実行します:

```
yum check-update
```

例:

```
~]# yum check-update
Loaded plugins: product-id, refresh-packagekit, subscription-manager
Updating Red Hat repositories.
```

```
INFO:rhsm-app.repolib:repos updated: 0
PackageKit.x86_64                   0.5.8-2.el6                 rhel
PackageKit-glib.x86_64              0.5.8-2.el6                 rhel
PackageKit-yum.x86_64               0.5.8-2.el6                 rhel
PackageKit-yum-plugin.x86_64        0.5.8-2.el6                 rhel
glibc.x86_64                        2.11.90-20.el6              rhel
glibc-common.x86_64                 2.10.90-22                  rhel
kernel.x86_64                       2.6.31-14.el6               rhel
kernel-firmware.noarch              2.6.31-14.el6               rhel
rpm.x86_64                          4.7.1-5.el6                 rhel
rpm-libs.x86_64                     4.7.1-5.el6                 rhel
rpm-python.x86_64                   4.7.1-5.el6                 rhel
udev.x86_64                         147-2.15.el6                rhel
yum.noarch                          3.2.24-4.el6                rhel
```

上記の出力に表示されているパッケージには、利用可能な更新があります。一覧の最初のパッケージは、**PackageKit** と呼ばれるグラフィカルパッケージマネージャーです。その行の意味は以下のとおりです。

- **PackageKit** — パッケージの名前
- **x86_64** — パッケージがビルドされた CPU アーキテクチャー
- **0.5.8** — インストールされる更新済みパッケージのバージョン
- **rhel** — 更新済みのパッケージがあるリポジトリ

また、上記の出力は、カーネル (kernel パッケージ)、Yum および RPM (yum および rpm のパッケージ)、さらにはそれらの依存関係 (kernel-firmware、rpm-libs、rpm-python のパッケージ) をすべて **yum** を使って更新できることも示しています。

6.1.2. パッケージの更新

1 回に更新するパッケージ数を 1 つ、複数、またはすべてのパッケージから選択することができます。更新するパッケージの依存関係 (またはパッケージ) に利用可能な更新がある場合は、それらも更新されます。

単一パッケージの更新
単一パッケージを更新するには、**root** で以下のコマンドを実行して下さい:

```
yum update package_name
```

例として、udev パッケージを更新するには以下のとおり入力します:

```
~]# yum update udev
Loaded plugins: product-id, refresh-packagekit, subscription-manager
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
Setting up Update Process
Resolving Dependencies
--> Running transaction check
---> Package udev.x86_64 0:147-2.15.el6 set to be updated
--> Finished Dependency Resolution

Dependencies Resolved
```

```
================================================================================
=
 Package        Arch             Version                   Repository
Size
================================================================================
=
Updating:
 udev           x86_64           147-2.15.el6              rhel           337
k

Transaction Summary
================================================================================
=
Install       0 Package(s)
Upgrade       1 Package(s)

Total download size: 337 k
Is this ok [y/N]:
```

この出力には興味を引く項目が数点あります。

1. **Loaded plugins: product-id, refresh-packagekit, subscription-manager** — **yum** は、どの Yum プラグインがインストールされ有効であるかを常に通知します。Yum プラグインに関する全般情報は 「Yum のプラグイン」 を参照してください。また、個別のプラグインに関する説明は 「プラグインの詳細」 を参照してください。

2. **udev.x86_64** — 新しい udev パッケージをダウンロードしてインストールできます。

3. **yum** は、更新情報を表示して、更新を実行したいかどうかを確認します。**yum** はデフォルトではインタラクティブに実行します。**yum** コマンドがどのトランザクションを実行するかが事前に分かっている場合は、**-y** オプションを使用して、**yum** が行うすべての質問に **yes** と自動回答するように設定できます (この場合はノンインタラクティブに実行されます)。ただし、**yum** がシステムに実行しようとしている変更を常に調べることをお勧めします。そうすることで、起こり得る問題を簡単に解決することができます。

   トランザクションが正しく行われなかった場合は、**yum history** コマンドを使用して 「トランザクション履歴の活用」 にあるように Yum のトランザクション履歴を閲覧することができます。

重要

rpm -i kernel のコマンドを使用すると、**RPM** が新しいカーネルをインストールするのと同じように、**yum** は常に新しいカーネルを **インストール** します。そのため、**yum** を使用する場合は、カーネルパッケージの **インストール** と **アップグレード** の区別について気にする必要はありません。つまり、**yum update** か **yum install** のコマンドどちらを使用するかに関わらず正しく実行します。

一方で、**RPM** を使用する場合は、**rpm -u kernel** (現在のカーネルと **置換**) の代わりに、**rpm -i kernel** コマンド (新しいカーネルのインストール) を使用することが重要です。**RPM** を使ったカーネルのインストール/アップグレードについての詳細は、「インストールとアップグレード」 を参照してください。

すべてのパッケージとそれらの依存関係の更新

すべてのパッケージとそれらの依存関係を更新するには、**yum update** と (引数なしで) 入力するだけです:

```
yum update
```

セキュリティ関連パッケージの更新
どのパッケージに利用可能なセキュリティの更新があるのかを見つけ出し、そうしたパッケージを迅速かつ簡単に更新することは重要です。このため、Yum はプラグインを提供します。**security** プラグインは、**yum** コマンドを非常に便利なセキュリティ中心のコマンド、サブコマンド、オプションのセットにより拡張します。具体的な情報は 「プラグインの詳細」 を参照して下さい。

6.1.3. 設定ファイルの変更の保存

Red Hat Enterprise Linux システムを使用していると、パッケージによりインストールされた設定ファイルに変更を加える必要がでてきます。システムに変更を加えるために Yum が使用する **RPM** には、整合性を確保する仕組みが備わっています。パッケージのアップグレード全般に渡る設定ファイルへの変更を管理する方法の詳細については、「インストールとアップグレード」 を参照して下さい。

6.2. パッケージとパッケージグループ

6.2.1. パッケージの検索

以下のコマンドを使用することで、すべての RPM のパッケージ名、詳細、サマリを検索することができます:

```
yum search term…
```

このコマンドにより、各用語に一致する項目を示した一覧が表示されます。例えば 「meld」 や 「kompare」 に一致するすべてのパッケージを一覧表示するには、以下を入力します:

```
~]# yum search meld kompare
Loaded plugins: product-id, refresh-packagekit, subscription-manager
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
============================== Matched: kompare
==============================
kdesdk.x86_64 : The KDE Software Development Kit (SDK)
Warning: No matches found for: meld
```

yum search のコマンドは、パッケージの名前は不明だがその関連用語を知っている場合の検索に役立ちます。

6.2.2. パッケージの一覧表示

yum list と関連コマンドは、パッケージ、パッケージグループ、リポジトリに関する情報を提供します。

Yum の全 list コマンドにより、1 つ以上の **glob 表現** を引数として追加することで結果をフィルターすることができます。glob 表現は、1 つ以上のワイルドカード文字 ***** (任意の文字に複数回一致) と **?** (任意の 1 文字に一致) を含む通常の文字列です。

注記

yum のコマンドに引数として glob 表現を渡す場合には、glob 表現をエスケープするように注意して下さい。そうしないと、bash シェルはこの表現を **パス名の展開** と解釈してしまい、glob と適合する現在のディレクトリ内の全ファイルを **yum** に渡す可能性があります。確実に glob 表現を **yum** に正しく渡すためには、以下のいずれかの方法を実行します:

- ワイルドカード文字の前にバックスラッシュ記号を入力して、ワイルドカード文字をエスケープする
- glob 表現全体を二重引用符または単一引用符でくくる

上記の方法を使ったサンプルは、例6.1「glob 表現を使ったすべての ABRT アドオンとプラグインの一覧表示」 と 例6.3「エスケープされたワイルドカード文字を含む単一の glob 表現を使った利用可能なパッケージの一覧表示」 を参照して下さい。

yum list *glob_expression*…

すべての glob 表現に一致する、インストール済みで利用可能なパッケージに関する情報を一覧表示します。

例6.1 glob 表現を使ったすべての ABRT アドオンとプラグインの一覧表示

各種の ABRT アドオンとプラグインを持つパッケージは 「abrt-addon-」か「abrt-plugin-」で始まります。こうしたパッケージを一覧表示するには、シェルプロンプトで以下を入力します:

```
~]# yum list abrt-addon\* abrt-plugin\*
Loaded plugins: product-id, refresh-packagekit, subscription-manager
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
Installed Packages
abrt-addon-ccpp.x86_64                          1.0.7-5.el6
@rhel
abrt-addon-kerneloops.x86_64                    1.0.7-5.el6
@rhel
abrt-addon-python.x86_64                        1.0.7-5.el6
@rhel
abrt-plugin-bugzilla.x86_64                     1.0.7-5.el6
@rhel
abrt-plugin-logger.x86_64                       1.0.7-5.el6
@rhel
abrt-plugin-sosreport.x86_64                    1.0.7-5.el6
@rhel
abrt-plugin-ticketuploader.x86_64               1.0.7-5.el6
@rhel
```

yum list all

すべてのインストール済み **かつ** 利用可能なパッケージを一覧表示します。

yum list installed

システムにインストール済みのすべてのパッケージを一覧表示します。出力の右端のカラムには、パッケージが取得されたリポジトリが一覧表示されています。

例6.2 二重引用符で囲まれた glob 表現を使ったインストール済みパッケージの一覧表示

「krb」 の後に 1 文字とハイフンが続く名前のすべてのインストール済みパッケージを一覧表示するには、以下を入力して下さい:

```
~]# yum list installed "krb?-*"
Loaded plugins: product-id, refresh-packagekit, subscription-manager
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
Installed Packages
krb5-libs.x86_64                          1.8.1-3.el6
@rhel
krb5-workstation.x86_64                   1.8.1-3.el6
@rhel
```

yum list available

すべての有効なリポジトリにある利用可能な全パッケージを一覧表示します。

例6.3 エスケープされたワイルドカード文字を含む単一の glob 表現を使った利用可能なパッケージの一覧表示

「gstreamer」 とその後に 「plugin」 を含む名前の利用可能なすべてのパッケージを一覧表示するには、以下のコマンドを実行します:

```
~]# yum list available gstreamer\*plugin\*
Loaded plugins: product-id, refresh-packagekit, subscription-manager
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
Available Packages
gstreamer-plugins-bad-free.i686              0.10.17-4.el6
rhel
gstreamer-plugins-base.i686                  0.10.26-1.el6
rhel
gstreamer-plugins-base-devel.i686            0.10.26-1.el6
rhel
gstreamer-plugins-base-devel.x86_64          0.10.26-1.el6
rhel
gstreamer-plugins-good.i686                  0.10.18-1.el6
rhel
```

yum grouplist

すべてのパッケージグループを一覧表示します。

yum repolist

リポジトリ ID、名前、それぞれの **有効な** リポジトリに提供するパッケージ数を一覧表示します。

6.2.3. パッケージ情報の表示

1 つ以上のパッケージ (glob 表現はここでも有効) に関する情報を表示するには、以下のコマンドを実行します:

```
yum info package_name…
```

例えば abrt パッケージに関する情報を表示するには、以下を入力します:

```
~]# yum info abrt
Loaded plugins: product-id, refresh-packagekit, subscription-manager
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
Installed Packages
Name       : abrt
Arch       : x86_64
Version    : 1.0.7
Release    : 5.el6
Size       : 578 k
Repo       : installed
From repo  : rhel
Summary    : Automatic bug detection and reporting tool
URL        : https://fedorahosted.org/abrt/
License    : GPLv2+
Description: abrt is a tool to help users to detect defects in
applications
           : and to create a bug report with all informations needed by
           : maintainer to fix it. It uses plugin system to extend its
           : functionality.
```

yum info *package_name* コマンドは **rpm -q --info *package_name*** コマンドに似ていますが、**yum info *package_name*** は RPM パッケージがある Yum リポジトリの ID を追加情報として提供します (出力の **From repo:** の行を参照)。

以下のコマンドを使用することで、パッケージに関する代替情報や有用な情報について Yum データベースにクエリすることもできます:

```
yumdb info package_name
```

このコマンドは、パッケージに関する追加情報を提供します。例えば、パッケージのチェックサム (及びチェックサムを算出するための SHA-256 などのアルゴリズム)、パッケージをインストールするために呼び出されたコマンドラインのコマンド (存在する場合)、パッケージがシステムにインストールされる理由 (**user** はユーザーによりインストール済み、**dep** は依存関係として取り入れたという意味) などです。例として、yum パッケージに関する追加情報を表示するには、以下を入力します:

```
~]# yumdb info yum
Loaded plugins: product-id, refresh-packagekit, subscription-manager
yum-3.2.27-4.el6.noarch
     checksum_data =
23d337ed51a9757bbfbdceb82c4eaca9808ff1009b51e9626d540f44fe95f771
     checksum_type = sha256
     from_repo = rhel
     from_repo_revision = 1298613159
     from_repo_timestamp = 1298614288
     installed_by = 4294967295
     reason = user
     releasever = 6.1
```

yumdb コマンドの詳細については、**yumdb**(8) の man ページを参照して下さい。

6.2.4. パッケージのインストール

Yum を使用したインストールでは、単一及び複数のパッケージ、さらには希望するパッケージグループもインストールすることができます。

個々のパッケージのインストール
単一パッケージとそのインストールされていないすべての依存関係をインストールするには、以下の形式でコマンドを入力します:

```
yum install package_name
```

引数としてパッケージの名前を追加することで、複数のパッケージを同時にインストールすることも可能です:

```
yum install package_name package_name…
```

AMD64 や Intel64 マシンのような *multilib* システムにパッケージをインストールする場合は、パッケージ名に *.arch* を追加することでパッケージのアーキテクチャーを指定できます (ただし、それが有効なリポジトリで利用可能な場合のみ)。例えば **i586** に sqlite2 パッケージをインストールするには、以下を入力します:

```
~]# yum install sqlite2.i586
```

glob 表現を使うと、複数の類似した名前のパッケージを迅速にインストールできます:

```
~]# yum install audacious-plugins-\*
```

パッケージ名と glob 表現だけでなく、**yum install** にはファイル名も加えることができます。インストールしたいバイナリ名は知っているが、パッケージ名が分からない場合は、**yum install** にパス名を付けて実行します:

```
~]# yum install /usr/sbin/named
```

yum はパッケージ一覧で検索を行い、**/usr/sbin/named** を提供するパッケージを探します。存在すれば、**yum** はインストールしたいかどうかを確認します。

注記

named バイナリを含むパッケージをインストールしたいが、ファイルがインストールされているのが **bin** ディレクトリか **sbin** ディレクトリか分からない場合は、glob 表現を付けて **yum provides** コマンドを実行します:

```
~]# yum provides "*bin/named"
Loaded plugins: product-id, refresh-packagekit, subscription-
manager
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
32:bind-9.7.0-4.P1.el6.x86_64 : The Berkeley Internet Name
Domain (BIND)
                              : DNS (Domain Name System)
server
Repo        : rhel
Matched from:
Filename    : /usr/sbin/named
```

yum provides "*/*file_name*" は *file_name* を含むパッケージを見つけるための一般的で便利な方法です。

パッケージグループのインストール

パッケージグループはパッケージと似ています。それだけでは役に立ちませんが、パッケージグループを 1 つインストールすると、共通の目的に合う依存関係のパッケージのグループをプルします。パッケージグループには名前と *groupid* があります。**yum grouplist -v** コマンドでは、すべてのパッケージグループ名と、各名前の横に丸括弧で groupid を一覧表示します。groupid は、常に丸括弧内の一番最後の単語になります。以下の例では **kde-desktop** がそれにあたります:

```
~]# yum -v grouplist kde\*
Loading "product-id" plugin
Loading "refresh-packagekit" plugin
Loading "subscription-manager" plugin
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
Config time: 0.123
Yum Version: 3.2.29
Setting up Group Process
Looking for repo options for [rhel]
rpmdb time: 0.001
group time: 1.291
Available Groups:
   KDE Desktop (kde-desktop)
Done
```

パッケージグループをインストールするには、**groupinstall** にグループ名全体 (groupid を含まず) を渡します:

```
yum groupinstall group_name
```

groupid を使ってインストールすることもできます:

```
yum groupinstall groupid
```

@-記号を付加することで、groupid (または引用名) を **install** コマンドに渡すことも可能です (これで **groupinstall** を実行したいというメッセージを **yum** に伝えます):

```
yum install @group
```

以下の例では、**KDE Desktop** グループのインストールを同等の別の方法で示しています:

```
~]# yum groupinstall "KDE Desktop"
~]# yum groupinstall kde-desktop
~]# yum install @kde-desktop
```

6.2.5. パッケージの削除

パッケージのインストールと同様に、Yum を使用すると個々のパッケージとパッケージグループの両方をアンインストールすることができます (**RPM** と Yum 用語では remove に該当)。

個々のパッケージの削除
特定のパッケージとそれに依存するすべてのパッケージをアンインストールするには、**root** で以下のコマンドを実行して下さい:

```
yum remove package_name…
```

複数のパッケージをインストールする場合と同様に、コマンドに複数のパッケージ名を追加することで一度に複数のパッケージを削除することができます。例えば、totem、rhythmbox、sound-juicer を削除するには、シェルプロンプトで以下のように入力します:

```
~]# yum remove totem rhythmbox sound-juicer
```

install と同じように、**remove** は以下の引数を取ることができます。

- パッケージ名
- glob 表現
- ファイル一覧
- パッケージが提供する機能

警告

Yum では、パッケージに依存しているパッケージを削除することなくそのパッケージを削除することはできません。こうした動作は **RPM** のみ実行でき、推奨されません。システムが機能しなくなる、またはアプリケーションに誤作動/クラッシュが生じる恐れがあるためです。詳細については、**RPM** の章の 「アンインストール」 を参照して下さい。

パッケージグループの削除
install 構文と同じ形の構文を使用してパッケージグループを削除することができます:

```
yum groupremove group
```

```
yum remove @group
```

以下は、**KDE Desktop** グループの削除を同等の別の方法で示しています:

```
~]# yum groupremove "KDE Desktop"
~]# yum groupremove kde-desktop
~]# yum remove @kde-desktop
```

重要

yum にパッケージグループを削除するよう伝えると、**yum** はそれらのパッケージが他のパッケージグループのメンバーまたは他のインストール済みのパッケージの依存関係であったとしても、そのパッケージグループ内にある全パッケージを削除します。ただし、**yum** に他のパッケージやグループから必要ないパッケージのみを削除するよう伝えたい場合は、**/etc/yum.conf** 設定ファイルの **[main]** セクションに **groupremove_leaf_only=1** 指示文を追加します。この指示文の詳細については 「[main] オプションの設定」 を参照して下さい。

6.2.6. トランザクション履歴の活用

yum history のコマンドを使用すると、ユーザーは Yum のトランザクションのタイムライン、トランザクションの発生日時、影響を受けたパッケージ数、トランザクション成功の有無、RPM データベースがトランザクション中に変更されたかどうかに関する情報を確認することができます。さらには、このコマンドを使うと、特定のトランザクションを元に戻す/やり直すことが可能です。

トランザクションの一覧表示
最近発生した 20 件のトランザクションの一覧を表示するには、 **root** として引数なしで **yum history** を実行するか、シェルプロンプトで以下を入力して下さい:

```
yum history list
```

すべてのトランザクションを表示するには、**all** のキーワードを追加します:

```
yum history list all
```

特定の範囲内のトランザクションのみを表示したい場合は、以下の形式でコマンドを使用します:

```
yum history list start_id..end_id
```

特定のパッケージに関するトランザクションのみを一覧表示することも可能です。そのためには、パッケージ名か glob 表現を付けてコマンドを実行します:

```
yum history list glob_expression…
```

例えば、最初の 5 件のトランザクションを示した一覧は以下のようになります。

```
~]# yum history list 1..5
Loaded plugins: product-id, refresh-packagekit, subscription-manager
ID     | Login user               | Date and time    | Action(s)      |
```

```
Altered
-------------------------------------------------------------------------
---------
     5 | Jaromir ... <jhradilek>  | 2011-07-29 15:33 | Install        |
1
     4 | Jaromir ... <jhradilek>  | 2011-07-21 15:10 | Install        |
1
     3 | Jaromir ... <jhradilek>  | 2011-07-16 15:27 | I, U           |
73
     2 | System <unset>           | 2011-07-16 15:19 | Update         |
1
     1 | System <unset>           | 2011-07-16 14:38 | Install        |
1106
history list
```

yum history list コマンドのすべての形式は、以下のカラムで構成される各行を含む表形式出力を生成します。

- **ID** — 特定のトランザクションを識別する整数値です。
- **Login user** — ユーザー名です。このログインセッションはトランザクションを開始するために使用されました。この情報は、通常 ***Full Name <username>*** の形式で表示されます。ユーザーにより発行されなかったトランザクションに関しては (システムの自動更新など)、代わりに **System <unset>** が使用されます。
- **Date and time** — トランザクションが発生した日時です。
- **Action(s)** — 表6.1「Action 欄の指定可能な値」 の説明のとおり、トランザクション中に実行された動作の一覧です。
- **Altered** — 表6.2「Altered 欄の指定可能な値」 の説明のとおり、トランザクションにより影響を受けたパッケージ数、場合によっては追加情報も含まれます。

表6.1 Action 欄の指定可能な値

Action	省略形	詳細
Downgrade	**D**	1 つ以上のパッケージが旧バージョンにダウングレードされました。
Erase	**E**	1 つ以上のパッケージが削除されました。
Install	**I**	1 つ以上の新しいパッケージがインストールされました。
Obsoleting	**O**	1 つ以上のパッケージが廃止として記録されました。
Reinstall	**R**	1 つ以上のパッケージが再インストールされました。
Update	**U**	1 つ以上のパッケージが新しいバージョンに更新されました。

表6.2 Altered 欄の指定可能な値

記号	詳細
<	トランザクションが終了する前に、**rpmdb** データベースが Yum 以外で変更されました。
>	トランザクションが終了した後に、**rpmdb** データベースが Yum 以外で変更されました。
*****	トランザクションは失敗して終了しました。
#	トランザクションは正常に終了しましたが、**yum** はゼロ以外の終了コードを返しました。
E	トランザクションは正常に終了しましたが、エラーまたは警告が表示されました。
P	トランザクションは正常に終了しましたが、**rpmdb** データベースに問題がすでに存在していました。
s	トランザクションは正常に終了しましたが、**--skip-broken** コマンドラインオプションが使用され、特定のパッケージが省略されました。

Yum を使用すると、過去に発生したすべてのトランザクションのサマリを表示することもできます。そのためには、**root** として以下の形式でコマンドを実行して下さい:

```
yum history summary
```

特定の範囲内でのトランザクションのみを表示するには、以下を入力します:

```
yum history summary start_id..end_id
```

yum history list コマンドと同様に、パッケージの名前または glob 表現を指定することで、特定のパッケージに関するトランザクションのサマリを表示することも可能です:

```
yum history summary glob_expression…
```

例えば、上記に表示されたトランザクションの履歴のサマリは以下のようになります:

```
~]# yum history summary 1..5
Loaded plugins: product-id, refresh-packagekit, subscription-manager
Login user                | Time                | Action(s)        |
Altered
------------------------------------------------------------------------
---------
Jaromir ... <jhradilek>   | Last day            | Install          |
1
Jaromir ... <jhradilek>   | Last week           | Install          |
1
Jaromir ... <jhradilek>   | Last 2 weeks        | I, U             |
73
```

```
System <unset>             | Last 2 weeks        | I, U             |
1107
history summary
```

yum history summary コマンドの全形式は、**yum history list** の出力に似た、簡略化した表形式出力を生成します。

上記のとおり、**yum history list** 及び **yum history summary** とも、トランザクション向けに設定されています。特定のパッケージに関連するトランザクションのみを表示することができますが、パッケージバージョンのような重要な詳細は表示されません。パッケージに関連するトランザクションを一覧表示するには、**root** で以下のコマンドを実行して下さい:

```
yum history package-list glob_expression…
```

例えば、subscription-manager 及び関連パッケージの履歴を調べるには、シェルプロンプトで以下を入力します:

```
~]# yum history package-list subscription-manager\*
Loaded plugins: product-id, refresh-packagekit, subscription-manager
ID     | Action(s)      | Package
----------------------------------------------------------------------
---------
     3 | Updated        | subscription-manager-0.95.11-1.el6.x86_64
     3 | Update         |                       0.95.17-1.el6_1.x86_64
     3 | Updated        | subscription-manager-firstboot-0.95.11-
1.el6.x86_64
     3 | Update         |                                0.95.17-
1.el6_1.x86_64
     3 | Updated        | subscription-manager-gnome-0.95.11-1.el6.x86_64
     3 | Update         |                              0.95.17-
1.el6_1.x86_64
     1 | Install        | subscription-manager-0.95.11-1.el6.x86_64
     1 | Install        | subscription-manager-firstboot-0.95.11-
1.el6.x86_64
     1 | Install        | subscription-manager-gnome-0.95.11-1.el6.x86_64
history package-list
```

上記の例では、初期のシステムインストール時に subscription-manager、subscription-manager-firstboot、subscription-manager-gnome の 3 パッケージがインストールされました。3 つ目のトランザクションでは、これらの全パッケージはバージョン 0.95.11 から 0.95.17 に更新されました。

トランザクションの検証

1 つのトランザクションのサマリを表示するには、**root** として以下の形式で **yum history summary** コマンドを使用します:

```
yum history summary id
```

特定のトランザクションを詳しく調べたい場合は、**root** で以下のコマンドを実行して下さい:

```
yum history info id…
```

id の引数はオプションです。これを省略する場合は、**yum** は自動的に最後のトランザクションを使用します。複数のトランザクションを指定する場合は、範囲を指定することもできます:

```
yum history info start_id..end_id
```

以下は、2 つのトランザクションに関する出力のサンプルです。それぞれ新しいパッケージを 1 つインストールしています:

```
~]# yum history info 4..5
Loaded plugins: product-id, refresh-packagekit, subscription-manager
Transaction ID : 4..5
Begin time     : Thu Jul 21 15:10:46 2011
Begin rpmdb    : 1107:0c67c32219c199f92ed8da7572b4c6df64eacd3a
End time       :            15:33:15 2011 (22 minutes)
End rpmdb      : 1109:1171025bd9b6b5f8db30d063598f590f1c1f3242
User           : Jaromir Hradilek <jhradilek>
Return-Code    : Success
Command Line   : install screen
Command Line   : install yum-plugin-fs-snapshot
Transaction performed with:
    Installed     rpm-4.8.0-16.el6.x86_64
    Installed     yum-3.2.29-17.el6.noarch
    Installed     yum-metadata-parser-1.1.2-16.el6.x86_64
Packages Altered:
    Install screen-4.0.3-16.el6.x86_64
    Install yum-plugin-fs-snapshot-1.1.30-6.el6.noarch
history info
```

また、トランザクション時に使用された設定オプション、特定のパッケージをインストールしたリポジトリ及びその理由などの追加情報も閲覧できます。特定のトランザクションに関して入手可能な追加情報を知りたい場合は、**root** としてシェルプロンプトで以下を入力して下さい:

```
yum history addon-info id
```

yum history info と同様に、*id* が指定されていない場合は、**yum** は自動的に最新のトランザクションを使用します。別の方法として、最新のトランザクションを参照するには、**last** キーワードを使用することもできます:

```
yum history addon-info last
```

例えば、前述の例にある最初のトランザクションでは、**yum history addon-info** コマンドにより以下の出力が生成されます:

```
~]# yum history addon-info 4
Loaded plugins: product-id, refresh-packagekit, subscription-manager
Transaction ID: 4
Available additional history information:
  config-main
  config-repos
  saved_tx

history addon-info
```

この例では、3 種類の情報があります。

- **config-main** — トランザクション時に使用された Yum のグローバルオプションです。グローバルオプションを変更する方法の詳細については、「[main] オプションの設定」 を参照して下さい。
- **config-repos** — 個々の Yum リポジトリ用のオプションです。個々のリポジトリ用のオプションを変更する方法については、「[repository] オプションの設定」 を参照して下さい。
- **saved_tx** — 別のマシンでトランザクションを繰り返すために **yum load-transaction** コマンドにより利用できるデータです (下記参照)。

選択した種類の追加情報を表示するには、**root** で以下のコマンドを実行して下さい:

```
yum history addon-info id information
```

トランザクションを元に戻す/繰り返す方法

トランザクション履歴の確認以外に、**yum history** コマンドは選択したトランザクションを元に戻す/繰り返す方法を提供します。トランザクションを元に戻すには、**root** としてシェルプロンプトで以下を入力して下さい:

```
yum history undo id
```

特定のトランザクションを繰り返すには、**root** で以下のコマンドを実行します:

```
yum history redo id
```

どちらのコマンドでも **last** キーワードを使用して、最新のトランザクションを元に戻す/繰り返すことができます。

yum history undo および **yum history redo** コマンドともトランザクション中に実行された手順を元に戻す/繰り返すだけである点に注意してください。トランザクションにより新しいパッケージがインストールされた場合、**yum history undo** コマンドはそれをアンインストールします。トランザクションによりパッケージがアンインストールされた場合、このコマンドは再度インストールします。また、このコマンドはすべての更新済みパッケージを以前のバージョンにダウングレードする試行も行います (古いパッケージが引き続き利用可能な場合)。システムを更新前の状態に復旧する必要がある場合は、「プラグインの詳細」 の説明のとおり **fs-snapshot** プラグインの使用を検討してください。

複数の同一システムを管理する場合、Yum を使用すると、1 つのシステム上でトランザクションを実行して、そのトランザクションの詳細をファイルに格納し、テスト期間の終了後に残りのシステム上で同じトランザクションを繰り返すことが可能です。トランザクションの詳細をファイルに格納するには、**root** としてシェルプロンプトで以下を入力します:

```
yum -q history addon-info id saved_tx > file_name
```

このファイルを目的のシステムにコピーしたら、**root** で以下のコマンドを使用してトランザクションを繰り返すことができます:

```
yum load-transaction file_name
```

ただし、ファイルに格納されている **rpmdb** バージョンは目的のシステムのバージョンと同一でなければならない点に注意してください。**rpmdb** のバージョンを調べるには、**yum version nogroups** コマンドを使用します。

新しいトランザクション履歴の開始

Yum は単一の SQLite データベースファイルにトランザクション履歴を格納します。新しいトランザクションの履歴を開始するには、**root** で以下のコマンドを実行して下さい:

```
yum history new
```

これにより **/var/lib/yum/history/** ディレクトリ内に新しい空のデータベースファイルが作成されます。古いトランザクション履歴は保存されますが、新しいデータベースファイルがディレクトリにある限りアクセスすることはできません。

6.3. YUM と YUM リポジトリの設定

yum および関連ユーティリティの設定ファイルは **/etc/yum.conf** にあります。このファイルには、必須の **[main]** セクションが 1 つあり、全体に影響を与える Yum オプションを設定できます。また、1 つ以上の **[*repository*]** セクションを含むこともでき、リポジトリ固有のオプションを設定できます。ただし、**/etc/yum.repos.d/** ディレクトリ内にある、新規または既存の **.repo** ファイルで個々のリポジトリを定義することが推奨されます。**/etc/yum.conf** ファイルの **[main]** セクションで定義する値は、個々の **[*repository*]** セクションで設定された値を上書きする場合があります。

本項では以下の方法を紹介します。

- **/etc/yum.conf** 設定ファイルの **[main]** セクションを編集して Yum のグローバルオプションを設定する方法
- **/etc/yum.conf** の **[*repository*]** セクションと **/etc/yum.repos.d/** ディレクトリ内の **.repo** ファイルを編集することで、個々のリポジトリのオプションを設定する方法
- 動的バージョンとアーキテクチャーの値が適切に処理されるように **/etc/yum.conf** の Yum 変数と **/etc/yum.repos.d/** ディレクトリ内のファイルを使用する方法
- コマンドラインで Yum リポジトリを追加/有効/無効にする方法
- カスタムの Yum リポジトリを設定する方法

6.3.1. [main] オプションの設定

/etc/yum.conf 設定ファイルには、1 つの **[main]** セクションが含まれます。このセクションにあるキー値ペアの中には **yum** の動作に影響を与えるものもあれば、Yum のリポジトリの処理方法に影響を与えるものもあります。**/etc/yum.conf** 内にある **[main]** のセクション見出しの下に、多くのオプションを追加することができます。

以下は、**/etc/yum.conf** 設定ファイルのサンプルです:

```
[main]
cachedir=/var/cache/yum/$basearch/$releasever
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
exactarch=1
obsoletes=1
gpgcheck=1
plugins=1
installonly_limit=3

[コメントは省略されています ]
```

```
# PUT YOUR REPOS HERE OR IN separate files named file.repo
# in /etc/yum.repos.d
```

以下は、**[main]** セクションで最もよく使用されるオプションです。

assumeyes=*value*

value は以下のいずれかです。

0 — **yum** は実行する重要な動作の確認をプロンプトします。これはデフォルトです。

1 — **yum** が実行する重要な動作の確認をプロンプトしません。**assumeyes=1** に設定すると、**yum** はコマンドラインのオプション **-y** が行うとの同じように実行します。

cachedir=*directory*

directory は、Yum がキャッシュとデータベースファイルを格納するディレクトリへの絶対パスです。デフォルトでは、Yum のキャッシュディレクトリは **/var/cache/yum/$basearch/$releasever** です。

$basearch 及び **$releasever** Yum 変数の詳細については、「Yum 変数の使用」 を参照して下さい。

debuglevel=*value*

value は **1** から **10** までの整数です。**debuglevel** 値を高い値に設定すると、**yum** はより詳しいデバッグ出力を表示します。**debuglevel=0** は、デバッグ出力を無効にします。デフォルトは **debuglevel=2** です。

exactarch=*value*

value は以下のいずれかです。

0 — パッケージの更新時には正しいアーキテクチャーを考慮に入れて実行しません。

1 — パッケージの更新時には正しいアーキテクチャーを考慮に入れて実行します。この設定では、**yum** はシステムにすでにインストール済みの i386 パッケージを更新するために i686 パッケージをインストールしません。これがデフォルトです。

exclude=*package_name* [*more_package_names*]

このオプションを使用すると、インストール/更新中にキーワードを使ってパッケージを除外することができます。除外する複数のパッケージを一覧表示したい場合は、パッケージを空白で区切ります。ワイルドカード (***** や **?** など) を使ったシェルグロブが使用できます。

gpgcheck=*value*

value は以下のいずれかです。

0 — インストールされるローカルパッケージなど、全リポジトリ内のパッケージ上での GPG 署名確認を無効にします。

1 — インストールされるローカルパッケージなど、全リポジトリ内の全パッケージ上での GPG 署名確認を有効にします。**gpgcheck=1** がデフォルトであるため、すべてのパッケージ署名は確認されます。

このオプションが **/etc/yum.conf** ファイルの **[main]** セクションで設定されている場合は、全リ

ポジトリに対して GPG 照合規則を設定します。ただし、代わりに個々のリポジトリに **gpgcheck=*value*** を設定することもできます。つまり、あるリポジトリで GPG 照合を有効にしながら別のリポジトリでは無効にすることができます。個々のリポジトリに対して対応する **.repo** ファイルで **gpgcheck=*value*** を設定すると、**/etc/yum.conf** にデフォルト値がある場合はそれを無効にします。

GPG 署名照合の詳細については、「パッケージの署名確認」 を参照して下さい。

groupremove_leaf_only=*value*

value は以下のいずれかです。

0 — **yum** はパッケージグループの削除時に、各パッケージの依存関係を確認 **しません**。この設定により、**yum** はパッケージが他のパッケージまたはグループにより必要とされているかどうかに関わらず、パッケージグループ内の全パッケージを削除します。デフォルトは **groupremove_leaf_only=0** です。

1 — **yum** はパッケージグループの削除時に、各パッケージの依存関係を確認して、他のパッケージやグループにより必要とされていないパッケージのみを削除します。

パッケージの削除についての詳細は インテリジェントなパッケージグループの削除方法 を参照して下さい。

installonlypkgs=*space separated list of packages*

yum が **インストール** は可能だが **更新** はしない、空白で区切られたパッケージの一覧を表示することができます。デフォルトでインストールのみに設定されているパッケージの一覧については、**yum.conf**(5) の man ページを参照して下さい。

installonlypkgs 指示文を **/etc/yum.conf** ファイルに追加する場合は、**yum.conf**(5) の**installonlypkgs** セクション下に表示されているものも含め、インストールのみである **すべての** パッケージを一覧表示するようにして下さい。特に、カーネルパッケージは常に **installonlypkgs** (デフォルトのとおり) に一覧表示するようにして下さい。また、デフォルトのカーネルがブートに失敗した場合でもバックアップカーネルを常に利用できるように、**installonly_limit** は常に **2** より大きい値に設定することをお勧めします。

installonly_limit=*value*

value は、**installonlypkgs** 指示文に表示されている単一のパッケージに同時にインストール可能なバージョンの最大数を表す整数です。

installonlypkgs 指示文のデフォルトには複数の様々なカーネルパッケージが含まれているため、**installonly_limit** の値を変更すると、単一のカーネルパッケージのインストール済みバージョンの最大数にも影響が及ぶ点に注意して下さい。**/etc/yum.conf** に表示されているデフォルト値は、**installonly_limit=3** です。また、この値を低く、特に **2** より下に設定することは推奨されません。

keepcache=*value*

value は以下のいずれかです。

0 — インストールの成功後、ヘッダーとパッケージのキャッシュを保持しません。これがデフォルトです。

1 — インストールの成功後、キャッシュを保持します。

logfile=*file_name*

file_name は、**yum** がログ出力を書き込むファイルへの絶対パスです。デフォルトでは、**yum** は **/var/log/yum.log** にログを記録します。

multilib_policy=*value*

value は以下のいずれかです。

best — このシステムに最適なアーキテクチャーをインストールします。例えば AMD64 システムに **multilib_policy=best** を設定すると、**yum** は全パッケージの 64-bit バージョンをインストールします。

all — 常に全パッケージ用の可能なあらゆるアーキテクチャーをインストールします。例えば、AMD64 システムで **multilib_policy** を **all** に設定すると、**yum** はパッケージの i586 及び AMD64 が利用可能であれば両バージョンをインストールします。

obsoletes=*value*

value は以下のいずれかです。

0 — 更新の実行時に **yum** の obsoletes 処理ロジックを無効にします。

1 — 更新の実行時に **yum** の obsoletes 処理ロジックを有効にします。あるパッケージがスペックファイル内で別のパッケージを *廃止する* ように宣言している場合、元のパッケージがインストールされた時に他のパッケージは元のパッケージによって置き換えられます。例えば、パッケージ名が変更された場合などに obsoletes は宣言されます。**obsoletes=1** がデフォルトです。

plugins=*value*

value は以下のいずれかです。

0 — Yum のプラグインを全体的に無効にします。

重要

一部のプラグインは、重要な **Yum** サービスを提供するためすべてのプラグインを無効にすることは推奨されません。特に、**rhnplugin** は **RHN Classic** へのサポートを、**product-id** 及び **subscription-manager** のプラグインは証明書ベースの **Content Delivery Network** (CDN) へのサポートをそれぞれ提供します。プラグインを全体的に無効にする便利なオプションもありますが、通常は **Yum** の使用に潜在的な問題があると判断された場合にのみ使用することが推奨されます。

1 — すべての Yum プラグインを全体的に有効にします。**plugins=1** に設定した場合、ある Yum プラグインの設定ファイル内で **enabled=0** を設定することでそのプラグインを無効にすることも可能です。

Yum の各種プラグインの詳細については、「Yum のプラグイン」 を参照して下さい。プラグインの制御に関する詳細は、「Yum プラグインを有効/設定/無効にする方法」 を参照して下さい。

reposdir=*directory*

directory は **.repo** ファイルがあるディレクトリへの絶対パスです。すべての **.repo** ファイルには、リポジトリ情報 (**/etc/yum.conf** の **[*repository*]** セクションと類似) が含まれています。**yum** は **.repo** ファイル及び **/etc/yum.conf** ファイルの **[*repository*]** セクションからすべ

てのリポジトリ情報を収集し、トランザクションに使用するリポジトリのマスターの一覧を作成します。**reposdir** が設定されていない場合は、**yum** はデフォルトのディレクトリである **/etc/yum.repos.d/** を使用します。

retries=*value*

value は、整数 **0** かそれより大きい数です。この値は、エラーを返す前に **yum** がファイルの取得を試行する回数を設定します。これを **0** に設定すると、**yum** はその試行を何度も続けます。デフォルト値は **10** です。

利用可能な **[main]** オプションの全一覧は、**yum.conf**(5) の man ページ の **[main] OPTIONS** セクションを参照して下さい。

6.3.2. [repository] オプションの設定

[*repository*] セクションでは、*repository* が **my_personal_repo** (空白は使用できません) などの一意のリポジトリ ID であるため、個々の Yum リポジトリを定義することができます。

[*repository*] セクションでは、以下の例のような最低限の形式が必要です:

```
[repository]
name=repository_name
baseurl=repository_url
```

すべての **[*repository*]** セクションには、以下の指示文を含む必要があります。

name=*repository_name*

repository_name は、リポジトリを説明する人間が読み取れる文字列です。

baseurl=*repository_url*

repository_url は、リポジトリの repodata ディレクトリがあるディレクトリへの URL です。

- リポジトリが HTTP にある場合は、**http://path/to/repo** を使用して下さい。
- リポジトリが FTP にある場合は、**ftp://path/to/repo** を使用して下さい。
- リポジトリがマシンのローカルにある場合は、**file:///path/to/local/repo** を使用して下さい。
- あるオンラインリポジトリにベーシック HTTP 認証が必要な場合は、***username*:*password*@*link*** として URL にユーザー名とパスワードを先頭に追加して、指定することができます。例えば、http://www.example.com/repo/ にあるリポジトリがユーザー名 「user」 とパスワード 「password」 が必要な場合、**baseurl** のリンクは、**http://user:password@www.example.com/repo/** として指定できます。

通常この URL は以下のような HTTP リンクです:

```
baseurl=http://path/to/repo/releases/$releasever/server/$basearch/os/
```

Yum は常に URL の **$releasever**、**$arch**、**$basearch** 変数を拡張する点に注意して下さい。Yum 変数の詳細については、「Yum 変数の使用」 を参照して下さい。

以下のように便利な **[*repository*]** 指示文もあります。

enabled=*value*

value は以下のいずれかです。

0 — 更新及びインストールの実行時には、パッケージソースとしてこのリポジトリを含めないで下さい。これはリポジトリを迅速に有効/無効にする簡単な方法です。また、更新/インストール用に有効にしたくないリポジトリから単一のパッケージを希望する時に役立ちます。

1 — パッケージソースとしてこのリポジトリを含みます。

リポジトリのオン/オフは、**--enablerepo=*repo_name*** もしくは **--disablerepo=*repo_name*** オプションを **yum** に渡すか、**PackageKit** ユーティリティの **ソフトウェアの追加/削除** ウィンドウから実行できます。

他にも多くの **[*repository*]** オプションがあります。全一覧については、**yum.conf**(5) の man ページの **[repository] OPTIONS** セクションを参照して下さい。

例6.4 /etc/yum.repos.d/redhat.repo ファイルのサンプル

以下は、**/etc/yum.repos.d/redhat.repo** ファイルのサンプルです:

```
#
# Red Hat Repositories
# Managed by (rhsm) subscription-manager
#

[red-hat-enterprise-linux-scalable-file-system-for-rhel-6-entitlement-
rpms]
name = Red Hat Enterprise Linux Scalable File System (for RHEL 6
Entitlement) (RPMs)
baseurl = https://cdn.redhat.com/content/dist/rhel/entitlement-
6/releases/$releasever/$basearch/scalablefilesystem/os
enabled = 1
gpgcheck = 1
gpgkey = file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
sslverify = 1
sslcacert = /etc/rhsm/ca/redhat-uep.pem
sslclientkey = /etc/pki/entitlement/key.pem
sslclientcert = /etc/pki/entitlement/11300387955690106.pem

[red-hat-enterprise-linux-scalable-file-system-for-rhel-6-entitlement-
source-rpms]
name = Red Hat Enterprise Linux Scalable File System (for RHEL 6
Entitlement) (Source RPMs)
baseurl = https://cdn.redhat.com/content/dist/rhel/entitlement-
6/releases/$releasever/$basearch/scalablefilesystem/source/SRPMS
enabled = 0
gpgcheck = 1
gpgkey = file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
sslverify = 1
sslcacert = /etc/rhsm/ca/redhat-uep.pem
sslclientkey = /etc/pki/entitlement/key.pem
sslclientcert = /etc/pki/entitlement/11300387955690106.pem
```

```
[red-hat-enterprise-linux-scalable-file-system-for-rhel-6-entitlement-
debug-rpms]
name = Red Hat Enterprise Linux Scalable File System (for RHEL 6
Entitlement) (Debug RPMs)
baseurl = https://cdn.redhat.com/content/dist/rhel/entitlement-
6/releases/$releasever/$basearch/scalablefilesystem/debug
enabled = 0
gpgcheck = 1
gpgkey = file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
sslverify = 1
sslcacert = /etc/rhsm/ca/redhat-uep.pem
sslclientkey = /etc/pki/entitlement/key.pem
sslclientcert = /etc/pki/entitlement/11300387955690106.pem
```

6.3.3. Yum 変数の使用

yum コマンド及びすべての Yum 設定ファイル内にある以下の組み込み変数を使用/参照することができます (つまり **/etc/yum.conf** 及び **/etc/yum.repos.d/** ディレクトリ内のすべての **.repo** ファイル):

$releasever

この変数を使用すると、Red Hat Enterprise Linux のリリースバージョンを参照することができます。Yum は **/etc/yum.conf** 設定ファイルにある **distroverpkg=*value*** の行より **$releasever** の値を取得します。**/etc/yum.conf** にそのような行がない場合は、**yum** は redhat-release パッケージよりバージョン番号を取得することで、正しい値を導きます。

$arch

この変数を使用して、Python の **os.uname()** 機能を呼び出す時に返り値としてシステムの CPU アーキテクチャーを参照できます。**$arch** の有効な値は、**i586**、**i686**、**x86_64** です。

$basearch

$basearch を使用すると、システムのベースアーキテクチャーを参照できます。例えば、i686 及び i586 両マシンは **i386** のベースアーキテクチャーを持っており、AMD64 及び Intel64 マシンは **x86_64** のベースアーキテクチャーを持っています。

$YUM0-9

これら 10 個の変数は、同じ名前を持つシェル環境変数の値とそれぞれ置換されます。これら変数のいずれかが (例えば **/etc/yum.conf** で) 参照され、同じ名前を持つシェル環境変数が存在しない場合は、設定ファイルの変数は置換されません。

カスタム変数の定義、既存の変数値の上書きを行うには、**/etc/yum/vars/** ディレクトリ内に変数と同じ名前を持つファイルを作成して (「**$**」 記号はなし) 、1 行目に希望する値を追加します。

例えば多くの場合、リポジトリの詳細にはオペレーティングシステムの名前が含まれます。**$osname** と呼ばれる新しい変数を定義するには、1 行目に 「Red Hat Enterprise Linux」 の名前を持つ新しいファイルを作成して、**/etc/yum/vars/osname** として保存します:

```
~]# echo "Red Hat Enterprise Linux" > /etc/yum/vars/osname
```

.repo ファイルでは、「Red Hat Enterprise Linux 6」 の代わりに以下を使用することができます:

```
name=$osname $releasever
```

6.3.4. 現在の設定の表示

Yum グローバルオプションの現在の値 (つまり **/etc/yum.conf** ファイルの **[main]** セクションで指定されたオプション) を表示するには、コマンドラインのオプションなしで **yum-config-manager** を実行します:

```
yum-config-manager
```

別の設定セクションの内容を一覧表示するには、以下の形式でコマンドを実行します:

```
yum-config-manager section…
```

glob 表現を使用して、適合する全セクションの設定を表示させることもできます:

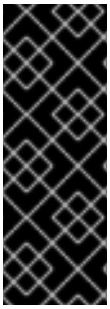
```
yum-config-manager glob_expression…
```

例えば、すべての設定オプションとそれらに対応する値を一覧表示するには、シェルプロンプトで以下を入力して下さい:

```
~]$ yum-config-manager main \*
Loaded plugins: product-id, refresh-packagekit, subscription-manager
================================== main
==================================
[main]
alwaysprompt = True
assumeyes = False
bandwith = 0
bugtracker_url = https://bugzilla.redhat.com/enter_bug.cgi?
product=Red%20Hat%20Enterprise%20Linux%206&component=yum
cache = 0
[出力は省略されています]
```

6.3.5. Yum リポジトリを有効/設定/無効にする方法

「[repository] オプションの設定」 では、Yum リポジトリを定義するために使用できる様々なオプションについて説明しました。本項では、**yum-config-manager** コマンドを使用してリポジトリを追加/有効/無効にする方法を説明します。



重要

システムが証明書ベースの **Red Hat Network** に登録された場合、**/etc/yum.repos.d/redhat.repo** ファイル内のリポジトリを管理するには **Red Hat サブスクリプションマネージャ** ツールを使用します。**Red Hat Network** へのシステムの登録方法、サブスクリプションを管理するための **Red Hat サブスクリプションマネージャ** ツールの使用方法については、5章 *システム登録およびサブスクリプションの管理* を参照して下さい。

Yum リポジトリの追加

新しいリポジトリを定義するには、**[*repository*]** セクションを **/etc/yum.conf** ファイルか、**/etc/yum.repos.d/** ディレクトリ内の **.repo** ファイルに追加します。**yum** は、このディレクトリ内にある **.repo** ファイル拡張子が付いたすべてのファイルを読み取ることができます。**/etc/yum.conf** 内ではなくここにリポジトリを定義することを推奨します。

警告

Red Hat Network 以外の未検証または信頼できないソフトウェアソースからソフトウェアパッケージを取得してインストールする場合には、セキュリティ上のリスクが伴います。セキュリティ、安定性、互換性、保全性に関する問題が発生する恐れがあります。

通常 Yum リポジトリにはそれぞれの **.repo** ファイルがあります。あるリポジトリをシステムに追加して、有効にするには、**root** で以下のコマンドを実行します:

```
yum-config-manager --add-repo repository_url
```

repository_url は **.repo** ファイルへのリンクです。例えば、http://www.example.com/example.repo にあるリポジトリを追加するには、シェルプロンプトで以下を入力します:

```
~]# yum-config-manager --add-repo http://www.example.com/example.repo
Loaded plugins: product-id, refresh-packagekit, subscription-manager
adding repo from: http://www.example.com/example.repo
grabbing file http://www.example.com/example.repo to
/etc/yum.repos.d/example.repo
example.repo                                               |  413 B
00:00
repo saved to /etc/yum.repos.d/example.repo
```

Yum リポジトリの有効化

特定のリポジトリを有効にするには、シェルプロンプトで **root** として以下を入力します:

```
yum-config-manager --enable repository…
```

repository は一意のリポジトリ ID です (利用可能なリポジトリ ID を一覧表示するには **yum repolist all** を使用)。別の方法として、glob 表現を使用すると、一致するすべてのリポジトリを有効にすることができます:

```
yum-config-manager --enable glob_expression…
```

例えば、**[example]**、**[example-debuginfo]**、**[example-source]** セクション内で定義されたリポジトリを有効にするには、以下を入力します。

```
~]# yum-config-manager --enable example\*
Loaded plugins: product-id, refresh-packagekit, subscription-manager
============================== repo: example
==============================
```

```
[example]
bandwidth = 0
base_persistdir = /var/lib/yum/repos/x86_64/6Server
baseurl = http://www.example.com/repo/6Server/x86_64/
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/example
[出力は省略されています]
```

成功すれば、**yum-config-manager --enable** コマンドは現在のリポジトリ設定を表示します。

Yum リポジトリの無効化
Yum リポジトリを無効にするには、**root** で以下のコマンドを実行して下さい:

```
yum-config-manager --disable repository…
```

repository は一意のリポジトリ ID です (利用可能なリポジトリ ID を一覧表示するには **yum repolist all** を使用)。**yum-config-manager --enable** と同様に、glob 表現を使用して、一致するすべてのリポジトリを同時に無効にすることができます:

```
yum-config-manager --disable glob_expression…
```

成功すれば、**yum-config-manager --disable** コマンドは現在の設定を表示します。

6.3.6. Yum リポジトリの作成

次のステップに従って Yum リポジトリを設定します。

1. シェルプロンプトで **root** として以下を入力し、createrepo パッケージをインストールします。

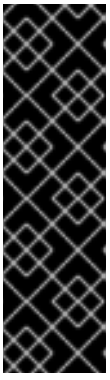
   ```
   yum install createrepo
   ```

2. リポジトリーに配置するパッケージすべてを、**/mnt/local_repo/** などの一つのディレクトリーにコピーします。

3. このディレクトリーに移動して、以下のコマンドを実行します。

   ```
   createrepo --database /mnt/local_repo
   ```

   これにより、Yum リポジトリに必要なメタデータ、さらには **sqlite** データベースが作成されるため **yum** の動作が迅速化します。



重要

Red Hat Enterprise Linux 5 と比較すると、Red Hat Enterprise Linux 6 の RPM パッケージは XZ 無損失データ圧縮形式で圧縮されていて、SHA-256 のような新しいハッシュアルゴリズムを使用して署名されることが可能です。このため、Red Hat Enterprise Linux 5 で **createrepo** コマンドを使用して、Red Hat Enterprise Linux 6 のパッケージメタデータを作成することは推奨されません。それでもこのシステム上で **createrepo** の使用を希望する場合は、EPEL (Extra Packages for Enterprise Linux) から python-hashlib パッケージをインストールして、レポデータも SHA-256 ハッシュアルゴリズムを使用できるようにします。

6.4. YUM のプラグイン

Yum は、動作を拡張し強化するプラグインを提供します。一部のプラグインは、デフォルトでインストールされています。**yum** コマンドを呼び出すといつでも、Yum はどのプラグインがロードされ、アクティブかを伝えます。例えば、以下のとおりです:

```
~]# yum info yum
Loaded plugins: product-id, refresh-packagekit, subscription-manager
[出力は省略されています]
```

Loaded plugins に続くプラグインの名前は **--disableplugins=*plugin_name*** オプションに渡す名前である点に注意して下さい。

6.4.1. Yum プラグインを有効/設定/無効にする方法

Yum プラグインを有効にするには、**plugins=** で始まる行が **/etc/yum.conf** の **[main]** セクションにあり、値が **1** であるようにします。

```
plugins=1
```

すべてのプラグインを無効にするには、この行を **plugins=0** に変更します。

重要

一部のプラグインは、重要な **Yum** サービスを提供するためすべてのプラグインを無効にすることは推奨されません。特に、**rhnplugin** は **RHN Classic** へのサポートを、**product-id** 及び **subscription-manager** のプラグインは証明書ベースの **Content Delivery Network** (CDN) へのサポートをそれぞれ提供します。プラグインを全体的に無効にする便利なオプションもありますが、通常は **Yum** の使用に潜在的な問題があると判断された場合にのみ使用することが推奨されます。

すべてのインストールされたプラグインは、**/etc/yum/pluginconf.d/** ディレクトリにそれぞれの設定ファイルを持っています。これらのファイルでプラグイン固有のオプションを設定できます。例えば、以下のように **refresh-packagekit** プラグインの **refresh-packagekit.conf** 設定ファイルがあるとします:

```
[main]
enabled=1
```

プラグインの設定ファイルには常に **[main]** セクション (Yum の **/etc/yum.conf** ファイルに類似) があります。そこには、 **yum** コマンドの実行時にプラグインが有効かどうかを制御する **enabled=** オプションがあります (ない場合は設定可能)。

/etc/yum.conf で **enabled=0** に設定することですべてのプラグインを無効にする場合は、すべてのプラグインは個々の設定ファイルで有効かどうかに関わらず無効になります。

単一の **yum** コマンドですべての Yum プラグインを無効にしたいだけなら、**--noplugins** オプションを使用します。

単一の **yum** コマンドで 1 つ以上の Yum プラグインを無効にしたい場合は、コマンドに **--disableplugin=*plugin_name*** オプションを追加します。例えば、システムの更新中に **presto** プラグインを無効にするには、以下を入力します:

```
~]# yum update --disableplugin=presto
```

--disableplugin= オプションに渡すプラグインの名前は、**yum** コマンドの出力内にある **Loaded plugins** の行の後に表示されている名前と同じです。複数のプラグインを無効にするには、名前をカンマで区切ります。さらに glob 表現を使用すると、複数のプラグイン名の適合や名前の短縮を行うことができます:

```
~]# yum update --disableplugin=presto,refresh-pack*
```

6.4.2. 追加の Yum プラグインのインストール

通常 Yum プラグインは **yum-plugin-*plugin_name*** パッケージの命名規則に準拠しますが、常にとは限りません。例として **presto** プラグインを提供するパッケージの名前は、**yum-presto** です。Yum プラグインのインストールは、他のパッケージをインストールする場合と同じように実行できます。例えば、**security** プラグインをインストールするには、シェルプロンプトで以下を入力します:

```
~]# yum install yum-plugin-security
```

6.4.3. プラグインの詳細

以下の一覧は、便利な Yum プラグインの詳細です。

fs-snapshot (yum-plugin-fs-snapshot)

fs-snapshot プラグインは、システムの更新やパッケージ削除などのトランザクションを続行する前に、Yum を拡張してファイルシステムのスナップショットを作成します。これにより、トランザクションにより加えられた変更が望ましくない場合は、ユーザーはスナップショットに格納された変更まで遡ることができます。

プラグインが機能するには、root のファイルシステム (**/**) は **LVM** (論理ボリュームマネージャー) か **Btrfs** ボリュームにある必要があります。LVM ボリュームで **fs-snapshot** プラグインを使用するには、以下の手順を実行します。

1. root のファイルシステムがあるボリュームグループに十分な空きエクステントがあることを確認します。必要なサイズは、スナップショットの使用中に予想される元の論理ボリュームへ加わる変更の大きさの関数です。適当なデフォルトは、元の論理ボリュームサイズの 50–80 % です。

   特定のボリュームグループに関する詳しい情報を表示するには、以下の形式で **root** として **vgdisplay** コマンドを実行して下さい:

   ```
   vgdisplay volume_group
   ```

   空きエクステントの数は **Free PE / Size** の行に記載されています。

2. root のファイルシステムがあるボリュームグループに十分な空きエクステントがない場合は、新しい物理ボリュームを追加します。

   a. 以下の形式で **root** で **pvcreate** コマンドを実行して、論理ボリュームマネージャーで使用するために物理ボリュームを初期化します:

      ```
      pvcreate device
      ```

b. 以下の形式で **root** で **vgextend** コマンドを使用して、ボリュームグループに物理ボリュームを追加します:

```
vgextend volume_group physical_volume
```

3. **/etc/yum/pluginconf.d/fs-snapshot.conf** にある設定ファイルを編集して、**[lvm]** セクションに次の変更を加えます。

   a. **enabled** オプションの値を **1** に変更します:

   ```
   enabled = 1
   ```

   b. **lvcreate_size_args** の行の始めにあるハッシュ記号 (#) を削除して、スナップショットに割り振る論理エクステントの数を調整します。例えば、元の論理ボリュームサイズの 80 % を割り振るには、以下のようにします。

   ```
   lvcreate_size_args = -l 80%ORIGIN
   ```

   利用可能な設定オプションの全一覧については、表6.3「サポートされている **fs-snapshot.conf** 指示文」 を参照して下さい。

4. 変更を確認してトランザクションを続行する前に、希望する **yum** コマンドを実行して、**fs-snapshot** がロード済みプラグイン (**Loaded plugins** の行) の一覧に含まれていることを確認して下さい。**fs-snapshot** プラグインは、それぞれ影響を受ける論理ボリュームに対して以下の形式で行を表示します:

```
fs-snapshot: snapshotting file_system
(/dev/volume_group/logical_volume): logical_volume_yum_timestamp
```

5. システムが適切に機能していることを確認します。

   - 変更を維持したい場合は、**root** で **lvremove** コマンドを実行してスナップショットを削除します:

   ```
   lvremove /dev/volume_group/logical_volume_yum_timestamp
   ```

   - 変更を元に戻してファイルシステムをスナップショットに保存した状態に復旧させたい場合は、以下の手順を実行します。

     a. 以下の形式で **root** としてコマンドを実行して、スナップショットを元の論理ボリュームにマージします:

     ```
     lvconvert --merge
     /dev/volume_group/logical_volume_yum_timestamp
     ```

     lvconvert コマンドは、変更を反映させるために再起動が必要であると通知します。

     b. 指示どおりにシステムを再起動します。**root** としてシェルプロンプトで以下を入力します:

     ```
     reboot
     ```

Btrfs ファイルシステムで **fs-snapshot** プラグインを使用するには、以下の手順を実行します。

1. 変更を確認してトランザクションを続行する前に、希望する **yum** コマンドを実行して、**fs-snapshot** がロード済みプラグイン (**Loaded plugins** の行) の一覧に含まれていることを確認して下さい。**fs-snapshot** プラグインは、それぞれ影響を受けるファイルシステムに対して以下の形式で行を表示します:

   ```
   fs-snapshot: snapshotting file_system: file_system/yum_timestamp
   ```

2. システムが適切に機能していることを確認します。

   - 変更を保存する場合は、オプションとして必要ないスナップショットを削除することができます。Btrfs スナップショットを削除するには、**root** で以下の形式でコマンドを実行します:

     ```
     btrfs subvolume delete file_system/yum_timestamp
     ```

   - 変更を元に戻してファイルシステムをスナップショットに保存した状態に復旧させたい場合は、以下の手順を実行します:

     a. **root** として以下のコマンドを使用して、特定のスナップショットの識別子を見つけ出します:

        ```
        btrfs subvolume list file_system
        ```

     b. デフォルト設定でこのスナップショットをマウントするよう **root** としてシステムを設定します:

        ```
        btrfs subvolume set-default id file_system
        ```

     c. システムを再起動します。**root** としてシェルプロンプトで以下を入力します:

        ```
        reboot
        ```

Red Hat Enterprise Linux 6 では、Btrfs がテクノロジープレビューとして含まれているため、このファイルシステムで試すことができます。Btrfs は 64-bit x86 アーキテクチャーでのみ利用可能です。重要なデータが含まれるパーティションや重要なシステム動作に不可欠なパーティションには Btrfs を使用しないで下さい。

論理ボリュームの管理、Btrfs、ファイルシステムのスナップショットの詳細については、『Red Hat Enterprise Linux 6 Storage Administration Guide (ストレージ管理ガイド)』 を参照して下さい。プラグインとその設定の追加情報については、**yum-fs-snapshot**(1) 及び **yum-fs-snapshot.conf**(5) の man ページを参照して下さい。

表6.3 サポートされている `fs-snapshot.conf` 指示文

セクション	指示文	詳細
[main]	**enabled**=*value*	プラグインを有効/無効にできます。*value* は **1** (有効)/**0** (無効) にする必要があります。インストール時には、プラグインはデフォルトで有効です。

セクション	指示文	詳細
	exclude=*list*	特定のファイルシステムを除外できます。値は、スナップショットの実行を **望まない** 空白で区切られたマウントポイントの *list* でなければなりません (例えば **/srv /mnt/backup**)。このオプションはデフォルトでは設定ファイルに含まれていません。
[lvm]	**enabled**=*value*	LVM ボリュームでのプラグインの使用を有効/無効にできます。*value* は **1** (有効)/**0** (無効) にする必要があります。このオプションはデフォルトでは無効です。
	lvcreate_size_args=*value*	論理ボリュームスナップショットのサイズを指定できます。*value* は、**lvcreate** ユーティリティとそれに続く有効な引数に対する **-l** または **-L** コマンドラインのオプションである必要があります (例えば **-l 80%ORIGIN**)。

kabi (kabi-yum-plugins)

kabi プラグインは、ドライバー更新パッケージが公式の Red Hat *kernel Application Binary Interface* (kABI) と適合するかどうかを確認します。このプラグインが有効な状態で、ユーザーがホワイトリストにないカーネルシンボルを使用するパッケージのインストールを試行する場合、警告メッセージがシステムログに書き込まれます。さらには、プラグインを enforcing モードで実行するよう設定すると、そうしたパッケージが決してインストールされないようにできます。

kabi プラグインを設定するには、**/etc/yum/pluginconf.d/kabi.conf** にある設定ファイルを編集して下さい。**[main]** セクションで使用できる指示文の一覧については、表6.4「サポートされている **kabi.conf** 指示文」 を参照して下さい。

表6.4 サポートされている kabi.conf 指示文

指示文	詳細
enabled=*value*	プラグインを有効/無効にできます。*value* は **1** (有効)/**0** (無効) にする必要があります。インストール時には、プラグインはデフォルトで有効です。
whitelists=*directory*	サポートされているカーネルシンボルを持つファイルがある *directory* を指定できます。デフォルトでは、**kabi** プラグインは kabi-whitelists パッケージ (**/lib/modules/kabi/** ディレクトリ) により提供されるファイルを使用します。
enforce=*value*	enforcing モードを有効/無効にできます。*value* は **1** (有効)/**0** (無効) にする必要があります。デフォルトでは、このオプションはコメントアウトされ **kabi** プラグインは警告メッセージのみを表示します。

presto (yum-presto)

presto プラグインは Yum を強化して、**presto** メタデータが有効なリポジトリから更新中に *差分*

RPM パッケージをダウンロードします。差分 RPM に含まれるのは、RPM パッケージを要求するクライアントでインストールされたパッケージのバージョンとリポジトリにある更新済みバージョンの差分のみです。

差分 RPM のダウンロードは、更新済みのパッケージ全体をダウンロードするよりもはるかに迅速で、更新を大幅にスピードアップできます。差分 RPM がダウンロードされたら、現在インストール済みのパッケージに差分を適用して、完全に更新されたパッケージを作成するために RPM をリビルドする必要があります。このプロセスはインストールされたマシンでの CPU の時間を消費します。よって、差分 RPM を使用することは、ネットワーク接続次第であるダウンロードの時間と、CPU バウンドなリビルドの時間とのトレードオフになります。遅いネットワーク接続を使用する高速なマシンとシステムには、**presto** プラグインを使用することが推奨されます。一方で、非常に高速な接続を使用する遅いマシンには、通常の RPM パッケージをダウンロードする、つまり **presto** を無効にすることでより利点があります。

product-id (subscription-manager)

product-id プラグインは、Content Delivery Network (コンテンツ配信ネットワーク) からインストールされた製品の製品識別証明書を管理します。**product-id** プラグインはデフォルトでインストールされています。

protect-packages (yum-plugin-protect-packages)

protect-packages プラグインは、yum パッケージとそれが依存するすべてのパッケージが故意にまたは誤って削除されないようにします。これで、システムを実行するために必要な多くの最重要パッケージが削除されないようにします。加えて、1 行につき 1 つずつパッケージを **/etc/sysconfig/protected-packages** ファイルに一覧表示できます[2] (そのファイルがなければ作成して下さい)。また、**protect-packages** はそうしたパッケージも削除されないようにします。

パッケージの保護を一時的に無効にするには、該当する **yum** コマンドに **--override-protection** オプションを付けて使用します。

refresh-packagekit (PackageKit-yum-plugin)

refresh-packagekit プラグインは、**yum** を実行する時はいつでも **PackageKit** のメタデータを更新します。**refresh-packagekit** プラグインはデフォルトでインストールされています。

rhnplugin (yum-rhn-plugin)

rhnplugin を使用すると **RHN Classic** へ接続できます。これにより、**RHN Classic** に登録しているシステムはこのシステムからパッケージの更新/インストールを行うことができます。Red Hat Enterprise Linux 6 への移行のため、**RHN Classic** が備わっているのは 旧 Red Hat Enterprise Linux システム (Red Hat Enterprise Linux 4.x、Red Hat Enterprise Linux 5.x 及び Satellite 5.x) のみである点に注意して下さい。**rhnplugin** はデフォルトでインストールされています。

プラグインの詳細については、**rhnplugin**(8) の man ページを参照してください。

security (yum-plugin-security)

セキュリティ更新に関する情報を見つけ出して、簡単かつ頻繁に適用することは、どのシステム管理者にとっても重要です。このため、Yum は **security** プラグインを提供します。このプラグインは、非常に有用なセキュリティ関連のコマンド、サブコマンド及びオプションのセットにより **yum** を拡張します。

以下のように、セキュリティ関連の更新を確認できます:

```
~]# yum check-update --security
```

```
Loaded plugins: product-id, refresh-packagekit, security, subscription-
manager
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
Limiting package lists to security relevant ones
Needed 3 of 7 packages, for security
elinks.x86_64                   0.12-0.13.el6                rhel
kernel.x86_64                   2.6.30.8-64.el6              rhel
kernel-headers.x86_64           2.6.30.8-64.el6              rhel
```

yum update --security または **yum update-minimal --security** を使用すると、セキュリティアドバイザリーの影響を受けるパッケージを更新することができます。どちらのコマンドとも、セキュリティアドバイザリーが発行されたシステム上の全パッケージを更新します。**yum update-minimal --security** は、それらをセキュリティアドバイザリーの一部としてリリースされた最新パッケージに更新します。一方で、**yum update --security** はセキュリティアドバイザリーの影響を受ける全パッケージを **利用可能なパッケージの最新版に** 更新します。

言い換えると以下のようになります。

- kernel-2.6.30.8-16 パッケージがシステムにインストールされており、
- kernel-2.6.30.8-32 パッケージがセキュリティ更新としてリリースされ、
- その後、kernel-2.6.30.8-64 がバグ修正の更新としてリリースされた場合は、

yum update-minimal --security によって kernel-2.6.30.8-32 に更新され、**yum update --security** により kernel-2.6.30.8-64 に更新されます。慎重なシステム管理者であれば、**update-minimal** を使用して、パッケージを更新することで生じるリスクをできるだけ減らすとよいでしょう。

security プラグインが **yum** に追加する拡張機能の詳しい使用方法と説明については、**yum-security**(8) の man ページを参照して下さい。

subscription-manager (subscription-manager)

subscription-manager プラグインを使用すると、**Red Hat Network** へ接続できます。これで **Red Hat Network** に登録されたシステムは、証明書ベースのコンテンツ配信ネットワークからパッケージの更新/インストールを行うことができます。**subscription-manager** プラグインは、デフォルトでインストールされています。

製品のサブスクリプションとエンタイトルメントの管理方法については、5章 *システム登録およびサブスクリプションの管理* を参照して下さい。

yum-downloadonly (yum-plugin-downloadonly)

yum-downloadonly プラグインは、Red Hat Network または設定した Yum リポジトリからパッケージをインストールせずにダウンロードするのために使用する **--downloadonly** コマンドラインオプションを提供します。

パッケージをインストールするには、「追加の Yum プラグインのインストール」 の手順に従います。インストール後は、プラグインが有効になっていることを **/etc/yum/pluginconf.d/downloadonly.conf** ファイルの内容で確認してください。

```
~]$ cat /etc/yum/pluginconf.d/downloadonly.conf
[main]
enabled=1
```

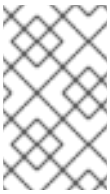
以下の例では、**yum install --downloadonly** コマンドを実行して httpd パッケージの最新バージョンをインストールせずにダウンロードします。

```
~]# yum install httpd --downloadonly
Loaded plugins: downloadonly, product-id, refresh-packagekit, rhnplugin,
              : subscription-manager
Updating Red Hat repositories.
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package httpd.x86_64 0:2.2.15-9.el6_1.2 will be updated
---> Package httpd.x86_64 0:2.2.15-15.el6_2.1 will be an update
--> Processing Dependency: httpd-tools = 2.2.15-15.el6_2.1 for package:
httpd-2.2.15-15.el6_2.1.x86_64
--> Running transaction check
---> Package httpd-tools.x86_64 0:2.2.15-9.el6_1.2 will be updated
---> Package httpd-tools.x86_64 0:2.2.15-15.el6_2.1 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
========
 Package        Arch      Version                 Repository
Size
================================================================================
========
Updating:
 httpd          x86_64    2.2.15-15.el6_2.1       rhel-x86_64-server-6
812 k
Updating for dependencies:
 httpd-tools    x86_64    2.2.15-15.el6_2.1       rhel-x86_64-server-6
70 k

Transaction Summary
================================================================================
========
Upgrade       2 Package(s)

Total download size: 882 k
Is this ok [y/N]: y
Downloading Packages:
(1/2): httpd-2.2.15-15.el6_2.1.x86_64.rpm               | 812 kB
00:00
(2/2): httpd-tools-2.2.15-15.el6_2.1.x86_64.rpm         |  70 kB
00:00
--------------------------------------------------------------------------------
------------
Total                                           301 kB/s | 882 kB
00:02


exiting because --downloadonly specified
```

デフォルトでは、**--downloadonly** オプションを使用してダウンロードされたパッケージは、Red Hat Enterprise Linux のバリアントやアーキテクチャーに応じて **/var/cache/yum** ディレクトリのサブディレクトリに保存されます。

パッケージを保存する別のディレクトリを指定したい場合は、**--downloadonly** とともに **--downloaddir** オプションを渡します。

```
~]# yum install --donwloadonly --downloaddir=/path/to/direcotry httpd
```



注記

yum-downloadonly プラグインに代わるものとして、パッケージをインストールせずにダウンロードするために yum-utils パッケージにより提供される **yumdownloader** ユーティリティを使用することができます。

6.5. その他のリソース

http://yum.baseurl.org/wiki/Guides

Yum wiki の **Yum Guides** セクションには多くのドキュメントが記載されています。

[2] **/etc/sysconfig/protected-packages.d/** ディレクトリに **.list** の拡張子が付いたファイルを置くことも可能です (そのディレクトリがなければ作成して下さい)。また、これらのファイルに 1 行に 1 つずつパッケージを一覧表示することもできます。**protect-packages** はこれらも保護します。

第7章 PACKAGEKIT

Red Hat は、システムと互換性があるパッケージを表示、管理、更新、インストール、アンインストールする **PackageKit** を提供しています。PackageKit は数種類のグラフィカルインターフェースで構成されおり、これは GNOME パネルメニューまたは利用可能な更新を PackageKit が警告する際に表示される通知スペースから開くことができます。**PackageKit** のアーキテクチャと利用可能なフロントエンドの詳細は 「PackageKit アーキテクチャ」 を参照してください。

7.1. ソフトウェア更新を使ったパッケージ更新

PackageKit は、システムにインストールできる更新があるといつでも通知スペースに星形のアイコンを表示します。

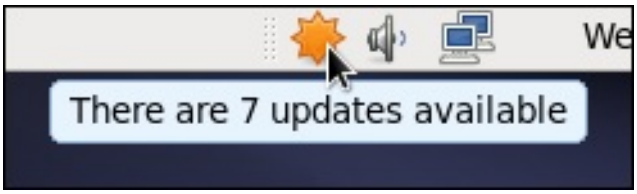


図7.1 通知スペースにある PackageKit のアイコン

通知アイコンをクリックすると、**Software Update (ソフトウェア更新)** ウィンドウが開きます。別の方法として、GNOME パネルから **システム** → **管理** → **ソフトウェア更新** の順にクリックするか、シェルプロンプトで **gpk-update-viewer** コマンドを実行しても同様に **ソフトウェア更新** を開くことができます。**ソフトウェア更新** ウィンドウでは、更新されるパッケージの名前 (**.rpm** の接尾辞なし、ただし CPU アーキテクチャは含む) 、パッケージの簡単なサマリ、及び通常は更新に伴う変更の簡単な説明と併せて、すべての利用可能な更新が表示されています。インストールしたくない更新がある場合は、該当する更新のチェックボックスからチェックマークを外してください。

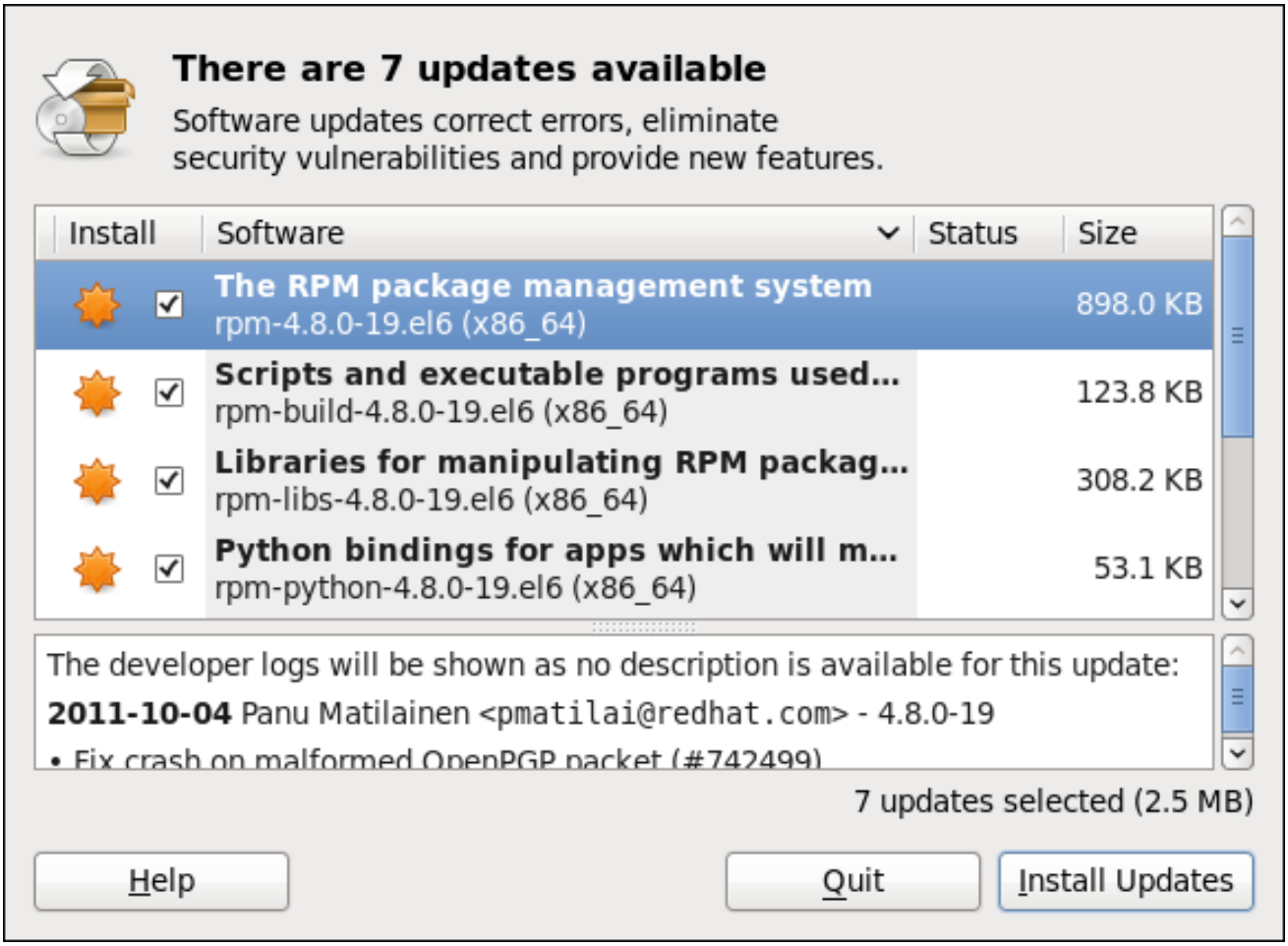


図7.2 ソフトウェア更新を使用した更新のインストール

ソフトウェア更新 ウィンドウに提示された更新は、システムに現在インストールされているパッケージの内、更新が利用できるものだけを示しています。これらのパッケージの依存関係、及びシステム上のパッケージが既存のものか、あるいは新規のものかは **Install Updates (更新のインストール)** をクリックするまで表示されません。

ユーザーが PackageKit に対してシステムへの変更を要求すると、 PackageKit は **PolicyKit** ツールキットが提供するきめの細かいユーザー認証機能を活用します。PackageKit にパッケージの更新、インストール、あるいは削除を指示すると、システムに変更を加える前にスーパーユーザーのパスワード入力を要求されます。

PackageKit に **kernel** パッケージの更新を指示すると、インストール後にシステムを再起動して新しくインストールされたカーネルでのブートを希望するか確認します。

更新の確認頻度の設定

PackageKit の通知スペースのアイコンを右クリックして、**詳細設定** をクリックすると、**ソフトウェア更新の詳細設定** ウィンドウが開きます。そこでは、PackageKit のパッケージ更新の確認頻度や自動的にインストールするのすべての更新か、またはセキュリティの更新だけかも定義することができます。**モバイルブロードバンド使用時に更新を確認する** のボックスのチェックマークを外すと、ダウンロードしたデータ量で課金されるワイヤレス接続を使用している時に無関係な帯域幅の使用を回避できるので便利です。

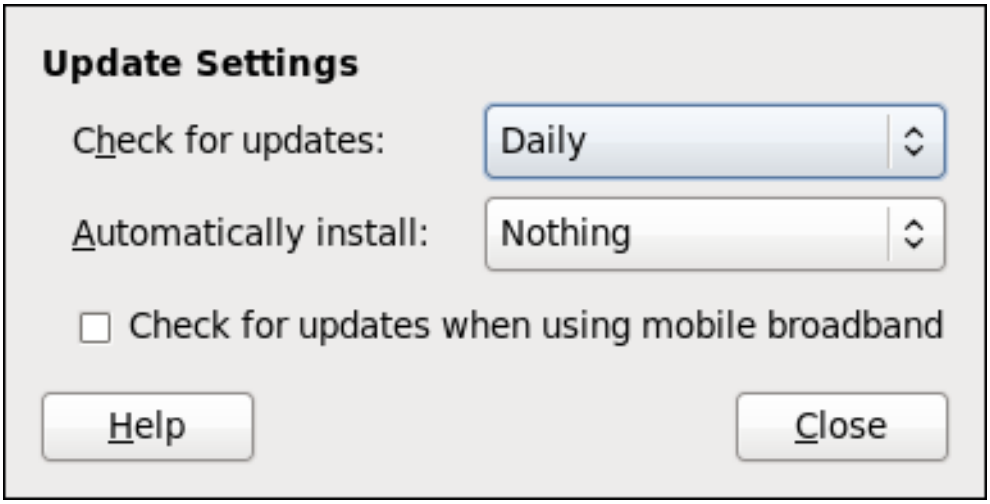


図7.3 PackageKit の更新の確認頻度の設定

7.2. ソフトウェアの追加/削除の使用

新規パッケージを検索してインストールするには、GNOME パネルで **システム** → **管理** → **ソフトウェアの追加/削除** の順にクリックするか、シェルプロンプトで **gpk-application** コマンドを実行します。

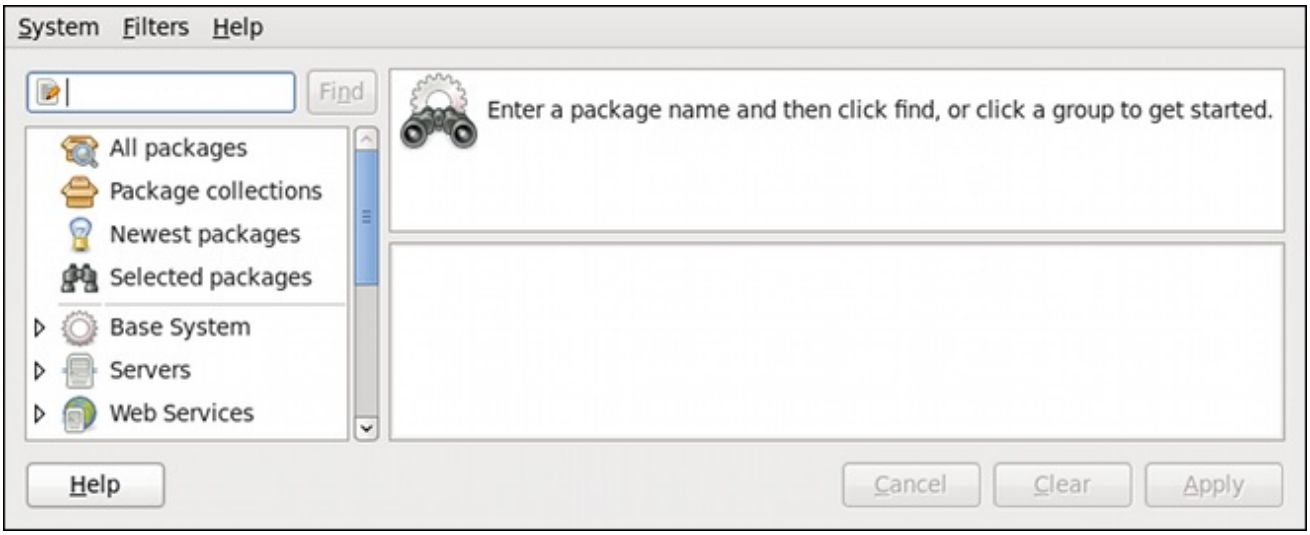


図7.4 PackageKit のソフトウェアの追加/削除ウィンドウ

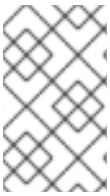
7.2.1. ソフトウェアソースを最新の情報への更新 (yum リポジトリ)

PackageKit は、**Yum** リポジトリをソフトウェアソースとして参照します。ここには、有効なソフトウェアソースからのパッケージすべてが含まれています。**設定済み** でフィルターされていない (以下を参照) **Yum** リポジトリの一覧をすべて表示するには、**ソフトウェアの追加/削除** を開いて、**システム** → **ソフトウェアソース** をクリックします。**ソフトウェアソース** ダイアログは、**/etc/yum.conf** 設定ファイル内の [*repository*] セクションすべての **name=<*My Repository Name*>** フィールドと、**/etc/yum.repos.d/** ディレクトリ内のすべての ***repository*.repo** ファイルに記述されているリポジトリー名を表示します。

有効 のコラムにチェックマークが入ったエントリは、更新およびインストールのリクエスト (依存関係の解決を含む) を満たすパッケージの特定に対応するリポジトリが使用されることを示しています。このチェックボックスの選択/非選択で、一覧の Yum リポジトリの有効化/無効化ができます。これを実行すると、**PolicyKit** がスーパーユーザーの認証を要求することに注意してください。

有効 のコラムは、[*repository*] セクションの **enabled=<*1 or 0*>** フィールドに該当します。チェックマークを選択すると、正しい [*repository*] セクションに **enabled=<*1 or 0*>** 行がない場合は PackageKit がこれを挿入し、ある場合は値を変更します。つまり、**ソフトウェアソース** ウィンドウでリポジトリを有効/無効にすると、ウィンドウを閉じた後やシステムを再起動した後でもその変更は保たれます。

PackageKit では yum リポジトリを追加または削除することはできない点に注意してください。



注記

ソフトウェアソース ウィンドウの最下部のボックスにチェックマークを入れると、PackageKit はソース RPM、テスト、debuginfo のリポジトリも表示します。このボックスはデフォルトではチェックマークは入っていません。

利用可能な Yum リポジトリに変更を加えた後、**システム** → **パッケージ一覧をリフレッシュ** をクリックして、パッケージ一覧が最新であることを確認します。

7.2.2. フィルターを使ったパッケージ検索

ソフトウェアソースの更新後は、フィルタを使用するとPackageKit がより速く **検索** クエリの結果を取得できるようなり、役立つ場合が多くあります。これは多くのパッケージ検索を行っている際に特に役立ちます。**フィルター** のドロップダウンメニューにある 4 つのフィルターは、1 つの基準の一致/不一致で結果を絞り込みます。PackageKit の起動時にはデフォルトでどのフィルターも適用されていませんが (**フィルタなし**)、どれか 1 つでフィルタを使用すると、そのフィルタは変更するか PackageKit を閉じるまで適用されたままになります。

通常はシステム上にインストールされて**いない** 利用可能なパッケージを検索するので、**フィルタ** → **インストール済** とクリックして、**利用可能のみ** ラジオボタンを選択します。

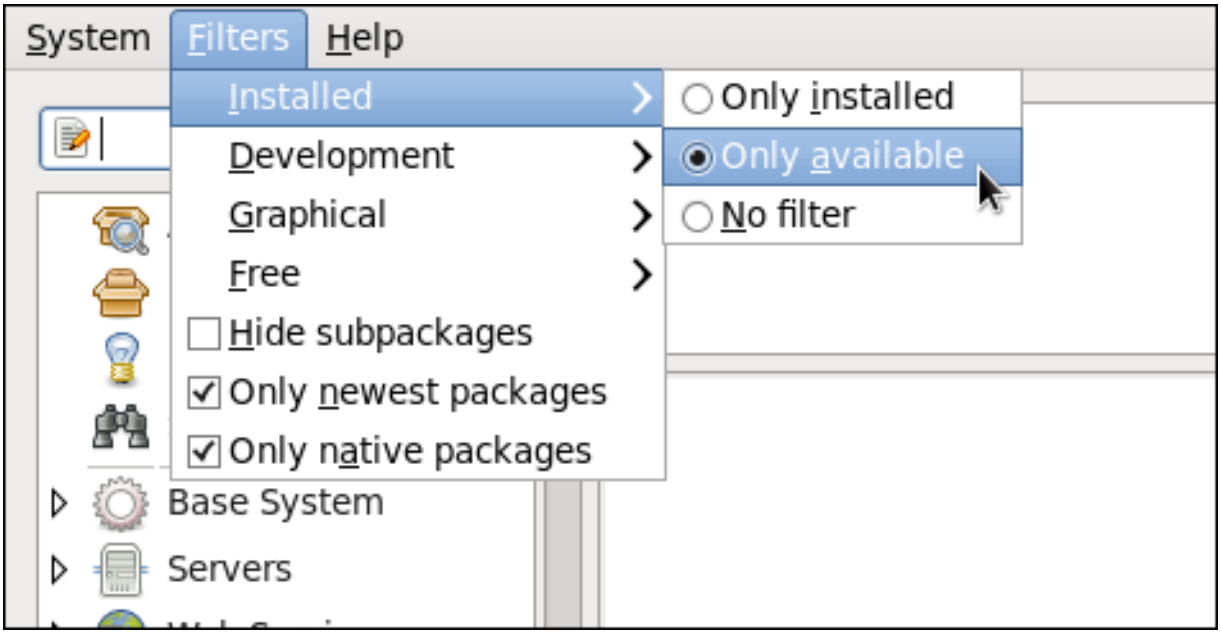


図7.5 すでにインストール済みのパッケージをフィルターにかけて除外する

また、C ヘッダーファイルなどの開発ファイル必要でなければ、**フィルタ** → **開発** をクリックして、**エンドユーザーファイルのみ** のラジオボタンを選択します。これにより、関係のない **<*package_name*>-devel** パッケージすべてを除外できます。

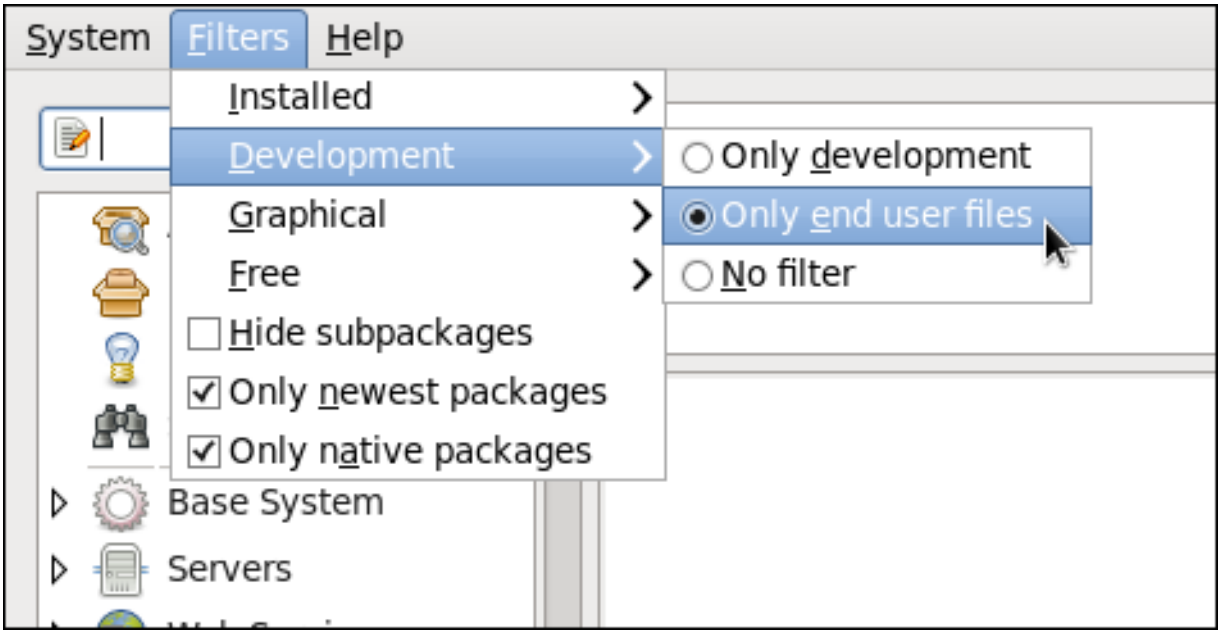


図7.6 検索結果の一覧から開発パッケージをフィルターにかけて除外する

サブメニューにある残りの 2 つのフィルターは、以下のとおりです。

グラフィカル

GUI インターフェースを提供するアプリケーション (**グラフィカルのみ**) か、提供しないアプリケーションかのどちらかに検索を絞り込みます。このフィルターは、特定の機能を実行する GUI アプリケーションを閲覧したい場合に役立ちます。

未使用

フリーソフトウェアと考慮されているパッケージの検索をします。認定ライセンスの詳細については、Fedora Licensing List を参照して下さい。

これらのフィルターは、チェックボックスを選択することで有効にできます。

サブパッケージを隠す

サブパッケージを隠す チェックボックスにチェックを入れると、必要となる他のパッケージに依存している典型的な関心の無いパッケージをフィルターで除外します。例えば、**Hide subpackages（サブパッケージを隠す)** にチェックを入れて、***<package>*** を検索すると、以下のような関連したパッケージが **Find（検索)** 結果からフィルターで除外されます (存在する場合) :

- ***<package>*-devel**
- ***<package>*-libs**
- ***<package>*-libs-devel**
- ***<package>*-debuginfo**

最新パッケージのみ

最新パッケージのみ にチェックマークを入れると、結果一覧から同一パッケージの以前のバージョンすべてが除外され、これが通常望まれる結果です。多くの場合、このフィルターは **利用可能のみ** フィルターと併用され、新しい (インストール済みでない) パッケージの利用可能な最新バージョンの検索に使われます。

ネイティブパッケージのみ

multilib システム上で **ネイティブパッケージのみ** のボックスにチェックマークを入れると、**PackageKit** は **互換モード** で実行するアーキテクチャ用にコンパイルされたパッケージの結果一覧を除外します。例えば、 AMD64 CPU 搭載の 64 ビットシステムでこのフィルターを有効にすると、AMD64 マシン上で実行できるものでも、32 ビットの x86 CPU アーキテクチャ用にビルドされたパッケージはすべて、結果一覧には表示されません。アーキテクチャー不問であるパッケージ (**`crontabs-1.10-32.1.el6.noarch.rpm`** などの *noarch* パッケージ) は、**ネイティブパッケージのみ** にチェックマークを入れてもフィルターで除外されることはありません。このフィルターは、x86 マシンのような multilib でないシステムには影響はありません。

7.2.3. パッケージ (及び依存関係) のインストールと削除

利用可能のみ と **エンドユーザーファイルのみ** の 2 つのフィルターを選択した状態で、コマンドラインの **screen** ウィンドウマネジャーを検索して、パッケージをハイライト表示にします。これでパッケージに関する便利な情報にアクセスできる状態になりました。例えば、プロジェクトホームページへのクリック可能なリンク、そのパッケージがある Yum パッケージグループ (ある場合)、パッケージのライセンス、該当する場合はアプリケーションを開く GNOME メニューの場所へのポインター、パッケージをダウンロードしてインストールする際に関係してくるパッケージサイズ、などです。

図7.7 PackageKit のソフトウェアの追加/削除ウィンドウでのパッケージの表示とインストール

パッケージやグループの隣にあるボックスにチェックマークが入っていれば、その項目はシステムにすでにインストール済みであることを示します。チェックが入っていないボックスにチェックマークを入れると、インストールするために **指定する** ことになり、**適用** ボタンをクリックした時にのみ実行されます。このように、実際のインストール処理が実行される前に複数のパッケージまたはパッケージグループを検索して、選択することができます。さらに、インストール済みパッケージのチェックマークを外すとそれらを削除することもでき、削除は **適用** を押した時に保留中のインストールとともに実施されます。依存関係の解決は、インストールまたは削除するパッケージを追加する場合があり、**適用** を押すと実施されます。PackageKit は、インストールまたは削除されるそれらの追加パッケージをウィンドウに一覧表示し、続行の確認を求めます。

screen を選択し、**適用** ボタンをクリックします。すると、スーパーユーザーのパスワードを求められ、これを入力すると PackageKit が **screen** をインストールします。インストール完了後に、PackageKit は新規にインストールされたアプリケーションの一覧を提示してそれらをすぐに実行する選択を提供することがあります。また、**ソフトウェアの追加/削除** ウィンドウでパッケージを検索して選択すると、GNOME メニューでそのアプリケーションショートカットがある **場所** が表示され、パッケージ実行の際に役立ちます。

インストールが終了すると、シェルプロンプトで **screen** を入力すると、 1 つのターミナルで複数のログインができるようにするスクリーンマネージャである **screen** を実行できます。

screen は便利なユーティリティですが、必要ないと判断してアンインストールするとします。これをインストールする際に使った **利用可能のみ** フィルターを、**フィルタ → インストール済** で **インストール済のみ** に変更し、**screen** を再度検索して、チェックマークを外します。プログラムはそれ自体の依存関係はインストールしていません。依存関係があったとしても、システムにまだインストールされている他のパッケージの依存関係でない限り、それらも自動的に削除されます。

警告

PackageKit はパッケージのインストールおよび削除の際に自動的に依存関係を解決しますが、該当パッケージに依存する他のパッケージを削除しないとこのパッケージの削除はできません。こうしたタイプの操作は **RPM** でのみ実行可能で、推奨されるものではありません。システムが機能しなくなるか、アプリケーションが誤作動/クラッシュする恐れがあります。

図7.8 PackageKit のソフトウェアの追加/削除ウィンドウでのパッケージの削除

7.2.4. パッケージグループのインストールと削除

PackageKit は、Yum パッケージグループのインストールもでき、これを **パッケージのコレクション** と呼びます。**ソフトウェア更新** ウィンドウ内のカテゴリの左上にある **パッケージのコレクション** をクリックすると、スクロールして、インストールしたいパッケージグループを見つけることができます。ここでは、チェコ語のサポート (**Czech Support (チェコ語のサポート)** のグループ) をインストールしたいとします。ボックスにチェックマークを入れて、**適用** をクリックすると、パッケージグループの依存関係を満たすためにインストールが必要な **追加** パッケージの数がわかります。

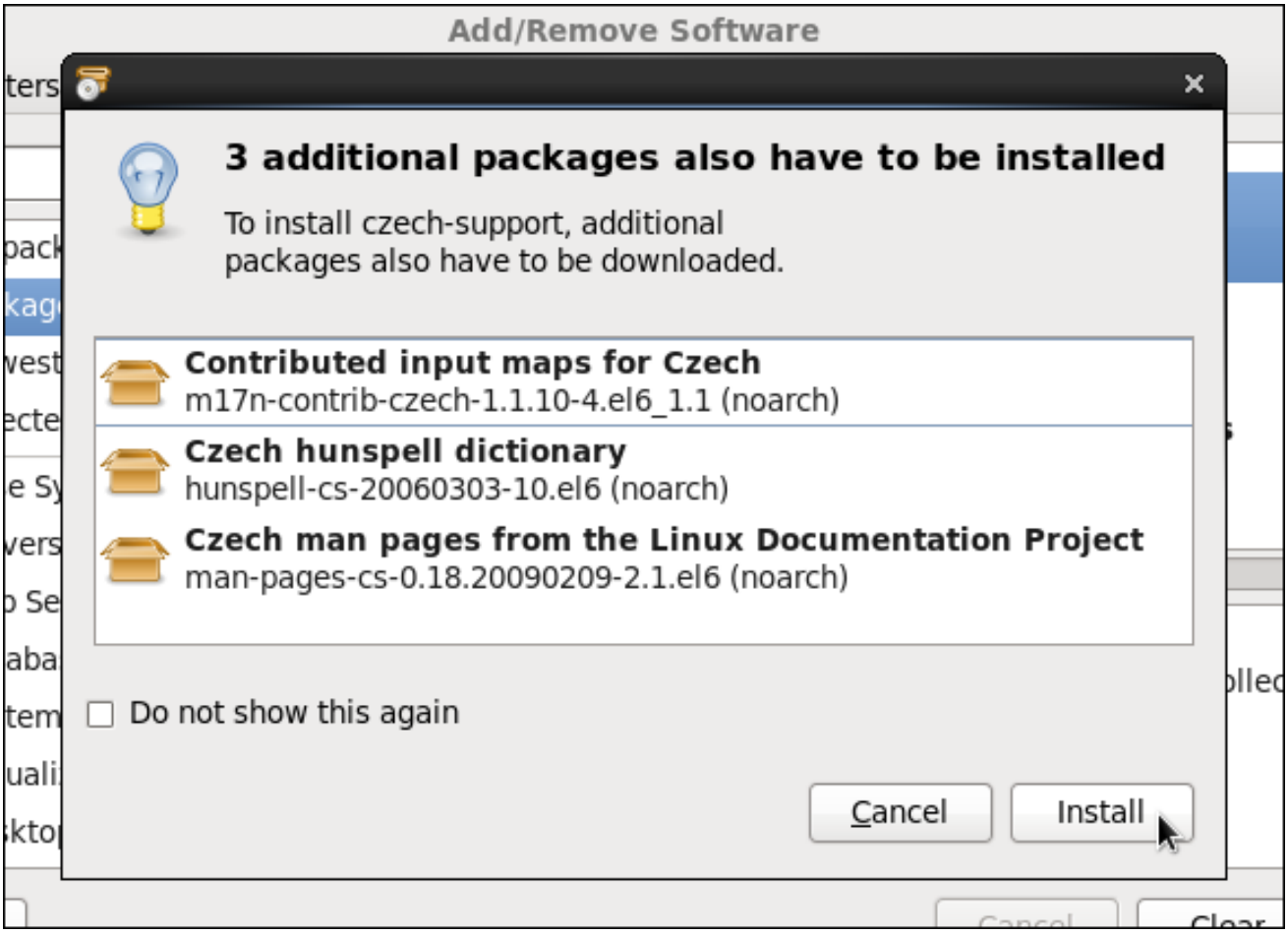


図7.9 チェコ語のサポートのパッケージグループのインストール

同様に、インストール済みのパッケージグループをアンインストールするには、**パッケージのコレクション** を選択して、適切なボックスのチェックマークを外して適用します。

7.2.5. トランザクションログの表示

PackageKit は、実行するトランザクションのログを維持します。ログを表示するには、**ソフトウェアの追加/削除** ウィンドウから **システム** → **ソフトウェアログ** をクリックするか、シェルプロンプトで **gpk-log** コマンドを実行します。

ソフトウェアログビューア は以下の情報を表示します。

- **日付** — トランザクションが実行された日付。
- **アクション** — トランザクション中に実行されたアクション。例としては、**Updated packages (パッケージの更新)** や **Installed packages (パッケージのインストール)** 。
- **詳細** — **Updated (更新)** や **Installed (インストール)** 、**Removed (削除)** などのトランザクションの種類の後に、対象となるパッケージ一覧を表示。
- **ユーザー名** — アクションを実行したユーザーの名前。
- **アプリケーション** — アクションの実行に使用されたフロントエンドアプリケーション。例えば、**Update System (システム更新)** 。

最上部のテキスト入力欄にパッケージ名を入力すると、トランザクションの一覧がフィルターされ、パッケージに影響があったトランザクションログを表示します。

図7.10 ソフトウェアログビューアを使ったパッケージ管理のトランザクションログの表示

7.3. PACKAGEKIT アーキテクチャ

Red Hat は、システムと互換性があるパッケージおよびパッケージグループの表示、更新、インストール、アンインストールを行うための PackageKit アプリケーションスイートを提供します。PackageKit のアーキテクチャ は、バックエンドの **packagekitd** デーモンと通信する数種のグラフィカルフロントエンドで構成されています。バックエンドの packagekitd デーモンは、Yum を利用してパッケージのインストールや削除など実際のトランザクションを実行するパッケージマネージャ固有のバックエンドと通信します。

表7.1「PackageKit の GUI ウィンドウ、メニューの場所、シェルプロンプトのコマンド」 では、GUI ウィンドウ名、GNOME デスクトップまたは **ソフトウェアの追加/削除** ウィンドウから GUI ウィンドウを起動する方法、 GUI ウィンドウを開くコマンドラインのアプリケーション名を示しています。

表7.1 PackageKit の GUI ウィンドウ、メニューの場所、シェルプロンプトのコマンド

ウィンドウのタイトル	機能	開き方	シェルコマンド
ソフトウェアの追加/削除	パッケージ情報のインストール、削除、表示	GNOME パネルより、**システム** → **管理** → **ソフトウェアの追加/削除** を選択	gpk-application
ソフトウェア更新	パッケージの更新を実行	GNOME パネルより、**システム** → **管理** → **ソフトウェア更新** を選択	gpk-update-viewer
ソフトウェアソース	Yum リポジトリーの有効/無効化	**ソフトウェアの追加/削除** より、**システム** → **ソフトウェアソース** を選択	gpk-repo
ソフトウェアログビューア	トランザクションログの表示	**ソフトウェアの追加/削除** より、**システム** → **ソフトウェアログ** を選択	gpk-log

ウィンドウのタイトル	機能	開き方	シェルコマンド
ソフトウェア更新の詳細設定	PackageKit の詳細設定		gpk-prefs
(通知スペースでの警告)	更新が利用可能時に通知	GNOME パネルより、**システム → 詳細設定 → Startup Applications (自動起動するアプリ)**, **Startup Programs (自動起動するプログラム)** タブを選択	gpk-update-icon

packagekitd デーモンは、ユーザーセッションの外で実行し、様々なグラフィカルなフロントエンドと通信します。**packagekitd** デーモンは、[3] **DBus** システムメッセージバスを経由して、別のバックエンドと通信します。これは、Yum の Python API を利用してクエリを実行し、システムに変更を加えます。Red Hat と Fedora 以外の Linux システムでは、**packagekitd** はそのシステムのネイティブのパッケージマネージャーの利用が可能な他のバックエンドと通信することができます。このモジュラーアーキテクチャは、グラフィカルインターフェースが多くの異なるパッケージマネージャーと共に動作して本質的に同じ種類のパッケージ管理タスクを実行するために必要な抽象化を提供します。PackageKit のフロントエンドの使用方法を学ぶと、Yum 以外のネイティブのパッケージマネージャを利用している時でも、多くの様々な Linux ディストリビューションにわたって、なじみのある同じグラフィカルインターフェースを使用することができます。

さらに、**PackageKit** の関与事項の分離により、GUI ウィンドウの 1 つに クラッシュが生じても (またはユーザーの X Window セッションでも) 、ユーザーセッションの外で実行する **packagekitd** デーモンが監視しているパッケージ管理タスクに影響を与えない、という信頼性を提供します。

本章で説明されているフロントエンドのグラフィカルアプリケーションはすべて、PackageKit とその依存関係ではなく、**gnome-packagekit** パッケージが提供します。

最後に、PackageKit には **pkcon** と呼ばれるコンソールベースのフロントエンドも付いてきます。

7.4. その他のリソース

PackageKit のホームページ — http://www.packagekit.org/index.html

PackageKit の情報とメーリングリストです。

PackageKit のよくある質問 (FAQ) — http://www.packagekit.org/pk-faq.html

PackageKit ソフトウェアスイートに関するよくある質問の一覧です。

PackageKit 機能マトリクス — http://www.packagekit.org/pk-matrix.html

PackageKit が提供する機能とパッケージマネージャのバックエンド一覧との相互参照表。

[3] システムデーモンは通常、ユーザーや他のプログラムにサービスを提供する、長いプロセスです。多くの場合、システムデーモンは特別な初期化スクリプト (**initscripts** と短縮される場合が多い) によりブート時に起動します。デーモンは **service** コマンドに応答し、**chkconfig on** または **chkconfig off** コマンドを使用す

ることで永久にオン/オフにすることができます。それらは、通常 **packagekitd** デーモンのように名前に 「**d**」 を付けることによって認識できます。システムのサービスに関する情報は 10章 *サービスとデーモン* を参照してください。

パート III. ネットワーキング

ここでは、Red Hat Enterprise Linux でのネットワーク設定方法について紹介します。

第8章 NETWORKMANAGER

NetworkManager (ネットワークマネージャ) は、ネットワークデバイスと接続が利用可能な時にそれらをアクティブに維持するよう試行する動的ネットワーク制御及び設定システムです。**NetworkManager** は、コアデーモン、ネットワークステータス情報を提供する GNOME 通知スペースアプレット、及び接続とインターフェースの作成、編集、削除が可能なグラフィカル設定ツールで構成されています。**NetworkManager** を使用すると、イーサネット、ワイヤレス、モバイルブロードバンド (携帯 3G など)、**DSL** や **PPPoE** (Point-to-Point over Ethernet イーサネット経由のポイントツーポイントプロトコル) のようなタイプの接続を設定できます。さらに、 **NetworkManager** により、ネットワークエイリアス、静的ルート、DNS 情報、VPN 接続の他、多くの接続固有のパラメータの設定が可能になります。そして、**NetworkManager** は D-Bus により効果的な API を提供するため、アプリケーションはネットワークの設定と状態をクエリ、制御することができます。

これまでの Red Hat Enterprise Linux のバージョンには、**Network Administration Tool (ネットワーク管理ツール)** が同梱されていました。これは、一般的にはコマンドラインの呼び出しにちなみ **system-config-network** として知られています。Red Hat Enterprise Linux 6 では、以前の **Network Administration Tool** が**NetworkManager** に代わり、ユーザー固有のモバイルブロードバンド設定など強化した機能を備えています。また、Red Hat Enterprise Linux 6 ではインターフェース設定ファイルを編集してネットワークを設定することも可能です。詳細は 9章 *ネットワークインターフェース* を参照してください。

NetworkManager は、デフォルトで Red Hat Enterprise Linux にインストールされている場合があります。確実にインストールするには、root ユーザーで、以下のコマンドを実行します:

```
~]# yum install NetworkManager
```

8.1. NETWORKMANAGER デーモン

NetworkManager デーモンは root 権限で実行し、通常ブート時に起動するように設定されています。root で次のコマンドを入力すると、**NetworkManager** デーモンが実行しているか確認できます:

```
~]# service NetworkManager status
    NetworkManager (pid  1527) is running...
```

service コマンドは、**NetworkManager** サービスが実行していない場合には **NetworkManager is stopped** (NetworkManager は停止しています) と報告します。現在のセッションで起動するには、以下のコマンドを入力します:

```
~]# service NetworkManager start
```

chkconfig コマンドを実行して、システムがブートするたびに **NetworkManager** が確実に起動するようにします:

```
~]# chkconfig NetworkManager on
```

サービスの起動、停止、管理とランレベルの詳細については 10章 *サービスとデーモン* を参照してください。

8.2. NETWORKMANAGER との対話

ユーザーは、**NetworkManager** システムサービスと直接対話は行いません。その代わりに、**NetworkManager** の通知スペースアプレットを使って、ネットワーク設定タスクを実行すること

ができます。このアプレットは複数の状態を表示し、現在使用している接続タイプを視覚的にお知らせします。アプレットのアイコン上にポインターを移動すると、現在の接続状態についてのツールチップ情報が表示されます。

図8.1 NetworkManager アプレットの状態

GNOME パネルに **NetworkManager** アプレットが表示されない場合、**NetworkManager** パッケージがお使いのシステム上にインストールされているとすれば、root ではなく、通常のユーザーで以下のコマンドを実行するとアプレットを起動することができます:

```
~]$ nm-applet &
```

このコマンドを実行すると、通知スペースにアプレットが表示されます。このアプレットがログイン時に毎回確実に実行されるようにするには、**システム** → **設定** → **Startup Applications (自動起動するアプリ)** の順にクリックし、**自動起動するアプリの設定** ウィンドウを開きます。次に、**Startup Programs (自動起動するプログラム)** タブを選択して、**NetworkManager (ネットワークマネージャ)** の横のボックスにチェックマークを入れます。

8.2.1. ネットワークへの接続

アプレットのアイコンを左クリックすると、以下が表示されます:

- 現在接続しているカテゴリ別になったネットワークの一覧 (**Wired (有線)** と **Wireless (無線)**)
- **NetworkManager** が検出した **Available Networks (利用可能なネットワーク)** の全一覧
- 設定済みの VPN (Virtual Private Network 仮想プライベートネットワーク) への接続オプション
- 非表示または新しい無線ネットワークへの接続オプション

ネットワークに接続していれば、その接続名は **有線** や **無線** ネットワーク種類の下で太字表示されます。無線アクセスポイントのように多くのネットワークが利用可能な場合、展開可能な**その他のネットワーク** メニューエントリーが表示されます。

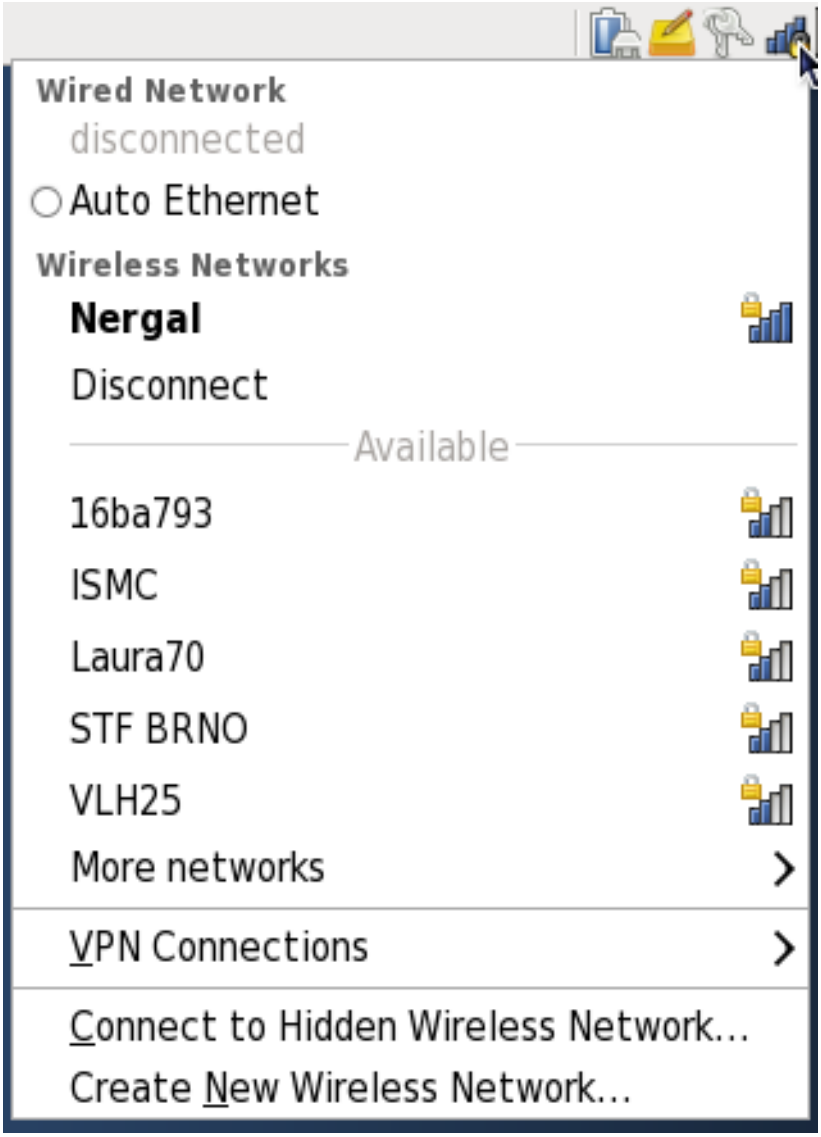


図8.2 NetworkManager アプレットの左クリックメニューで、利用可能及び接続しているネットワークがすべて表示されます。

8.2.2. 新規接続の設定と既存接続の編集

次に、**NetworkManager** アプレットを右クリックして、コンテキストメニューを開きます。これは、**NetworkManager** と対話して接続を設定する主要エントリーポイントです。

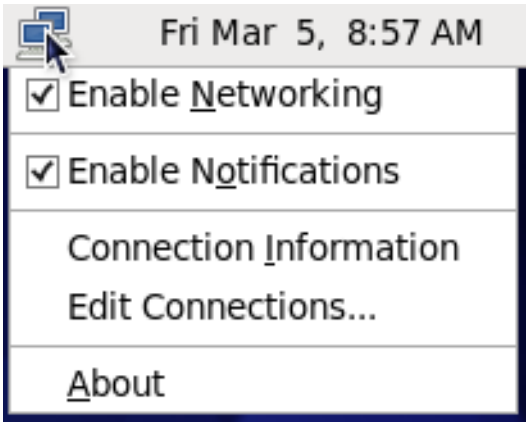


図8.3 NetworkManager アプレットのコンテキストメニュー

ネットワークを有効にする のボックスにチェックマークが付いていることを確認してください。システ

ムがワイヤレスカードを検出すると、**無線を有効にする** のメニューのオプションも表示されます。**無線を有効にする** のチェックボックスにもチェックマークを付けます。**通知を有効にする** のボックスにチェックマークを付けると、**NetworkManager** は、ネットワークの接続状態の変更を通知します。**接続情報** エントリをクリックすると、接続のタイプやインターフェース、IP アドレスとルーティングの詳細などを一覧表示した、詳しい情報を示す **接続情報** ウィンドウが表示されます。

そして最後に、**接続を編集する** をクリックして、**ネットワーク接続** ウィンドウを開きます。ここでは、大半のネットワーク設定タスクを実行することができます。このウィンドウは、通常のユーザーとして以下を実行して開くことができる点にも注目してください。

```
~]$ nm-connection-editor &
```

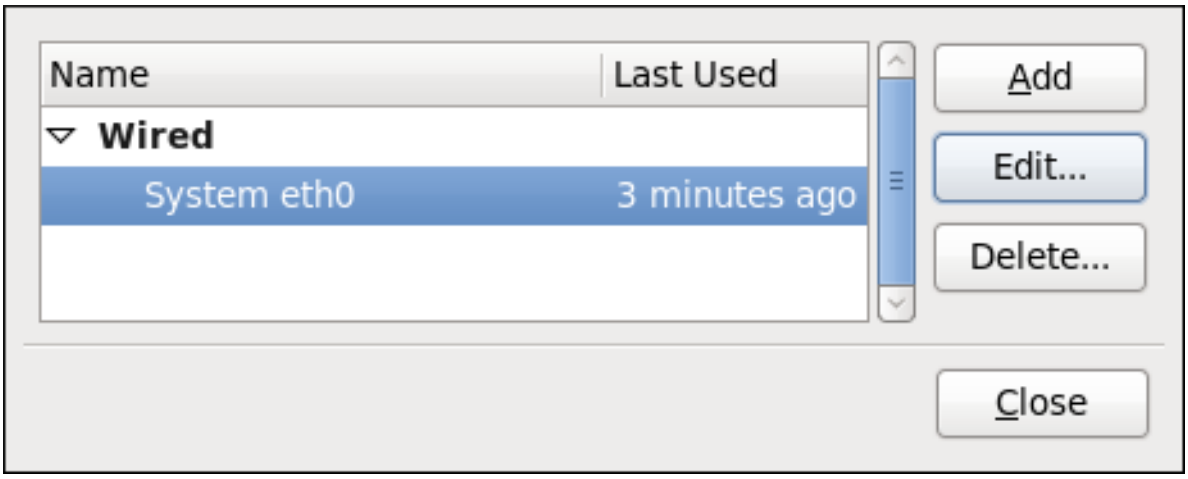


図8.4 ネットワーク接続ウィンドウを使用したネットワークの設定

左側の矢印をクリックすると、必要に応じてエントリーの表示/非表示ができます。新規接続を作成するには、**追加** ボタンをクリックして選択リストを表示し、接続タイプを選択して **適用** ボタンをクリックします。また、既存の接続を編集するには、リストからインターフェース名を選択し、**編集** ボタンをクリックします。

そして、以下のいずれかの設定をします:

- 有線イーサネット接続を設定します。「有線 (イーサネット) 接続の確立」 に進みます。
- 無線接続を設定します。「無線接続の確立」 に進みます。
- モバイルブロードバンド接続を設定します。「モバイルブロードバンド接続の確立」 に進みます。
- VPN 接続を設定します。「VPN 接続の確立」 に進みます。

8.2.3. ネットワークへの自動接続

追加や設定を行うすべての接続に関して、ネットワークが利用可能な時に **NetworkManager** が自動的に接続を試行するかどうか選択することができます。

手順8.1 ネットワークが検出されると、自動接続するよう NetworkManager を設定する

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。
2. 必要に応じて矢印をクリックして、接続リストを表示します。

3. 設定したい接続を選択して、**編集** をクリックします。

4. **Connect automatically (自動接続する)** にチェックマークを入れると、**NetworkManager** は、そのネットワークが利用可能だと検出すると常に自動接続します。**NetworkManager** が自動接続しないようにするには、このチェックボックスからチェックマークを外してください。チェックマークが付いていない場合は、**NetworkManager** アプレットの左クリックメニューで手動で接続を選択して、接続する必要があります。

8.2.4. ユーザー接続とシステム接続

NetworkManager の接続は、常に **ユーザー接続** か **システム接続** のどちらかです。管理者が設定したシステム固有のポリシーによっては、ユーザーがシステム接続の作成や変更を行うのに root 権限が必要となる場合があります。**NetworkManager** のデフォルトのポリシーにより、ユーザーはユーザー接続の作成と変更を行うことができますがシステム接続の追加、変更、削除を行うには root 権限が必要です。

ユーザー接続は、それらを作成するユーザーに固有なためそのように呼ばれます。システム接続では、設定が **/etc/sysconfig/network-scripts/** ディレクトリ (主に **ifcfg-*<network_type>*** インターフェース設定ファイル内) に保管されるのに対して、ユーザー接続の設定は GConf 設定データベースと GNOME キーリングに保管され、ユーザー接続の設定を作成したユーザーのログインセッション中のみに利用可能です。このため、デスクトップセッションをログアウトすると、ユーザー固有の接続は利用できなくなります。

注記

NetworkManager では、GConf 及び GNOME のキーリングアプリケーションを使用して、ユーザー接続設定を保管します。これらの設定はデスクトップセッションに固有であるため、個々の VPN 接続はユーザー接続として設定することが強く推奨されます。その場合、システム上の root 以外のユーザーはこの接続の表示やアクセスは全くできなくなります。

これに対し、システム接続は、ブート時に利用可能となるためデスクトップセッションに最初にログインしなくても、システム上の他のユーザーは使用することができます。

NetworkManager は、ユーザー接続とシステム接続間の切り替えを迅速かつ便利に行うことができます。ユーザー接続をシステム接続に変換すると、**NetworkManager** は **/etc/sysconfig/network-scripts/** ディレクトリの関連したインターフェース設定ファイルを作成し、ユーザーセッションから GConf 設定を削除します。逆に、システム接続をユーザー固有の接続に変換すると、**NetworkManager** はシステム全体の設定ファイルを削除して、対応する GConf/GNOME キーリング設定を作成します。

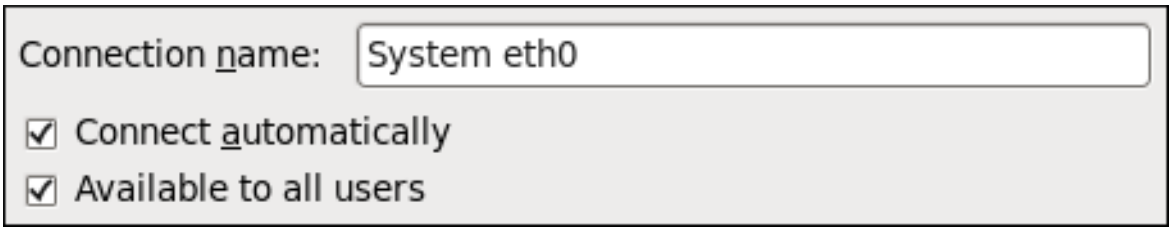


図8.5 Available to all users (すべてのユーザーに利用可能) のチェックボックスで、接続をユーザー固有、またはシステム全体で有効にするかを制御します。

手順8.2 システム全体の接続からユーザー固有の接続への変更、またはその逆

注記

システムのポリシーによっては、接続をユーザー固有またはシステム全体に変更するために、システムで root 権限が必要となる場合があります。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。

2. 必要に応じて、左側の矢印をクリックし、利用可能なネットワーク接続のタイプを表示/非表示させます。

3. 設定したい接続を選択して、**編集** をクリックします。

4. **すべてのユーザーに利用可能** のチェックボックスにチェックマークを付けることで、**NetworkManager** は接続をシステム全体の接続にします。システムのポリシーによっては、**PolicyKit** アプリケーションが root パスワードの入力を求める場合があります。その場合には、root パスワードを入力して、変更を完了します。

   逆に、**すべてのユーザーに利用可能** のチェックボックスからチェックマークを外すと、接続はユーザー固有となります。

8.3. 接続の確立

8.3.1. 有線 (イーサネット) 接続の確立

有線ネットワーク接続を確立するには、**NetworkManager** アプレットを右クリックして、コンテクストメニューを開きます。**ネットワークを有効にする** ボックスにチェックが入っていることを確認して、**接続を編集する** をクリックします。これで **ネットワーク接続** ウィンドウが表示されます。このウィンドウは、通常ユーザーとして以下のコマンドを実行して開くこともできます。

```
~]$ nm-connection-editor &
```

必要に応じて矢印をクリックして、接続リストを表示/非表示させます。

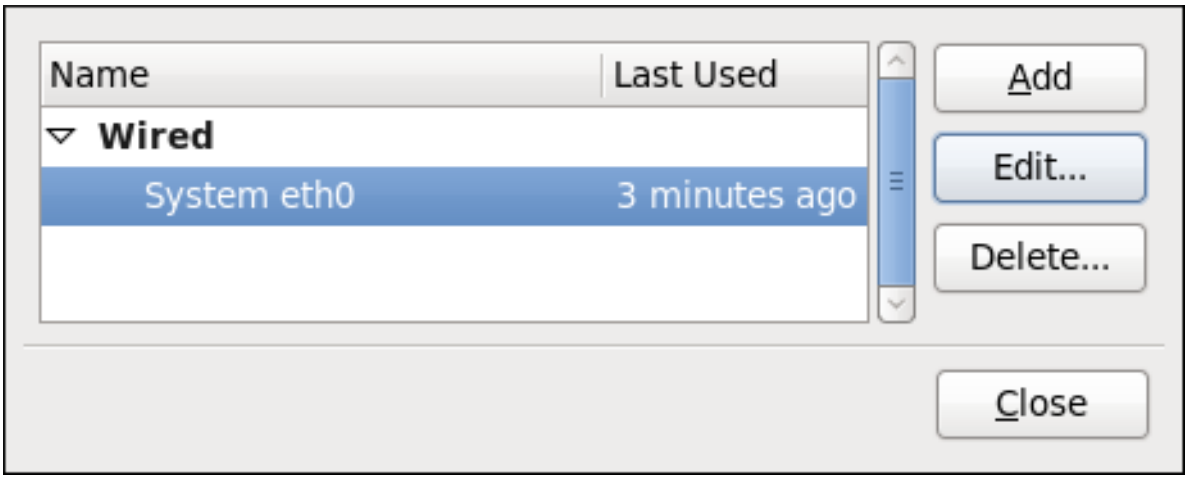


図8.6 新規作成の System eth0 接続を表示しているネットワーク接続ウィンドウ

システムのスタートアップスクリプトは、デフォルトで全システム上で **System eth0** と呼ばれる単独の有線接続を作成・設定します。**System eth0** は編集できますが、個人のカスタム設定で新規に有線接続を作成することが推奨されます。新規の有線接続を作成するには、**追加** ボタンをクリックし、表示されるリストから **有線** を選択して **作成** ボタンをクリックします。

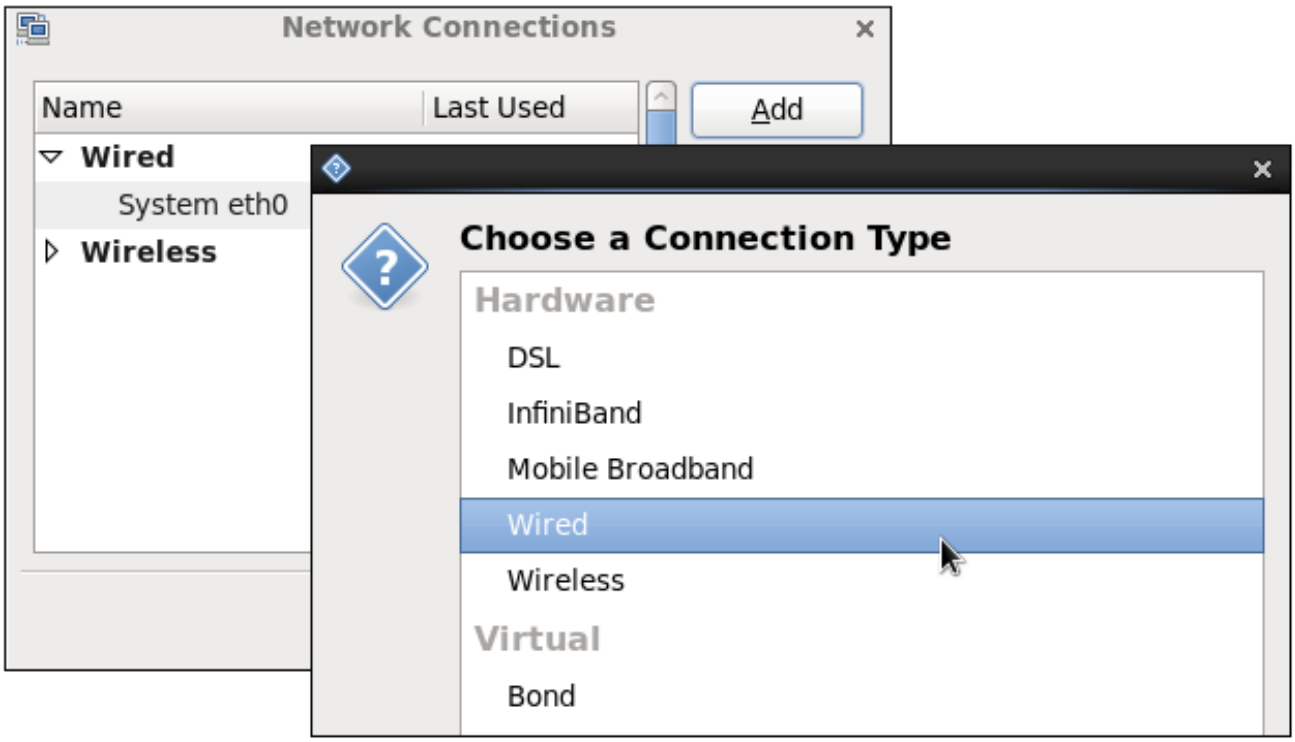


図8.7 「接続のタイプを選択」から新規接続のタイプを選択

注記

追加 ボタンをクリックして新規接続を追加すると、接続タイプのリストが表示されます。選択後に**作成** ボタンをクリックすると、**NetworkManager** はその接続の新規設定ファイルを作成し、既存接続の編集に使用されるダイアログと同じものが開きます。これらのダイアログには違いはありません。結局は常に接続を編集することになります。唯一の違いは、その接続が以前から存在してものか、あるいは **追加** をクリックして **NetworkManager** が作成したばかりのものか、という点だけです。

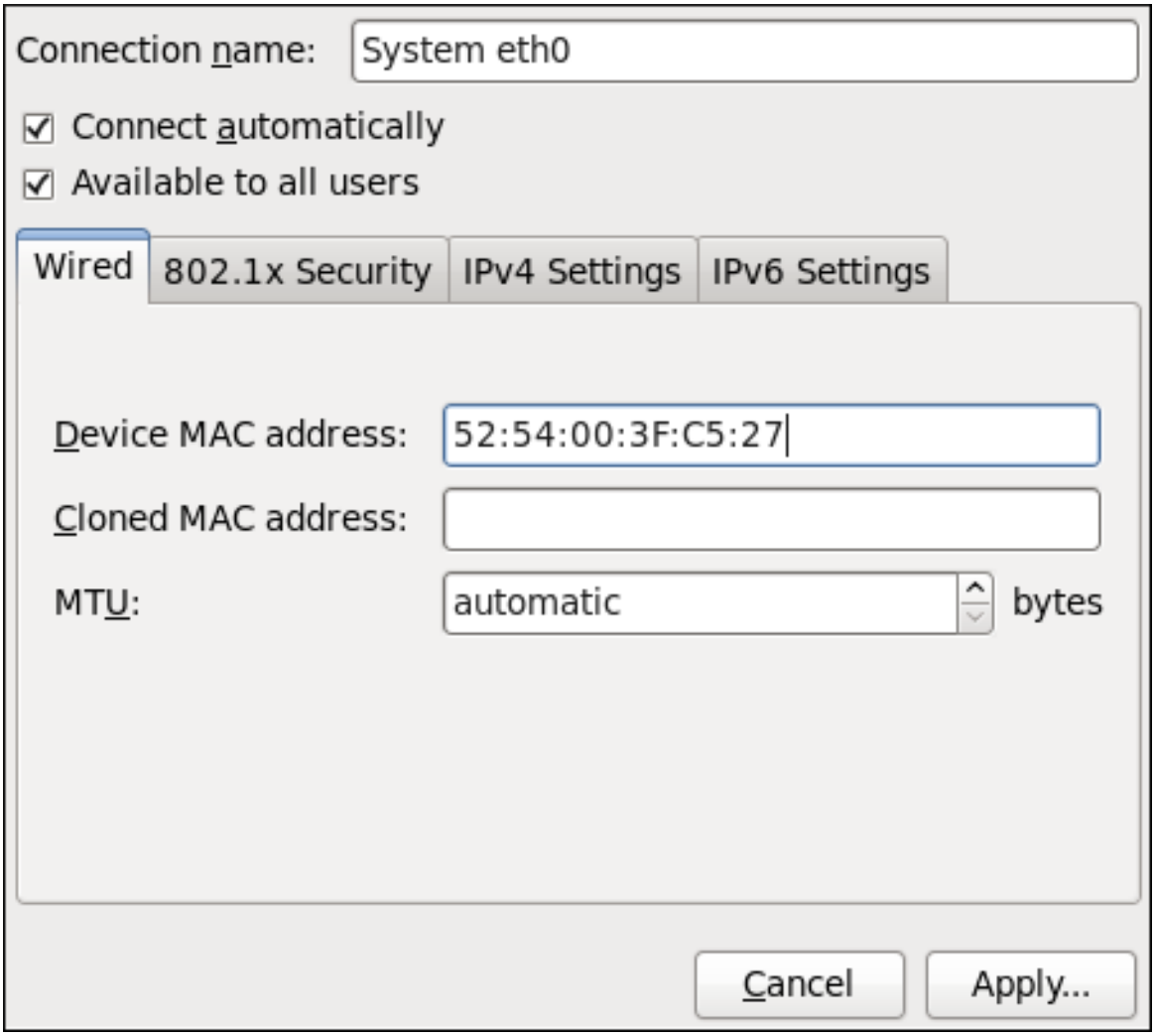


図8.8 新規作成された有線接続 System eth0 の編集

接続名、自動接続の動作、及び可用性セッティングの設定

編集 ダイアログ内の３つのセッティングはすべての接続タイプに共通です:

- **接続名** — ネットワーク接続の名前を入力します。この名前は、**ネットワーク接続** ウィンドウの **有線** セクションでこの接続の表示に使用されます。
- **自動接続する** — このボックスにチェックを入れると、この接続が利用可能な時に**NetworkManager** が自動接続します。詳細については 「ネットワークへの自動接続」 を参照してください。
- **すべてのユーザーに利用可能** — このシステム上のすべてのユーザーに利用可能な接続を作成するには、このボックスにチェックを入れます。この設定を変更するには root 権限が必要な場合があります。詳細については「ユーザー接続とシステム接続」 を参照してください。

有線タブの設定

設定可能な残りの 3 つのセッティングは **有線** タブ内にあります。1 つ目は MAC (Media Access Control) アドレスを指定するテキスト入力フィールドで、2 つ目はクローンした MAC アドレス を指定し、3 つ目は MTU (Maximum Transmission Unit) の値を指定します。通常、**MAC アドレス** のフィールドは空白に、**MTU** は **自動** にしておきます。有線接続を 2 つ目または特定の NIC に関連付けたり、高度なネットワーキングを実行していない限り、これらのデフォルト値で十分です。特別な設定については以下の説明を参照して下さい。

MAC アドレス

ネットワークインターフェイスカード (NIC) などのネットワークハードウェアは、システムに対してハードウェアを識別する特有の MAC アドレス (Media Access Control; 別名 *ハードウェアアドレス*) を持っています。**ip addr** コマンドを実行すると、各インターフェイスに関連した MAC アドレスが表示されます。例えば、以下の **ip addr** の出力では、**eth0** インターフェイス用の MAC アドレス (**52:54:00:26:9e:f1**) は **link/ether** キーワードの直後に表示されています。

```
~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UNKNOWN qlen 1000
    link/ether 52:54:00:26:9e:f1 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.251/24 brd 192.168.122.255 scope global eth0
    inet6 fe80::5054:ff:fe26:9ef1/64 scope link
       valid_lft forever preferred_lft forever
```

1 つのシステムに単独の、または複数の NIC をインストールできます。ですから、**MAC アドレス** フィールドでは特定の接続 (または接続群) に対して特定の NIC を関連付けることができます。前述のように **ip addr** コマンドを使用することで、MAC アドレスが判定できるため、その値をコピーして **MAC アドレス**テキスト入力フィールドに貼り付けることができます。

クローンした MAC アドレスフィールドは、ネットワークサービスが特定の MAC アドレスに限定され、その MAC アドレスをエミュレートする必要があるという状況で多く使用されます。

MTU

MTU (Maximum Transmission Unit) の値は、接続が転送に使用する最大パケット数のバイトサイズを示します。この値は、IPv4 の使用ではデフォルトで **1500** となり、IPv6 の使用では可変数 **1280** かそれ以上がデフォルトとなります。一般的に指定や変更をする必要はありません。

新規 (または修正した) 接続を保存して他の設定をする

有線接続の編集が終了して、**適用** ボタンをクリックすると**NetworkManager** は即座にカスタマイズした設定を保存します。設定が正しければ、**NetworkManager** 通知スペースアプレットから新規またはカスタマイズした接続を選択して、その接続が確立されます。新規または変更した接続の使用に関する詳細については 「ネットワークへの接続」 を参照してください。

既存の接続をさらに設定をするには、**ネットワーク接続** ウィンドウ内でその接続を選択し、**編集** をクリックして **編集** ダイアログに戻ります。

そして、以下のいずれかの設定をします:

- ポートベースネットワークアクセスコントロール (PNAC): **802.1x セキュリティ** タブをクリックして「802.1x セキュリティの設定」 へ進みます;
- 接続に対する IPv4 のセッティング: **IPv4 のセッティング** タブをクリックして 「IPv4 セッティングの設定」 へ進むか、または;
- 接続に対する IPv6 のセッティング: **IPv6 のセッティング** タブをクリックして 「IPv6 セッティングの設定」 に進みます。

8.3.2. 無線接続の確立

このセクションでは、**NetworkManager** でアクセスポイントへの無線接続 (別名 Wi-Fi または 802.1*a/b/g/n*) の設定方法を説明します。

(3G などの) モバイルブロードバンドの設定方法については、「モバイルブロードバンド接続の確立」を参照してください。

利用可能なアクセスポイントに迅速に接続

利用可能なアクセスポイントに接続する最も簡単な方法は、**NetworkManager** アプレットを左クリックして **利用可能** ネットワークの一覧内でアクセスポイントの *サービスセット識別子* (SSID) を見つけてそれをクリックすることです。アクセスポイントが保護されている場合は、認証を求めるダイアログが出てきます。

図8.9 無線アクセスポイントへの認証

NetworkManager は、アクセスポイントで使われているセキュリティタイプを自動検出しようとします。複数の可能性がある場合、**NetworkManager** はセキュリティタイプを予想し、それを **無線セキュリティ** ドロップダウンメニューに表示します。複数の選択肢があるかどうかは、**無線セキュリティ** ドロップダウンメニューをクリックしてアクセスポイントが使用しているセキュリティタイプを選択するとわかります。不明な場合は、それぞれのタイプに接続してみます。最後に **パスワード** フィールドにキー、またはパスフレーズを記入します。40-bit WEP や 128-bit WPA キーなどの特定のパスワードのタイプは、指定の長さに達していないと無効になります。選択したセキュリティタイプで要求される長さのキーを記入するまでは、**接続** ボタンは灰色のままです。無線セキュリティに関する情報については「無線セキュリティの設定」 を参照して下さい。

注記

WPA および WPA2 (個人およびエンタープライズ) の場合、自動、WPA、WPA2 を選択するオプションが加えられます。このオプションは、WPA と WPA2 の両方が利用可能なアクセスポイントでの使用を念頭に置いています。両方のプロトコル間でのローミングを防止したい場合は、どちらかのプロトコルを選択します。同一アクセスポイントでの WPA と WPA2 間のローミングは、サービスの損失につながる恐れがあります。

NetworkManager が正しくアクセスポイントに接続できた場合には、そのアプレットアイコンが無線接続信号の強度を示すグラフィカルインディケーターに変化します。

図8.10 無線接続信号の強度 75% を示しているアプレットアイコン

また、自動作成されたアクセスポイント接続の設定を、まるで自分で追加したように編集することもできます。**ネットワーク接続** ウィンドウの **無線** タブは以前に接続を試みたすべての接続先を一覧表示します。**NetworkManager** はそれらの 1 つ 1 つを**Auto *<SSID>*** 形式で名付けます。ここでの SSID とはアクセスポイントの *サービスセット識別子* のことです。

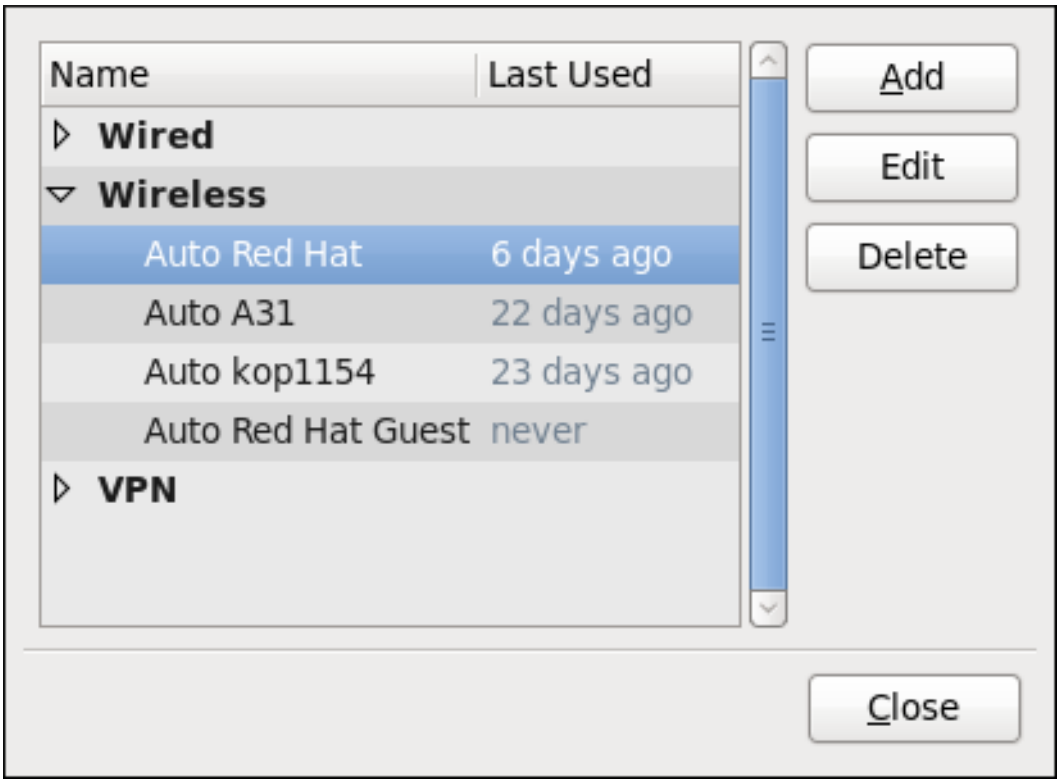


図8.11 以前に接続したことのあるアクセスポイントの例

非表示無線ネットワークに接続

すべてのアクセスポイントには、識別するために *SSID* (SSID) があります。しかし、あるアクセスポイントはその SSID をブロードキャストしないように設定されている場合もあり、そのケースでは **非表示** となり、**NetworkManager** 内の**利用可能** ネットワーク一覧には表示されません。ただし、SSID と認証方法、秘密情報が分かっていれば、非表示の無線アクセスポイントにも接続することは可能です。

非表示無線ネットワークに接続するには、**NetworkManager** のアプレットアイコンを左クリックして **非表示無線ネットワークに接続** を選択することで、ダイアログが出現するようにします。以前に非表示ネットワークに接続したことがある場合は、**接続** ドロップダウンメニューを使用してそれを選択し、**接続** ボタンをクリックします。以前に使用したことがない場合は、**接続** ドロップダウンを **新規** のままにしておきます。非表示ネットワークの SSID を記入してからその **無線セキュリティ** メソッドを選択し、正しい認証用秘密情報を記入してから**Connect** ボタンをクリックします。

無線セキュリティの設定に関する詳細情報については、「無線セキュリティの設定」 を参照して下さい。

接続の編集、または全く新しい接続を作成

過去に試行した、あるいは接続に成功したことのある既存の接続を編集するには、**ネットワーク接続** の **無線** タブを開いて、その名前 (**Auto** の次に来る単語はアクセスポイントの SSID を参照) で接続を選択します。そして **編集** をクリックします。

新しい接続を作成するには、**ネットワーク接続** ウィンドウを開いてから、**追加** ボタンをクリックし、**無線** を選択して **作成** ボタンをクリックします。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。

2. **追加** ボタンをクリックします。

3. リストから **無線** エントリを選択します。

4. **作成**ボタンをクリックします。

図8.12 新規作成した無線接続 1 の編集

接続名、自動接続の動作、及び可用性セッティングの設定

編集 ダイアログ内の３つのセッティングはすべての接続タイプに共通です:

- **接続名** — ネットワーク接続の名前を入力します。この名前は、**ネットワーク接続** ウィンドウの **無線** セクションでこの接続の表示に使用されます。デフォルトでは、無線接続は無線アクセスポイントの *SSID* と同じ名前になっています。無線接続の名前を変更しても接続機能に影響は与えませんが、SSID 名の維持が推奨されます。

- **自動接続する** — このボックスにチェックを入れると、この接続が利用可能な時に**NetworkManager** が自動接続します。詳細については 「ネットワークへの自動接続」 を参照してください。

- **すべてのユーザーに利用可能** — このシステム上のすべてのユーザーに利用可能な接続を作成するには、このボックスにチェックを入れます。この設定を変更するには root 権限が必要な場合があります。詳細については「ユーザー接続とシステム接続」 を参照してください。

無線タブの設定

SSID

すべてのアクセスポイントは、それら自身の識別のために *SSID* (Service Set identifier) を持っています。しかし、あるアクセスポイントはその SSID を広報しないように設定されているかも知れません。その場合、それは**非表示** となり、**NetworkManager** 内の**使用可能** ネットワーク一覧に出現しないことになります。但し、その SSID と認証秘密情報が判明していれば、SSID 非表示の無線アクセスポイントにも接続することは可能です。\n

非表示無線ネットワークへの接続については 「非表示無線ネットワークに接続」 を参照して下さい。

モード

インフラストラクチャ — 専用の無線アクセスポイントや、ルーターやスイッチなどのネットワークデバイスに構築されているアクセスポイントに接続している場合は、**モード** を **インフラストラクチャ** に設定します。

アドホック — 2 つ以上のモバイルデバイス間での直接通信用のピアツーピアネットワークを作成している場合は、**モード** を **アドホック** に設定します。802.11 標準では *Independent Basic Service Set* (IBSS) と呼ばれる **アドホック** モードを使用する場合は、すべての参加無線デバイスで同一 **SSID** が設定され、それらすべてが同一チャンネル上で通信していることを確認する必要があります。

BSSID

BSSID (基本的サービスセット識別子) は、**インフラストラクチャ** モードで接続している際の特定の無線アクセスポイントの MAC アドレスです。このフィールドはデフォルトで空白になっており、**BSSID** を指定せずに、**SSID** で無線アクセスポイントに接続できます。BSSID が指定されると、システムは強制的に特定のアクセスポイントのみに関連付けられます。

アドホックネットワークが生成される際に、mac80211 サブシステムがランダムに BSSID を生成します。これは **NetworkManager** では表示されません。

MAC アドレス

イーサネットのネットワークインターフェイスカード (NIC) と同様に、無線アダプタはそれ自身をシステムに対して識別する特有の MAC アドレス (Media Access Control; 別名 *ハードウェアアドレス*) を持っています。**ip addr** コマンドを実行すると、各インターフェイスに関連付けられた MAC アドレスが表示されます。例えば、以下の **ip addr** 出力では、**wlan0** インターフェイスの MAC アドレス(**00:1c:bf:02:f8:70**) は、キーワード**link/ether** の直後に続いています。

```
~]# ip addr
```

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UNKNOWN qlen 1000
    link/ether 52:54:00:26:9e:f1 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.251/24 brd 192.168.122.255 scope global eth0
    inet6 fe80::5054:ff:fe26:9ef1/64 scope link
       valid_lft forever preferred_lft forever
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
qlen 1000
    link/ether 00:1c:bf:02:f8:70 brd ff:ff:ff:ff:ff:ff
    inet 10.200.130.67/24 brd 10.200.130.255 scope global wlan0
    inet6 fe80::21c:bfff:fe02:f870/64 scope link
       valid_lft forever preferred_lft forever
```

単独のシステムでも 1 つ、またはそれ以上の無線ネットワークアダプターを接続することができます。そのため、**MAC アドレス** フィールドでは特定の無線アダプターが特定の接続と関連付けできるようになっています。前述のように **ip addr** コマンドを使用すれば MAC アドレスを判定できるため、その値をコピーして **MAC アドレス** テキスト入力フィールドに貼り付けることができます。

MTU

MTU (Maximum Transmission Unit) (最大転送ユニット)の値は、接続が転送に使用する最大限パケットサイズをバイト単位で示すものです。ゼロ以外の数値でセットしてある場合は、その指定サイズか、それ以下のパケットのみが転送されます。それ以上のパケットサイズは複数のイーサネットフレームに分断されます。このセッティングは **自動** のままにしておくことが推奨されます。

新規 (または修正した) 接続を保存して他の設定をする

無線接続の編集が終了すると、**適用** ボタンをクリックします。すると**NetworkManager** はすぐにカスタマイズした設定を保存します。設定が正しく実行されている場合は、**NetworkManager** 通知スペースアプレットからそれを選択すると、変更した接続先へ正しく接続するようになります。

既存の接続をさらに設定をするには、**ネットワーク接続** ウィンドウ内でその接続を選択し、**編集** をクリックして **編集** ダイアログに戻ります。

そして、以下のいずれかの設定をします:

- 無線接続用のセキュリティ認証には、**無線セキュリティ** タブをクリックして 「無線セキュリティの設定」 へ進みます;
- 接続に対する IPv4 のセッティング: **IPv4 のセッティング** タブをクリックして 「IPv4 セッティングの設定」 へ進むか、または;
- 接続に対する IPv6 のセッティング: **IPv6 のセッティング** タブをクリックして 「IPv6 セッティングの設定」 に進みます。

8.3.3. モバイルブロードバンド接続の確立

NetworkManager のモバイルブロードバンド接続機能を使用すると、以下の*2G* と *3G* のサービスに接続することができます:

- 2G — *GPRS* (*General Packet Radio Service*) または *EDGE* (*Enhanced Data Rates for GSM Evolution*)
- 3G — *UMTS* (*Universal Mobile Telecommunications System*) または *HSPA* (*High Speed Packet Access*)

接続を作成するためには、システムが検索し、認識しているモバイルブロードバンドのデバイス (モデム) を、コンピュータが備えている必要があります。そのようなデバイスはコンピュータに内蔵されている場合 (多くのノートブックやネットブック) と、外付けまたは内蔵のハードウェアとして提供されている場合があります。例えば、PC カードや USB モデム、ドングル、モデムとして機能する携帯電話などです。

手順8.3 新規のモバイルブロードバンド接続の追加

モバイルブロードバンド接続の設定は、**ネットワーク接続** ウィンドウを開いてから、**追加** をクリックし、**モバイルブロードバンド** を選択します。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。
2. **追加** ボタンをクリックして選択一覧を開きます。**モバイルブロードバンド** を選択し、**作成** をクリックします。**モバイルブロードバンド接続のセットアップ**アシスタントが表示されます。
3. **このモバイルブロードバンドデバイス用の接続を作成** の下で、その接続で使用する 2G または 3G 機能のデバイスを選択します。ドロップダウンメニューが使用できない場合は、システムがモバイルブロードバンドの機能を持つデバイスを検出できなかったことを示します。この状況では、**キャンセル** をクリックして、モバイルブロードバンド機能のデバイスが接続してあること、そしてそれがコンピュータに認識されていることを確認してください。それからこの手順を再試行します。**進む** ボタンをクリックします。
4. 使用するサービスプロバイダーの国をリストから選んで **進む** ボタンをクリックします。
5. プロバイダーをリストから選ぶか手動で入力し、**進む** ボタンをクリックします。
6. ドロップダウンメニューから請求書プランを選んで、*Access Point Name* (APN) が正しいか確認します。**進む** ボタンをクリックします。
7. 設定を確認して、**適用** ボタンをクリックします。
8. モバイルブロードバンド特定の設定での編集は、以下の **モバイルブロードバンドタブの設定** を参照してください。

手順8.4 既存のモバイルブロードバンド接続の編集

既存のモバイルブロードバンド接続を編集するには以下の手順に従います。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。
2. 設定したい接続を選択して、 **編集** ボタンをクリックします。
3. **モバイルブロードバンド** タブを選択します。
4. 接続名、自動接続の動作、及び可用性のセッティングを設定します。

   編集 ダイアログ内の３つのセッティングはすべての接続タイプに共通です:

- **接続名** — ネットワーク接続の名前を入力します。この名前は、**ネットワーク接続** ウィンドウの **モバイルブロードバンド** セクションでこの接続の表示に使用されます。

- **自動接続する** — このボックスにチェックを入れると、この接続が利用可能な時に**NetworkManager** が自動接続します。詳細については 「ネットワークへの自動接続」 を参照してください。

- **すべてのユーザーに利用可能** — このシステム上のすべてのユーザーに利用可能な接続を作成するには、このボックスにチェックを入れます。この設定を変更するには root 権限が必要な場合があります。詳細については「ユーザー接続とシステム接続」 を参照してください。

5. モバイルブロードバンド特定の設定での編集は、以下の **モバイルブロードバンドタブの設定** を参照してください。

新規 (または修正した) 接続を保存して他の設定をする

モバイルブロードバンド接続の編集が終了して、**適用** ボタンをクリックすると**NetworkManager** は即座にカスタマイズした設定を保存します。設定が正しければ、**NetworkManager** 通知スペースアプレットから新規またはカスタマイズした接続を選択して、その接続が確立されます。新規または変更した接続の使用に関する詳細については 「ネットワークへの接続」 を参照してください。

既存の接続をさらに設定をするには、**ネットワーク接続** ウィンドウ内でその接続を選択し、**編集** をクリックして **編集** ダイアログに戻ります。

そして、以下のいずれかの設定をします:

- 接続のポイントツーポイント (Point-to-point) セッティングには、**PPP セッティング** タブをクリックして、「PPP (Point-to-Point) セッティングの設定」 に進みます;

- 接続に対する IPv4 のセッティング: **IPv4 のセッティング** タブをクリックして 「IPv4 セッティングの設定」 へ進むか、または;

- 接続に対する IPv6 のセッティング: **IPv6 のセッティング** タブをクリックして 「IPv6 セッティングの設定」 に進みます。

モバイルブロードバンドタブの設定

アシスタント (その案内には 手順8.3「新規のモバイルブロードバンド接続の追加」 を参照) を使用して既に新規のモバイルブロードバンド接続を追加しているのであれば、**モバイルブロードバンド** タブを編集して、ホームネットワークが使用不可能な場合はローミングを無効にしたり、この接続を使用する時にネットワーク ID を割り当てたり、または特定の技術 (3G や 2Gなど) を優先するように **NetworkManager** に指示することができます。

番号

GSM ベースのモバイルブロードバンドネットワークでの PPP 接続を確立するためにダイアルする番号です。このフィールドは、ブロードバンドデバイスの初期インストールの際に自動設定されている場合があります。通常、このフィールドは空白で残し、代わりに **APN** を記入します。

ユーザー名

ネットワークでの認証に使用するユーザー名を記入します。一部のプロバイダーは、ユーザー名を提供しないことや、ネットワーク接続の時点でユーザー名を受け付けたりすることがあります。

パスワード

ネットワークで認証に使用するパスワードを記入します。一部のプロバイダーはパスワードを提供しなかったり、またはすべてのパスワードを受け付けたりします。

APN

GSM ベースのネットワークとの接続を確立するために使用する *Access Point Name* (APN) を記入します。これは以下の項目を決定するので、正しい APN を記入することが重要になります。

- ネットワーク使用量についてユーザーが請求される方法、及び/あるいは
- ユーザーがインターネット、イントラネット、サブネットワークにアクセスがあるかどうか。

ネットワーク ID

ネットワーク ID を記入すると、**NetworkManager** は強制的にデバイスが特定のネットワークのみに登録されるようにします。これにより、ローミングを直接に制御できない時に接続がローミングしないようにします。

タイプ

Any — **Any** のデフォルト値では、モデムが最速のネットワークを選択します。

3G (UMTS/HSPA) — 接続が 3G ネットワーク技術のみを使用するように強制します。

2G (GPRS/EDGE) — 接続が 2G ネットワーク技術のみを使用するように強制します。

Prefer 3G (UMTS/HSPA) — 最初に HSPA または UMTS などの 3G 技術を使用した接続を試み、失敗した後にのみ GPRS または EDGE にフォールバックします。

Prefer 2G (GPRS/EDGE) — 最初に GPRS または EDGE などの 2G 技術を使用した接続を試み、失敗した後にのみ HSPA または UMTS にフォールバックします。

ホームネットワークが使用できない場合にローミングを許可

ホームネットワークからローミングへの移行ではなく、**NetworkManager** が接続を終了するようにするには、このボックスからチェックを外します。これにより、ローミング料金を回避できます。ボックスにチェックが入っていると、**NetworkManager** はホームネットワークからローミングに、またはその逆に切り替えることで接続を維持しようとします。

PIN

デバイスの *SIM* (*Subscriber Identity Module (購読者識別モジュール)*) が *PIN* (*Personal Identification Number (個人識別番号)*) でロックされている場合は、その PIN を入力して **NetworkManager** がデバイスのロックを解除できるようにします。どんな目的でもデバイスの使用に PIN を必要とする場合は、**NetworkManager** は SIM をロック解除する必要があります。

8.3.4. VPN 接続の確立

仮想プライベートネットワーク (VPN) を確立すると、使用中の LAN (ローカルエリアネットワーク) ともう 1 つの LAN との間で安全に通信ができるようになります。VPN 接続を正しく確立した後には、VPN ルーターまたはゲートウェイはユーザーが転送したパケットに対して以下のアクションを実行します。

1. ルーティングと認証の目的で *認証*ヘッダー を追加
2. パケットデータを暗号化。及び
3. 復号化と処理の方法から構成されている ESP (Encapsulating Security Payload) を使用してデータを包装

受信側の VPN ルーターはヘッダー情報を開いて、データを解読し、それを想定された目的地 (ネットワーク上のワークステーション、または他のノード) に回送します。ネットワーク対ネットワーク接続を使用すると、ローカルネットワークの受信側ノードは既に解読されてプロセスの準備ができた状態のパケットを受信します。ネットワーク対ネットワークの VPN 接続での暗号化と復号化のプロセスはこのようにクライアントに透過的になっています。

VPN は、認証と暗号化で数種のレイヤーを採用するため、統合されたイントラネットとして機能する複数のリモートノードを接続するのに安全で効果的な手段となります。

手順8.5 新規 VPN 接続の追加

1. 新規 VPN 接続の設定は、**ネットワーク接続** ウィンドウを開いてから、**追加** をクリックし、新規接続リストの **VPN** セクションから VPN のタイプを選択します。

2. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。

3. **追加** ボタンをクリックします。

4. **VPN 接続のタイプを選択** 一覧が表示されます。

5.

   注記

   設定する VPN タイプに適切な NetworkManager VPN プラグインをインストールする必要があります。(Red Hat Enterprise Linux 6 に新規パッケージをインストールする方法については、「パッケージのインストール」 を参照してください)。適切なプラグインがインストールされていないと、**接続のタイプを選択** 一覧の**VPN** セクションは表示 **されません** 。

6. 接続先のゲートウェイの VPN プロトコルを **接続のタイプを選択** 一覧から選びます。リストで利用可能な VPN プロトコルは、インストール済みの NetworkManager VPN プラグインに対応します。例えば、NetworkManager VPN plug-in for openswan がインストールされていれば、IPsec ベースの VPN が **接続のタイプを選択** 一覧から選べます。

   正しいプロトコルを選択した後、**作成** ボタンをクリックします。

7. そうすると、**VPN 接続 *1* の編集** ウィンドウが出てきます。このウィンドウは ステップ 6 で選択した VPN 接続のタイプ用にカスタマイズしたセッティングを表示します。

手順8.6 既存の VPN 接続の編集

既存の VPN 接続を設定するには、**ネットワーク接続** ウィンドウを開いてから、リストにある接続名を選択します。そして **編集** ボタンをクリックします。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。

2. 設定したい接続を選択して、 **編集** ボタンをクリックします。

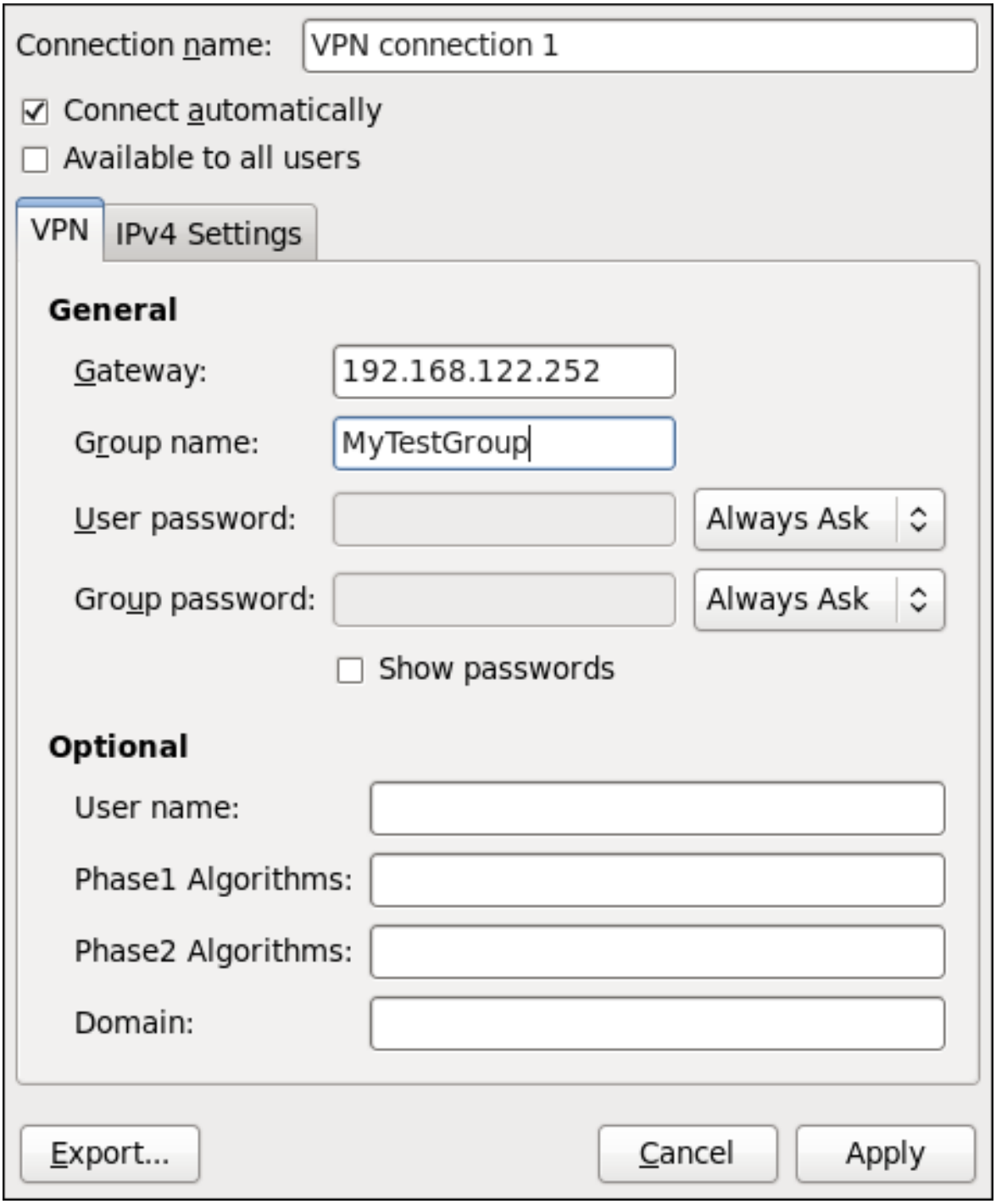


図8.13 新規作成した VPN 接続 1 の編集

接続名、自動接続の動作、及び可用性セッティングの設定

編集 ダイアログ内の３つのセッティングはすべての接続タイプに共通です:

- **接続名** — ネットワーク接続の名前を入力します。この名前は、**ネットワーク接続** ウィンドウの **VPN** セクションでこの接続の表示に使用されます。
- **自動接続する** — このボックスにチェックを入れると、この接続が利用可能な時に**NetworkManager** が自動接続します。詳細については 「ネットワークへの自動接続」 を参照してください。
- **すべてのユーザーに利用可能** — このシステム上のすべてのユーザーに利用可能な接続を作成するには、このボックスにチェックを入れます。この設定を変更するには root 権限が必要な場合があります。詳細については「ユーザー接続とシステム接続」 を参照してください。

VPN タブの設定

ゲートウェイ

リモート VPN ゲートウェイの名前または IP アドレスです。

グループ名

リモートゲートウェイで設定された VPN グループ名です。

ユーザーパスワード

必要な場合は、VPN 認証で使用するパスワードを入力します。

グループパスワード

必要な場合は、VPN 認証で使用するパスワードを入力します。

ユーザー名

必要な場合は、VPN 認証で使用するユーザー名を入力します。

`Phase1 Algorithms`

必要な場合は、暗号化チャンネルの認証および設定で使用するアルゴリズムを入力します。

`Phase2 Algorithms`

必要な場合は、IPsec ネゴシエーションで使用するアルゴリズムを入力します。

ドメイン

必要な場合は、ドメイン名を入力します。

`NAT` トラバーサル

Cisco UDP (デフォルト) — UDP を介した IPsec

NAT-T — NAT トラバーサル処理に ESP カプセル化と IKE 拡張を使用

無効 — 特別な NAT 措置は不要

`Disable Dead Peer Detection` — リモートゲートウェイやエンドポイントへのプローブの送信を無効にします。

新規 (または修正した) 接続を保存して他の設定をする

新規の VPN 接続の編集が終了した時点で **適用** ボタンをクリックすると、**NetworkManager** はすぐにカスタマイズした設定を保存します。設定が正しいならば、**NetworkManager** の通知スペースアプレットからそれを選択することで新しい、またはカスタマイズした接続先に接続することができます。新規の、または変更した接続の使用法については 「ネットワークへの接続」 を参照して下さい。

既存の接続をさらに設定をするには、**ネットワーク接続** ウィンドウ内でその接続を選択し、**編集** をクリックして **編集** ダイアログに戻ります。

そして、以下のいずれかの設定をします:

- 接続に対する IPv4 のセッティング: **`IPv4` のセッティング** タブをクリックして 「IPv4 セッティングの設定」 に進みます。

8.3.5. DSL 接続の確立

このセクションでは、個人ユーザーや SOHO インストールでよくある DSL モデムルーターの外部の組み合わせではなく、ホスト内に DSL カードが組み込まれているインストールについて説明します。

手順8.7 新規 DSL 接続の追加

新しい DSL 接続を設定するには、**ネットワーク接続** ウィンドウを開いてから、**追加** ボタンをクリックし、新規接続一覧の **ハードウェア** セクションから **DSL** を選択します。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。
2. **追加** ボタンをクリックします。
3. **VPN 接続のタイプを選択** 一覧が表示されます。
4. **DSL** を選択し、**作成** ボタンをクリックします。
5. **DSL 接続 *1* の編集** ウィンドウが表示されます。

手順8.8 既存の DSL 接続の編集

既存の DSL 接続を設定するには、**ネットワーク接続** ウィンドウを開いてから、リストにある接続名を選択します。そして **編集** ボタンをクリックします。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。
2. 設定したい接続を選択して、 **編集** ボタンをクリックします。

接続名、自動接続の動作、及び可用性セッティングの設定

編集 ダイアログ内の３つのセッティングはすべての接続タイプに共通です:

- **接続名** — ネットワーク接続の名前を入力します。この名前は、**ネットワーク接続** ウィンドウの **DSL** セクションでこの接続の表示に使用されます。
- **自動接続する** — このボックスにチェックを入れると、この接続が利用可能な時に**NetworkManager** が自動接続します。詳細については 「ネットワークへの自動接続」 を参照してください。
- **すべてのユーザーに利用可能** — このシステム上のすべてのユーザーに利用可能な接続を作成するには、このボックスにチェックを入れます。この設定を変更するには root 権限が必要な場合があります。詳細については「ユーザー接続とシステム接続」 を参照してください。

DSL タブの設定

ユーザー名

サービスプロバイダー認証で使用するユーザー名を入力します。

サービス

特別な指示がない限り、空白のままにします。

パスワード

サービスプロバイダーから提供されたパスワードを入力します。

新規 (または修正した) 接続を保存して他の設定をする
DSL 接続の編集が終了して、**適用** ボタンをクリックすると**NetworkManager** は即座にカスタマイズした設定を保存します。設定が正しければ、**NetworkManager** 通知スペースアプレットから新規またはカスタマイズした接続を選択して、その接続が確立されます。新規または変更した接続の使用に関する詳細については 「ネットワークへの接続」 を参照してください。

既存の接続をさらに設定をするには、**ネットワーク接続** ウィンドウ内でその接続を選択し、**編集** をクリックして **編集** ダイアログに戻ります。

そして、以下のいずれかの設定をします:

- MAC アドレスおよび MTU 設定、**有線** タブをクリックして 「有線タブの設定」 に進みます。
- 接続のポイントツーポイント (Point-to-point) セッティングには、**PPP セッティング** タブをクリックして、「PPP (Point-to-Point) セッティングの設定」 に進みます;
- 接続に対する IPv4 のセッティング: **IPv4 のセッティング** タブをクリックして 「IPv4 セッティングの設定」 に進みます。

8.3.6. Bond 接続の確立

NetworkManager を使って、2 つ以上の有線もしくは Infiniband 接続から Bond を作成することができます。接続が最初に結びついている必要はありません。Bond を設定するプロセスの一部として設定が可能です。この設定プロセスを完了するには、利用可能なインターフェースの MAC アドレスが必要です。

手順8.9 新規 Bond 接続の追加

Bond 接続を設定するには、**ネットワーク接続** ウィンドウを開いてから、**追加** ボタンをクリックし、一覧から **Bond** を選択します。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。
2. **追加** ボタンをクリックして選択一覧を開きます。**Bond** を選択し、**作成** をクリックします。**Bond 接続 *1* の編集** ウィンドウが表示されます。
3. **Bond** タブで **追加** をクリックし、Bond 接続で使用するインターフェースのタイプを選択します。**作成** ボタンをクリックします。スレーブタイプを選択するダイアログが表示されるは、最初のスレーブを作成する時のみです。その後は、すべてのスレーブに同じタイプが自動的に使われます。
4. **bond1 slave1 の編集** ウィンドウが表示されます。最初に結びつけるインターフェースの MAC アドレスを入力します。**適用** ボタンをクリックします。
5. **認証** ウィンドウが表示されます。root パスワードを入力して続けます。**認証** ボタンをクリックします。
6. **結合接続ウィンドウ** に結合されたスレーブの名前が表示されます。**追加** ボタンをクリックしてさらにスレーブ接続を追加します。
7. 設定を確認して、**適用** ボタンをクリックします。

8. Bondj 特定の設定での編集は、以下の 「Bond タブの設定」 を参照してください。

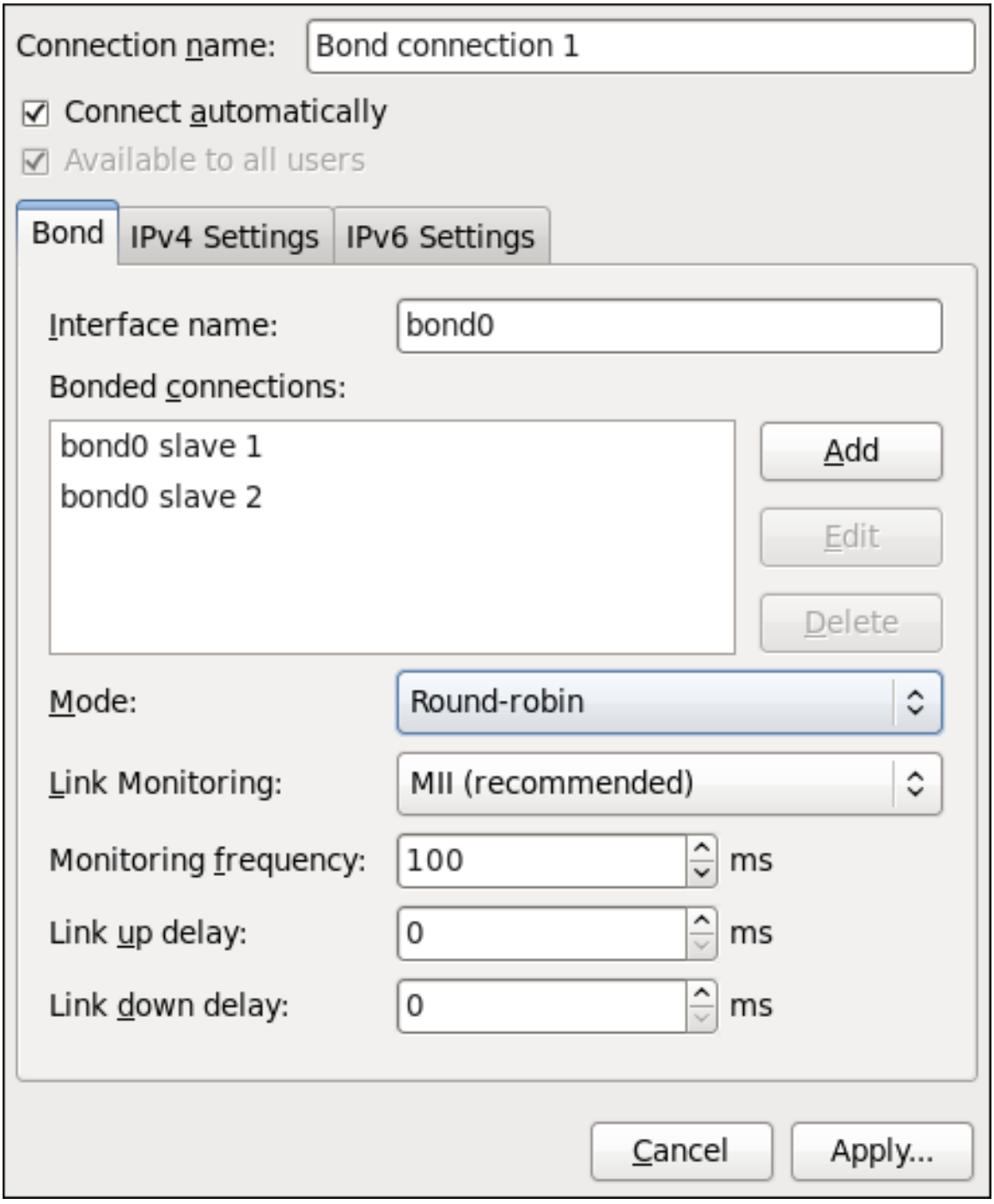


図8.14 新規作成した Bond 接続 1 の編集

手順8.10 既存の Bond 接続の編集

既存の Bond 接続を編集するには以下の手順に従います。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。
2. 設定したい接続を選択して、 **編集** ボタンをクリックします。
3. **Bond** タブを選択します。
4. 接続名、自動接続の動作、及び可用性のセッティングを設定します。

編集 ダイアログ内の３つのセッティングはすべての接続タイプに共通です:

- **接続名** — ネットワーク接続の名前を入力します。この名前は、**ネットワーク接続** ウィンドウの **Bond** セクションでこの接続の表示に使用されます。
- **自動接続する** — このボックスにチェックを入れると、この接続が利用可能な時に**NetworkManager** が自動接続します。詳細については 「ネットワークへの自動接続」 を参照してください。
- **すべてのユーザーに利用可能** — このシステム上のすべてのユーザーに利用可能な接続を作成するには、このボックスにチェックを入れます。この設定を変更するには root 権限が必要な場合があります。詳細については「ユーザー接続とシステム接続」 を参照してください。

5. Bondj 特定の設定での編集は、以下の 「Bond タブの設定」 を参照してください。

新規 (または修正した) 接続を保存して他の設定をする

Bond 接続の編集が終了して、**適用** ボタンをクリックすると**NetworkManager** は即座にカスタマイズした設定を保存します。設定が正しければ、**NetworkManager** 通知スペースアプレットから新規またはカスタマイズした接続を選択して、その接続が確立されます。新規または変更した接続の使用に関する詳細については 「ネットワークへの接続」 を参照してください。

既存の接続をさらに設定をするには、**ネットワーク接続** ウィンドウ内でその接続を選択し、**編集** をクリックして **編集** ダイアログに戻ります。

そして、以下のいずれかの設定をします:

- 接続に対する IPv4 のセッティング: **IPv4 のセッティング** タブをクリックして 「IPv4 セッティングの設定」 へ進むか、または;
- 接続に対する IPv6 のセッティング: **IPv6 のセッティング** タブをクリックして 「IPv6 セッティングの設定」 に進みます。

Bond タブの設定

新規の Bond 接続を既に追加している場合 (手順に関しては、手順8.9「新規 Bond 接続の追加」 を参照) 、**Bond** タブを編集して、負荷分散モードとスレーブ接続の障害検出に使用するリンク監視のタイプを設定できます。

モード

Bond を構成するスレーブ接続でのトラフィック分散に使われるモード。デフォルトは、**ラウンドロビン** 。802.3ad などの他の負荷分散モードは、ドロップダウンリストから選択することができます。

リンク監視

ネットワークトラフィックを伝送するスレーブの能力を監視する方法。

以下の負荷分散モードが、**モード** のドロップダウンリストから選択できます。

ラウンドロビン

耐障害性とロードバランシングのためラウンドロビンポリシーを設定します。利用可能になったインターフェースから順に、それぞれのボンディングされたスレーブインターフェースで送受信が行われます。

アクティブバックアップ

耐障害性のためアクティブなバックアップポリシーを設定します。利用可能になった最初のボンディングされたスレーブインターフェースにより送受信が行われます。別のボンディングされたスレーブインターフェースは、アクティブなボンディングされたスレーブインターフェースが失敗した場合にのみ使用されます。これは、InfiniBand デバイスのボンドで利用可能な唯一のモードです。

XOR

耐障害性とロードバランシングのため XOR (排他的論理和) ポリシーを設定します。この方法を使用すると、インターフェースによって受信要求の MAC アドレスとスレーブ NIC の 1 つの MAC アドレスが一致します。このリンクが確立すると、利用可能になった最初のインターフェースから順に送信が行われます。

ブロードキャスト

耐障害性のためブロードキャストポリシーを設定します。すべての送信は、すべてのスレーブインターフェースで行われます。

802.3ad

IEEE 802.3ad 動的リンクアグリゲーションのポリシーを設定します。同一の速度とデュプレックス設定を共有するアグリゲーショングループを作成します。アクティブなアグリゲーターのすべてのスレーブで送受信を行います。802.3ad に対応するスイッチが必要です。

適応型送信ロードバランシング

耐障害性とロードバランシングのため適応型送信ロードバランシング (TLB) ポリシーを設定します。発信トラフィックは、各スレーブインターフェースの現在の負荷に従って分散されます。受信トラフィックは、現在のスレーブにより受信されます。受信しているスレーブが失敗すると、別のスレーブが失敗したスレーブの MAC アドレスを引き継ぎます。

アクティブロードバランシング

耐障害性とロードバランシングのためアクティブロードバランシング (ALB) ポリシーを設定します。IPv4 トラフィック用の送受信ロードバランシングが含まれます。ARP ネゴシエーションにより、受信ロードバランシングが可能です。

以下のリンク監視のタイプは、**リンク監視** ドロップダウンリストから選択できます。自分のボンディングされたインターフェースでどのチャンネルボンディングのモジュールパラメーターが一番うまく動作するかテストすることが推奨されます。

MII (Media Independent Interface)

インターフェースのキャリア波の状態を監視します。実行方法は、ドライバーへのクエリ、MII レジスタへの直接クエリ、Ethtool を使ったデバイスへのクエリがあります。利用可能なっ 3 つのオプションは以下のとおりです。

周波数の監視

ドライバーもしくはMII レジスタへのクエリの間隔時間 (ミリ秒単位)

リンクアップの遅延

アップとレポートされたリンクの使用を試みるまでの待ち時間 (ミリ秒単位) 。リンクが 「アップ」 とレポートされてからすぐに不当な ARP リクエストが失われた場合に、この遅延は使用できます。これが発生するのは、例えばスイッチ初期化などの間です。

リンクダウンの遅延

これまでアクティブだったリンクが 「ダウン」 とレポートされた際に、別のリンクに変更するまでの待ち時間 (ミリ秒単位) 。アタッチされたスイッチがバックアップモードに変更するまで比較的長い時間がかかる場合に、この遅延は使用できます。

ARP

アドレス解決プロトコル (Address Resolution Protocol, ARP) は、リンク層接続の動作具合を決定するために 1 つ以上のピアにプローブする際に使用されます。これは、送信開始時間および最終受信時間を提供しているデバイスドライバーに依存しています。以下の 2 つのオプションがあります。

周波数の監視

ARP リクエストを送信する間隔時間 (ミリ秒単位) 。

ARP ターゲット

ARP リクエストを送信する IP アドレスをコンマで区切ったもの。

8.3.7. VLAN 接続の確立

NetworkManager を使って、既存インターフェースを使用する VLAN を作成することができます。本書執筆の時点では、イーサネットデバイス上でのみ VLAN の作成が可能です。

手順8.11 新規 VLAN 接続の追加

VLAN 接続を設定するには、**ネットワーク接続** ウィンドウを開いてから、**追加** ボタンをクリックし、一覧から **VLAN** を選択します。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。
2. **追加** ボタンをクリックして選択一覧を開きます。**VLAN** を選択し、**作成** をクリックします。**VLAN 接続 *1* の編集** ウィンドウが表示されます。
3. **VLAN** タブで、VLAN 接続に使用する親インターフェースをドロップダウンリストから選びます。
4. VLAN ID を入力します。
5. VLAN インターフェース名を入力します。これは、作成される VLAN インターフェースの名前です。例えば、"eth0.1" や "vlan2" などです。(通常、この名前は親インターフェース名に "." と VLAN ID を加えたものか、"vlan" と VLAN ID を加えたものになります。)
6. 設定を確認して、**適用** ボタンをクリックします。
7. VLAN 特定の設定での編集は、以下の **VLAN タブの設定** を参照してください。

手順8.12 既存の VLAN 接続の編集

既存の VLAN 接続を編集するには以下の手順に従います。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。

2. 設定したい接続を選択して、 **編集** ボタンをクリックします。

3. **VLAN** タブを選びます。

4. 接続名、自動接続の動作、及び可用性のセッティングを設定します。

   編集 ダイアログ内の３つのセッティングはすべての接続タイプに共通です:

   - **接続名** — ネットワーク接続の名前を入力します。この名前は、**ネットワーク接続** ウィンドウの **VLAN** セクションでこの接続の表示に使用されます。
   - **自動接続する** — このボックスにチェックを入れると、この接続が利用可能な時に**NetworkManager** が自動接続します。詳細については 「ネットワークへの自動接続」 を参照してください。
   - **すべてのユーザーに利用可能** — このシステム上のすべてのユーザーに利用可能な接続を作成するには、このボックスにチェックを入れます。この設定を変更するには root 権限が必要な場合があります。詳細については「ユーザー接続とシステム接続」 を参照してください。

5. VLAN 特定の設定での編集は、以下の **VLAN タブの設定** を参照してください。

新規 (または修正した) 接続を保存して他の設定をする

VLAN 接続の編集が終了して、**適用** ボタンをクリックすると**NetworkManager** は即座にカスタマイズした設定を保存します。設定が正しければ、**NetworkManager** 通知スペースアプレットから新規またはカスタマイズした接続を選択して、その接続が確立されます。新規または変更した接続の使用に関する詳細については 「ネットワークへの接続」 を参照してください。

既存の接続をさらに設定をするには、**ネットワーク接続** ウィンドウ内でその接続を選択し、**編集** をクリックして **編集** ダイアログに戻ります。

そして、以下のいずれかの設定をします:

- 接続に対する IPv4 のセッティング: **IPv4 のセッティング** タブをクリックして 「IPv4 セッティングの設定」 に進みます。

VLAN タブの設定

新規の VLAN 接続を既に追加している場合 (手順に関しては、手順8.11「新規 VLAN 接続の追加」 を参照) 、**VLAN** タブを編集して、親インターフェースと VLAN ID を設定できます。

親インターフェース

ドロップダウンリストから以前に設定したインターフェースを選択できます。

VLAN ID

VLAN ネットワークのトラフィックのタグ付けに使用する ID 番号。

VLAN インターフェース名

作成される VLAN インターフェースの名前。例えば、"eth0.1" や "vlan2" など。

クローンした MAC アドレス

VLAN インターフェースの特定に使用する別の MAC アドレスをオプションで設定します。このアドレスを使って、この VLAN 上で送信されたパケットのソース MAC アドレスを変更することができます。

MTU

VLAN 接続で送信されるパケットに使用する最大転送単位 (MTU) のサイズをオプションで設定します。

8.3.8. IP-over-InfiniBand (IPoIB) 接続の確立

NetworkManager を使って InfiniBand 接続を作成することができます。

手順8.13 新規 InfiniBand 接続の追加

InfiniBand 接続を設定するには、**ネットワーク接続** ウィンドウを開いてから、**追加** ボタンをクリックし、一覧から **InfiniBand** を選択します。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。
2. **追加** ボタンをクリックして選択一覧を開きます。**InfiniBand** を選択し、**作成** をクリックします。**InfiniBand 接続 *1* の編集** ウィンドウが表示されます。
3. **InfiniBand** タブで、InfiniBand 接続に使用するトランスポートモードをドロップダウンリストから選びます。
4. InfiniBand MAC アドレスを入力します。
5. 設定を確認して、**適用** ボタンをクリックします。
6. InfiniBand 特定の設定での編集は、以下の **InfiniBand タブの設定** を参照してください。

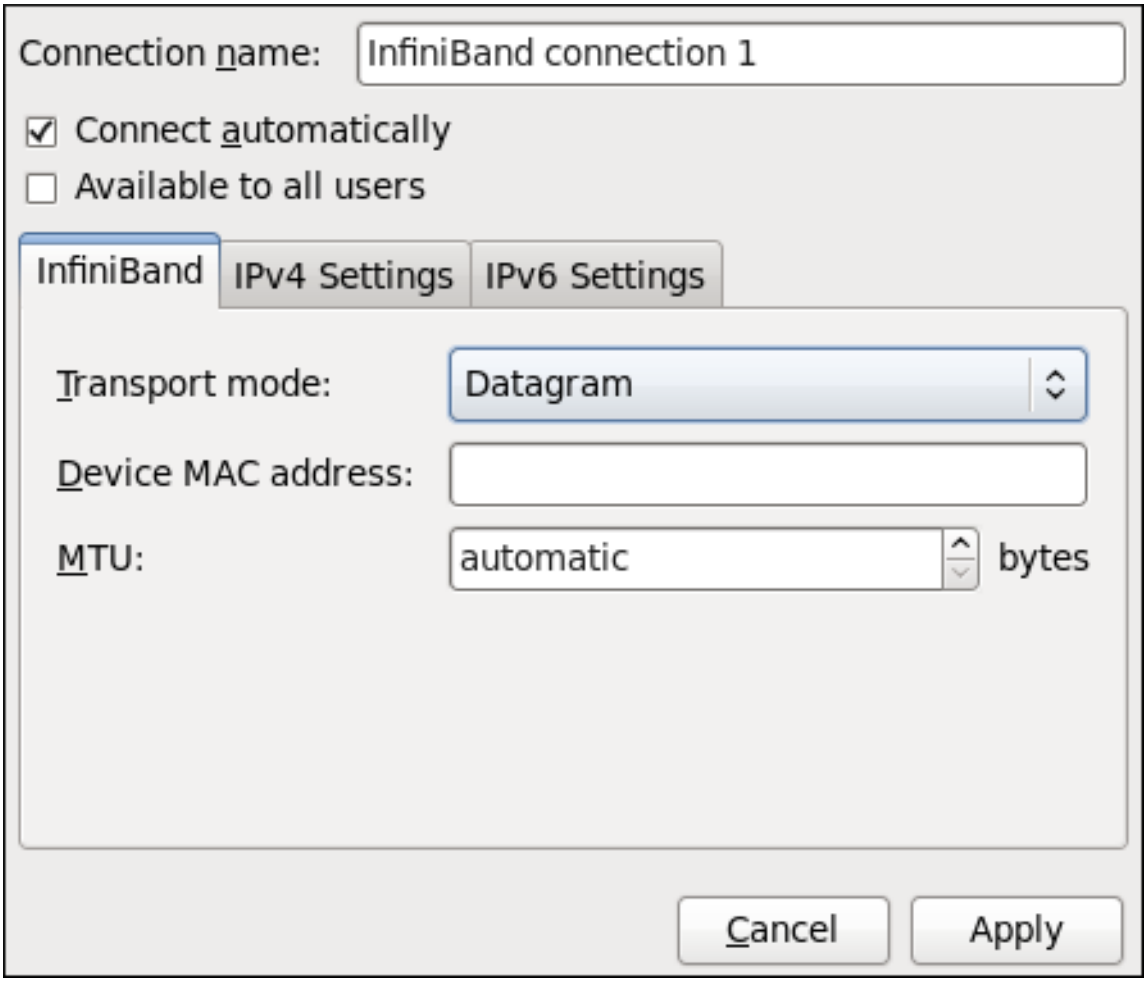


図8.15 新規作成した InfiniBand 接続 1 の編集

手順8.14 既存の InfiniBand 接続の編集

既存の InfiniBand 接続を編集するには以下の手順に従います。

1. 通知スペースにある **NetworkManager** アプレットのアイコンを右クリックして、**接続を編集する** をクリックすると、**ネットワーク接続** ウィンドウが表示されます。
2. 設定したい接続を選択して、 **編集** ボタンをクリックします。
3. **`InfiniBand`** タブを選択します。
4. 接続名、自動接続の動作、及び可用性のセッティングを設定します。

   編集 ダイアログ内の３つのセッティングはすべての接続タイプに共通です:

   - **接続名** — ネットワーク接続の名前を入力します。この名前は、**ネットワーク接続** ウィンドウの **`InfiniBand`** セクションでこの接続の表示に使用されます。
   - **自動接続する** — このボックスにチェックを入れると、この接続が利用可能な時に**NetworkManager** が自動接続します。詳細については 「ネットワークへの自動接続」を参照してください。
   - **すべてのユーザーに利用可能** — このシステム上のすべてのユーザーに利用可能な接続を作成するには、このボックスにチェックを入れます。この設定を変更するには root 権限が必要な場合があります。詳細については「ユーザー接続とシステム接続」 を参照してください。

5. InfiniBand 特定の設定での編集は、以下の **InfiniBand タブの設定** を参照してください。

新規 (または修正した) 接続を保存して他の設定をする
InfiniBand 接続の編集が終了して、**適用** ボタンをクリックすると**NetworkManager** は即座にカスタマイズした設定を保存します。設定が正しければ、**NetworkManager** 通知スペースアプレットから新規またはカスタマイズした接続を選択して、その接続が確立されます。新規または変更した接続の使用に関する詳細については 「ネットワークへの接続」 を参照してください。

既存の接続をさらに設定をするには、**ネットワーク接続** ウィンドウ内でその接続を選択し、**編集** をクリックして **編集** ダイアログに戻ります。

そして、以下のいずれかの設定をします:

- 接続に対する IPv4 のセッティング: **IPv4 のセッティング** タブをクリックして 「IPv4 セッティングの設定」 へ進むか、または;
- 接続に対する IPv6 のセッティング: **IPv6 のセッティング** タブをクリックして 「IPv6 セッティングの設定」 に進みます。

InfiniBand タブの設定
新規の InfiniBand 接続を既に追加している場合 (手順に関しては、手順8.13「新規 InfiniBand 接続の追加」 を参照) 、**InfiniBand** タブを編集して、親インターフェースと InfiniBand ID を設定できます。

トランスポートモード

ドロップダウンリストから、データグラムまたは接続モードを選択できます。他の IPoIB ネットワークで使用しているモードと同じものを選びます。

Device MAC アドレス

InfiniBand ネットワークのトラフィックで使用される InfiniBand 対応デバイスの MAC アドレスです。InfiniBand ハードウェアがインストールされていれば、このハードウェアのアドレスフィールドは事前に記入されます。

MTU

InfiniBand 接続で送信されるパケットに使用する最大転送単位 (MTU) のサイズをオプションで設定します。

8.3.9. 接続セッティングの設定

8.3.9.1. 802.1x セキュリティの設定

802.1x セキュリティとは、PNAC (port-based Network Access Control) 用の IEEE 基準の名前です。簡単に言えば、802.1x セキュリティは、物理ネットワークから *論理ネットワーク* を定義する手段です。論理ネットワークに参加を希望するすべてのクライアントは、正しい 802.1x 認証方法を使用してサーバー (例えば、ルーター) で認証を受けなければなりません。

802.1x セキュリティは最も多く無線ネットワーク (WLANs) の保護に関連付けられていますが、ネットワーク (LAN) へのアクセスを持つ侵入者を阻止するためにも使用できます。過去に於いて、DHCP サーバーは認証の無いユーザーに IP アドレスをリースするように設定されていましたが、各種理由でこれは非現実的で安全ではないため、今ではもう推奨できません。その代わりにポートベースの認証を介して論理的に安全なネットワークを確実にするために 802.1x セキュリティが使用されます。

802.1x は、WLAN と LAN のアクセス制御用のフレームワークを提供して、EAP (Extensible Authentication Protocol) タイプの 1 つを運搬する包装として機能します。EAP のタイプとは、ネットワーク上で WLAN セキュリティの達成方法を定義するプロトコルです。

有線、あるいは無線接続タイプ用に 802.1x セキュリティを設定するには、**ネットワーク接続** ウィンドウを開いて (「新規接続の設定と既存接続の編集」参照) 該当する手順に従います:

手順8.15 有線接続には...

1. **追加** ボタンをクリックして 802.1x セキュリティを設定する新規のネットワーク接続を選択してから **作成** をクリックするか、既存の接続を選択して **編集** をクリックします。
2. **802.1x セキュリティ** タブを選択して、そこにある **この接続に 802.1x セキュリティを使用する** チェックボックスにチェックを入れることで設定を有効にします。
3. 「TLS (Transport Layer Security) セッティングの設定」 に進みます。

手順8.16 無線接続には...

1. **追加** ボタンをクリックして 802.1x セキュリティを設定する新規のネットワーク接続を選択してから **作成** をクリックするか、既存の接続を選択して **編集** をクリックします。
2. **無線セキュリティ** タブを選択します。
3. **セキュリティ** ドロップダウンをクリックして以下のセキュリティメソッドの 1 つを選択します:**LEAP**、**動的 WEP (802.1x)**、あるいは **WPA & WPA2 Enterprise**。
4. どの EAP タイプが **セキュリティ** ドロップダウン内の選択肢に対応しているかについての説明は、「TLS (Transport Layer Security) セッティングの設定」 を参照してください。

8.3.9.1.1. TLS (Transport Layer Security) セッティングの設定

TLS (トランスポートレイヤーセキュリティ) では、クライアントとサーバーは TLS プロトコルを使用して相互に認証し合います。サーバーはデジタル証明書を維持していることを示して、クライアントはそのクライアント側の証明書を使用して自身の識別を証明することで、キー情報が交換されます。認証が完了すると、TLS トンネルの使用は終了します。その代わりにクライアントとサーバーは交換したキーで、AES, TKIP または WEP を使用してデータを暗号化します。

認証を希望する全クライアントに証明書が配布される必要があるということは、EAP-TLS 認証のメソッドが非常に強力であることを意味しますが、セットアップはより複雑になります。TLS セキュリティを使用すると、証明書を管理するための PKI (パブリックキーインフラストラクチャ) の諸経費が必要になります。TLS セキュリティを使用する利点は、パスワードが危険にさらされても (W)LAN へのアクセスが許可されないことです。侵入者は、認証するクライアントのプライベートキーにもアクセスを必要とします。

Network Manger は、対応する TLS のバージョンを決定しません。Network Manager は、ユーザーが入力するパラメーターを集め、手順を処理するデーモンである wpa_supplicant に渡します。このデーモンは、OpenSSL を使って TLS トンネルを確立します。OpenSSL は、SSL/TLS プロトコルバージョンを処理します。このバージョンには、両方の末端でサポートされる最高のバージョンが使用されます。

識別

ユーザー名やログイン名などの、EAP 認証メソッドの識別ストリングです。

ユーザーの証明書

クリックしてブラウズし、ユーザーの証明書を選択します。

CA 証明書

クリックしてブラウズし、認証局 (CA) の証明書を選択します。

プライベートキー

クリックしてブラウズし、ユーザーのプライベートキーファイルを選択します。

プライベートキーのパスワード

ユーザーのプライベートキーに対応するユーザーのパスワードを入力します。

8.3.9.1.2. トンネル化した TLS セッティングの設定

匿名識別

この値は、非暗号化 ID として使用されます。

CA 証明書

クリックしてブラウズし、認証局 (CA) の証明書を選択します。

内部認証

PAP — パスワード認証プロトコル

MSCHAP — チャレンジ ハンドシェイク認証プロトコル

MSCHAPv2 — Microsoft チャレンジハンドシェイク認証プロトコルバージョン 2

CHAP — チャレンジ ハンドシェイク認証プロトコル

ユーザー名

認証プロセスで使用するユーザー名を入力します。

パスワード

認証プロセスで使用するパスワードを入力します。

8.3.9.1.3. 保護された EAP (PEAP) セッティングの設定

匿名識別

この値は、非暗号化 ID として使用されます。

CA 証明書

クリックしてブラウズし、認証局 (CA) の証明書を選択します。

PEAP バージョン

使用する、保護された EAP のバージョン。Automatic、0、1 のいずれか。

内部認証

MSCHAPv2 — Microsoft チャレンジハンドシェイク認証プロトコルバージョン 2

MD5 — メッセージダイジェスト 5、暗号化ハッシュ関数。

GTC — Generic Token Card

ユーザー名

認証プロセスで使用するユーザー名を入力します。

パスワード

認証プロセスで使用するパスワードを入力します。

8.3.9.2. 無線セキュリティの設定

セキュリティ

なし — Wi-Fi 接続を暗号化しません。

WEP 40/128-bit キー — IEEE 802.11 標準からの Wired Equivalent Privacy (WEP) 。単一の事前共有キー (PSK) を使用。

WEP 128-bit パスフレーズ — パスフレーズの MD5 ハッシュを使用して WEP キーを引き出します。

LEAP — Cisco Systems の Lightweight Extensible Authentication Protocol 。

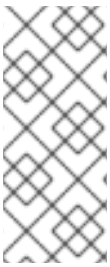
動的 WEP (802.1x) — WEP キーは動的に変更されます。

WPA & WPA2 パーソナル — IEEE 802.11i 標準からの Wi-Fi Protected Access (WPA) 。WEP に代わるもの。802.11i-2004 標準からの Wi-Fi Protected Access II (WPA2) 。パーソナルモードは、事前共有キー (WPA-PSK) を使用。

WPA & WPA2 エンタープライズ — RADUIS 認証サーバーで使用する WPA で、IEEE 802.1x ネットワークアクセスコントロールを提供。

パスワード

認証プロセスで使用するパスワードを入力します。



注記

WPA および WPA2 (個人およびエンタープライズ) の場合、自動、WPA、WPA2 を選択するオプションが加えられます。このオプションは、WPA と WPA2 の両方が利用可能なアクセスポイントでの使用を念頭に置いています。両方のプロトコル間でのローミングを防止したい場合は、どちらかのプロトコルを選択します。同一アクセスポイントでの WPA と WPA2 間のローミングは、サービスの損失につながる恐れがあります。

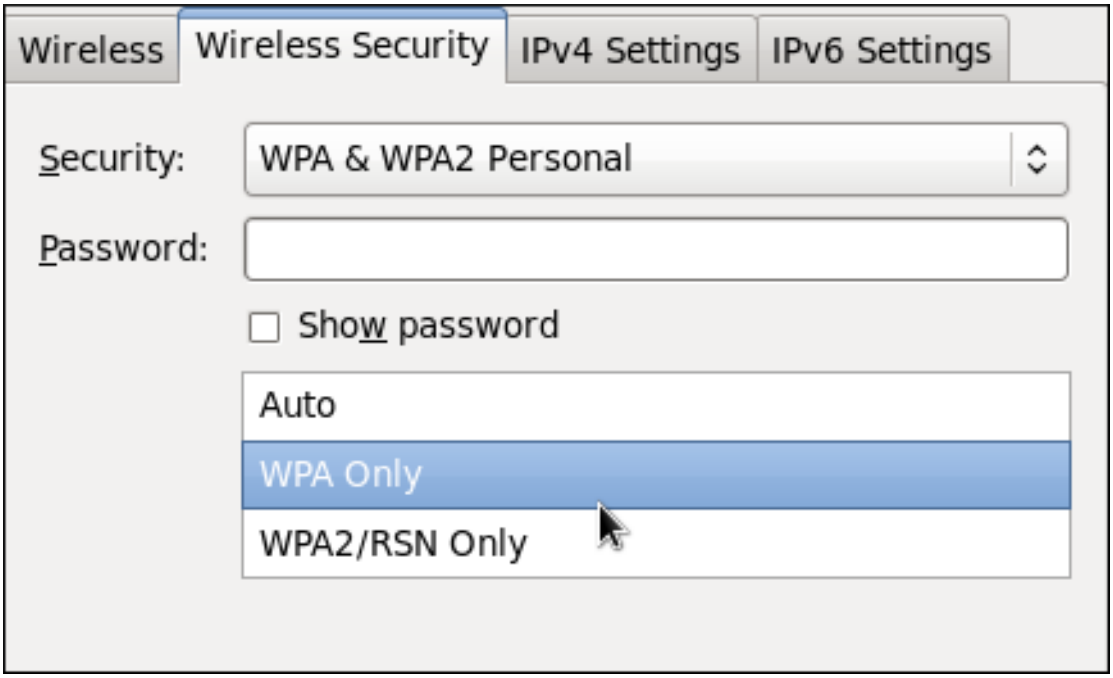


図8.16 無線セキュリティタブの編集および WPA プロトコルの選択

8.3.9.3. PPP (Point-to-Point) セッティングの設定

メソッドの設定

`MPPE (point-to-point encryption)` を使用

Microsoft Point-To-Point Encryption プロトコル (RFC 3078)

`BSD` データ圧縮を許可

PPP BSD Compression プロトコル (RFC 1977)

`Deflate` データ圧縮を許可

PPP Deflate プロトコル (RFC 1979)

`TCP` ヘッダー圧縮を使用

低速シリアルリンク用に TCP/IP ヘッダーを圧縮 (RFC 1144)

`PPP` エコーパケットを送信

ループバックテスト用の LCP エコー要求およびエコー応答コード (RFC 1661)

8.3.9.4. IPv4 セッティングの設定

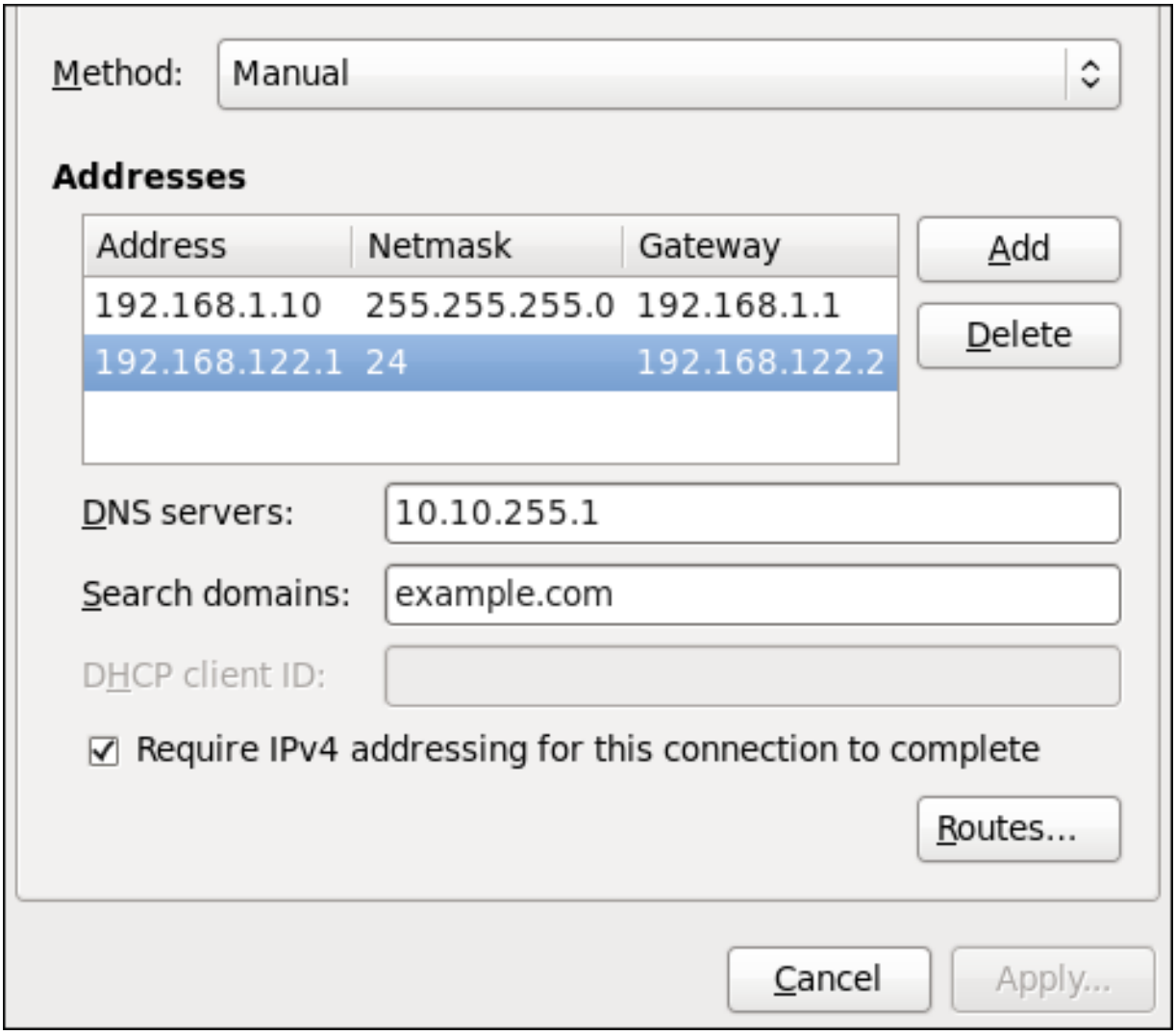


図8.17 IPv4 セッティングタブの編集

IPv4 のセッティング タブではインターネットへの接続メソッドを設定して、必要に応じて IP アドレス、ルートおよび DNS 情報を入力することができます。**IPv4 のセッティング** タブは、以下の接続タイプの 1 つを作成して修正する時に利用可能になります。有線、無線、モバイルブロードバンド、VPN、 DSL。

DHCP を使用して DHCP サーバーから動的に IP アドレスを取得する場合は、**方式** を単に **自動 (DHCP)** にセットするだけです。

方式のセッティング

接続タイプ別で利用可能な IPv4 方式

方式 ドロップダウンメニューをクリックすると、設定している接続のタイプによっては、以下の IPv4 接続方式の 1 つを選択することができます。関連のある接続タイプ群に応じてすべての方針がここに一覧表示されています。

方式

自動 (DHCP) — 接続しているネットワークが IP アドレスの割り当てにDHCP サーバーを使用する場合は、このオプションを選択します。**DHCP クライアント ID** フィールドに記入する必要はありません。

自動 (DHCP) アドレスのみ — 接続しているネットワークが IP アドレスの割り当てにDHCP サーバーを使用しているが、DNS サーバーを手動で割り当てたい場合は、このオプションを選択します。

ローカルのみリンク（Link-Local Only） — 接続しているネットワークに DHCP サーバーがなく、IP アドレスを手動で割り当てたくない場合に、このオプションを選択します。RFC 3927 にしたがってランダムなアドレスが選択されます。

他のコンピュータへ共有 — 設定しているインターフェースがインターネットもしくは WAN 接続の共有用である場合は、このオプションを選択します。

有線、無線、及び DSL 接続の方式

手動 — 接続しているネットワークに DHCP サーバーがなく、IP アドレスを手動で割り当てたい場合に、このオプションを選択します。

モバイルブロードバンド接続の方式

自動 (PPP) — 接続しているネットワークが IP アドレスの割り当てにDHCP サーバーを使用する場合は、このオプションを選択します。

自動 (PPP) アドレスのみ — 接続しているネットワークが IP アドレスの割り当てにDHCP サーバーを使用しているが、DNS サーバーを手動で割り当てたい場合は、このオプションを選択します。

VPN 接続の方式

自動 (VPN) — 接続しているネットワークが IP アドレスの割り当てにDHCP サーバーを使用する場合は、このオプションを選択します。

自動 (VPN) アドレスのみ — 接続しているネットワークが IP アドレスの割り当てにDHCP サーバーを使用しているが、DNS サーバーを手動で割り当てたい場合は、このオプションを選択します。

DSL 接続の方式

自動 (PPPoE) — 接続しているネットワークが IP アドレスの割り当てにDHCP サーバーを使用する場合は、このオプションを選択します。

自動 (PPPoE) アドレスのみ — 接続しているネットワークが IP アドレスの割り当てにDHCP サーバーを使用しているが、DNS サーバーを手動で割り当てたい場合は、このオプションを選択します。

ネットワーク接続の静的ルート設定に関する詳細は、「ルートの設定」 を参照してください。

8.3.9.5. IPv6 セッティングの設定

方式

無視する — IPv6 セッティングを無効にしたい場合は、このオプションを選びます。

自動 — 接続しているネットワークが IP アドレスの割り当てにDHCP サーバーを使用する場合は、このオプションを選択します。

自動、アドレスのみ — 接続しているネットワークが IP アドレスの割り当てにDHCP サーバーを使用しているが、DNS サーバーを手動で割り当てたい場合は、このオプションを選択します。

手動 — 接続しているネットワークに DHCP サーバーがなく、IP アドレスを手動で割り当てたい場合に、このオプションを選択します。

ローカルのみリンク（Link-Local Only） — 接続しているネットワークに DHCP サーバーがなく、IP アドレスを手動で割り当てたくない場合に、このオプションを選択します。RFC 4862 にしたがってランダムなアドレスが選択されます。

他のコンピュータへ共有 — 設定しているインターフェースがインターネットもしくは WAN 接続の共有用である場合は、このオプションを選択します。

アドレス

DNS サーバー — コンマで区切られた DNS サーバーのリストを入力します。

ドメインを検索 — コンマで区切られたドメインコントローラーのリストを入力します。

ネットワーク接続の静的ルート設定に関する詳細は、「ルートの設定」 を参照してください。

8.3.9.6. ルートの設定

ホストのルーティング表は、ネットワークに直接接続されているルートで自動的に設定されます。このルートは、「アップ」 の状態の時にネットワークインターフェースが監視することで学習されます。このセクションでは、VPN や 専用線などの媒介ネットワークや接続を移動してアクセスできるネットワークやホストへの静的ルートの入力方法について説明します。

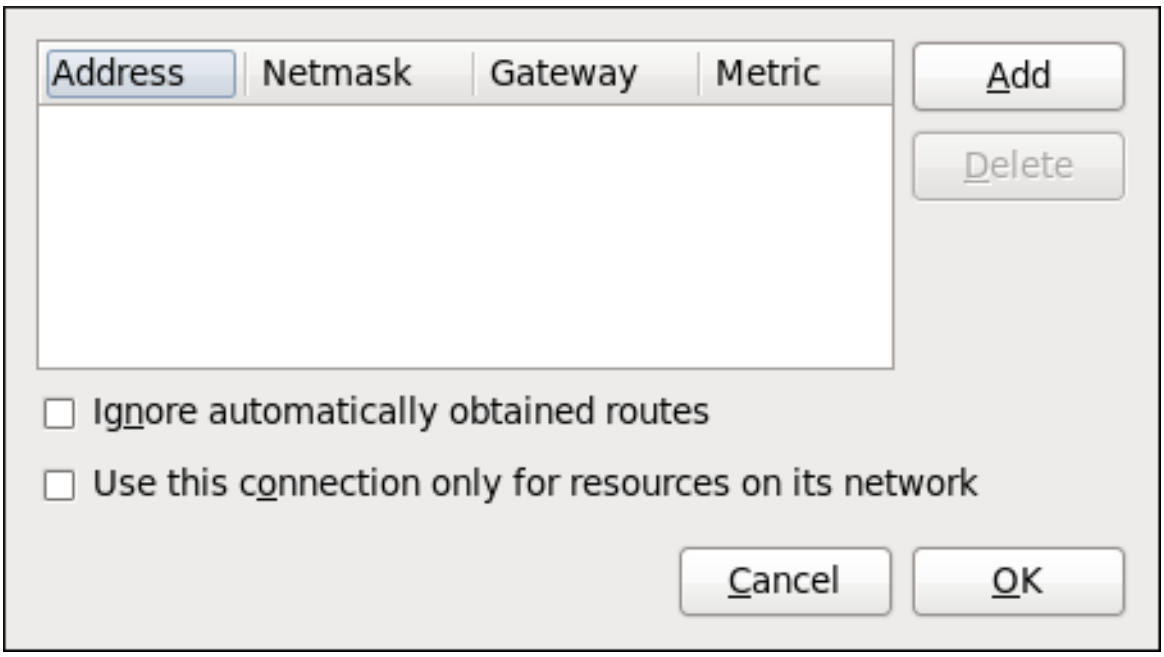


図8.18 静的ネットワークルートの設定

アドレス

アドレス — ネットワーク、サブネット、ホストの IP アドレスです。

ネットマスク — ネットマスクまたは入力した IP アドレスのプレフィックスの長さ

ゲートウェイ — ネットワーク、サブネット、ホストにつながるゲートウェイの IP アドレスです。

メトリック — ネットワークコスト、つまりこのルートに与える優先値。高い値よりも低い値の方が望ましい。

自動取得したルートを無視

接続に手動入力したルートのみを使用する場合は、このボックスにチェックを入れます。

ネットワーク上のリソースにのみこの接続を使用

この接続がデフォルトルートにならないようにするには、このチェックボックスを選択します。よくある例は、ヘッドオフィスへの接続が VPN や専用線で、インターネットのトラフィックにこの接続を使用してほしくない場合です。このオプションを選択すると、この接続で自動的に学習されたルートを使用することが明確なトラフィックか、手動で入力されたトラフィックのみがこの接続を経由します。

8.4. NETWORKMANAGER のアーキテクチャ

http://live.gnome.org/NetworkManagerConfiguration をご覧下さい

第9章 ネットワークインターフェース

Red Hat Enterprise Linux では、ネットワーク通信はすべてシステムに接続された、設定済みのソフトウェア *インターフェース* と *物理ネットワークデバイス* の間で発生します。

ネットワークインターフェース用の設定ファイルは、**/etc/sysconfig/network-scripts/** ディレクトリにあります。これらのネットワークインターフェースをアクティベート、非アクティベートするスクリプトも同じディレクトリにあります。システムによっては、インターフェースファイルの数とタイプは異なる場合がありますが、このディレクトリに存在するファイルには３つのカテゴリーがあります。

1. *インターフェース設定ファイル*
2. *インターフェース制御スクリプト*
3. *ネットワーク関数ファイル*

これらの各カテゴリーのファイルが一体となって機能し、様々なネットワークデバイスを有効にします。

本章では、これらのファイルの関係と使用方法について詳しく説明していきます。

9.1. ネットワーク設定ファイル

インターフェース設定ファイルについて詳しく見ていく前に、まずネットワーク設定に使用する主要な設定ファイルについて項目別に説明しましょう。ネットワークスタックを設定するにあたってこうしたファイルの役割を理解しておくと、Red Hat Enterprise Linux システムをカスタマイズする時に役立ちます。

主要なネットワーク設定ファイルは以下の通りです。

/etc/hosts

このファイルの主要目的は、他の方法では解決できないホスト名を解決することです。DNS サーバーのない、小規模ネットワーク上のホスト名を解決することも可能です。コンピュータが接続されているネットワークのタイプにかかわらず、このファイルには、**localhost.localdomain** として、ループバックデバイスの IP アドレス (**127.0.0.1**) を指定する行が含まれているはずです。詳しくは、**hosts**(5) の man ページを参照してください。

/etc/resolv.conf

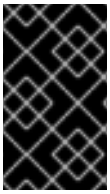
このファイルは、DNS サーバーの IP アドレスと検索ドメインを指定します。他の動作を行うように設定されていない限り、ネットワーク初期化スクリプトはそうしたデータをこのファイルに入れます。このファイルの詳細情報は、**resolv.conf**(5) の man ページを参照してください。

/etc/sysconfig/network

このファイルは、すべてのネットワークインターフェースのルーティングとホストの情報を指定します。グローバルな影響があり、インターフェース固有でない指示文の格納に使われます。このファイルとそれが使用できる指示文の詳細は、「/etc/sysconfig/network」 を参照してください。

/etc/sysconfig/network-scripts/ifcfg-*interface-name*

各ネットワークインターフェースには、対応するインターフェース設定スクリプトがあります。これらの各ファイルは、特定のネットワークインターフェースに固有の情報を提供します。このタイプのファイルとそれが使用できる指示文の詳細については、「インターフェース設定ファイル」 を参照してください。



重要

ネットワークインターフェースの名前はハードウェアのタイプによって異なるかも知れません。詳細については、付録A *一貫したネットワークデバイスの命名* を参照して下さい。

警告

/etc/sysconfig/networking/ ディレクトリは、非推奨の **Network Administration Tool** (**system-config-network**) が使用し、そのコンテンツは手動で編集 **すべきではありません**。設定を削除してしまうリスクがあるため、ネットワーク設定には 1 つの方法だけを使用することを強く推奨します。グラフィカル設定ツールを使用したネットワークインターフェイスの設定に関する情報は、8章*NetworkManager* を参照して下さい。

9.2. インターフェース設定ファイル

インターフェース設定ファイルは、個々のネットワークデバイスのソフトウェアインターフェースを制御します。システムは、ブート時にこれらのファイルを使用して、どのインターフェースをアクティブにして、どのように設定するかを決定します。これらのファイルは、通常 **ifcfg-*name*** と命名されます。*name* は設定ファイルが制御するデバイスの名前を指します。

9.2.1. イーサネットインターフェース

etc/sysconfig/network-scripts/ifcfg-eth0 は、最も一般的なインターフェースファイルの 1 つです。これは、システムの一番目のイーサネットの *ネットワークインターフェースカード* または NIC を制御します。複数の NIC を持つシステムには、複数の **ifcfg-eth*X*** ファイルが存在します (*X* は特定のインターフェースに対応した一意の番号です)。デバイスにはそれぞれ独自の設定ファイルがあるため、管理者は、各インターフェースがどう機能するか個別に制御できます。

固定 IP アドレスを使用するシステム用の **ifcfg-eth0** ファイルのサンプルは以下のとおりです:

```
DEVICE=eth0
BOOTPROTO=none
ONBOOT=yes
NETMASK=255.255.255.0
IPADDR=10.0.1.27
USERCTL=no
```

インターフェース設定ファイルで必要な値は、他の値に基づいて変化します。例えば、DHCP を使用するインターフェース用の **ifcfg-eth0** ファイルは、IP 情報が DHCP サーバーにより提供されるため異なって見えます:

```
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
```

NetworkManager は、様々なネットワークインターフェースの設定ファイルに変更を加える簡単な方法です (このツールの使用に関する詳しい説明は 8章*NetworkManager* を参照してください)。

ただし、任意のネットワークインターフェースの設定ファイルを手動で編集することも可能です。

以下は、イーサネットインターフェースの設定ファイルにある設定可能なパラメータの一覧です:

BONDING_OPTS=*パラメータ*

ボンディングデバイス用の設定パラメータを設定し、**/etc/sysconfig/network-scripts/ifcfg-bond*N*** で使用されます (「チャンネルボンディングインターフェース」 を参照)。これらのパラメータは、**/sys/class/net/*bonding device*/bonding** でボンディングデバイスに使用されているパラメータ、及び **bonding モジュール指示文** に記載のボンディングドライバー用のモジュールパラメータと全く同じです。

この設定方法を用いると、複数のボンディングデバイスを異なる設定にできます。すべてのボンディングオプションは、**ifcfg-*name*** の **BONDING_OPTS** 指示文の後ろに置くことが強く推奨されます。**/etc/modprobe.d/*bonding*.conf** または廃棄になった **/etc/modprobe.conf** ファイルでは、ボンディングデバイスのオプションは指定 **しないでください**。

BOOTPROTO=*プロトコル*

プロトコル が以下のいずれかの場合です:

- **none** — ブート時プロトコルを使用しません。
- **bootp** — BOOTP プロトコルを使用する必要があります。
- **dhcp** — DHCP プロトコルを使用する必要があります。

BROADCAST=*アドレス*

アドレス はブロードキャストアドレスです。この指示文は、**ifcalc** で自動的に値が算出されるため破棄されています。

DEVICE=*名前*

名前 は物理デバイス名です (動的に割り振られる PPP デバイスを除きます。その場合は **論理名** です)。

DHCP_HOSTNAME=*名前*

名前 は DHCP サーバーに送信される短いホスト名です。クライアントが IP アドレスを受信する前にホスト名を指定するように DHCP サーバーが 要求する場合に限り、このオプションを使用します。

DNS*{1,2}*=*アドレス*

PEERDNS 指示文が **yes** にセットされている場合、*アドレス* は **/etc/resolv.conf** に配置されるネームサーバーのアドレスです 。

ETHTOOL_OPTS=*オプション*

オプション は **ethtool** がサポートする任意のデバイス固有オプションです。例えば 100Mb、フルデュプレックスを強制したい場合は以下のようにします :

```
ETHTOOL_OPTS="autoneg off speed 100 duplex full"
```

カスタムの initscript の代わりに、**ETHTOOL_OPTS** を使用して、インターフェースのスピードとデュプレックス設定を行います。ネットワーク initscript 外で実行されるカスタム initscript により、ブート後のネットワークサービスの再起動中に予期できない結果をもたらします。

重要

スピードまたはデュプレックスの設定を変更する場合は、ほとんどの場合 **autoneg off** オプションを使用してオートネゴシエーションを無効にする必要があります。オプションエントリは順序に依存するため、このオプションは最初に記載する必要があります。

Ethtool オプションの詳細については、「Ethtool」 を参照して下さい。

HOTPLUG=*回答*

回答 が以下のいずれかの場合です:

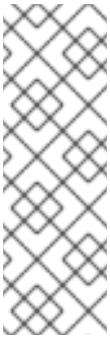
- **yes** — ホットプラグになっている時にこのデバイスをアクティブにします (これがデフォルトのオプション)。
- **no** — ホットプラグの時でもこのデバイスをアクティブに **しません** 。

HOTPLUG=no オプションを使用すると、ボンディングカーネルモジュールがロードされる際にチャンネルボンディングインターフェイスがアクティブになるのを阻止できます。

チャンネルボンディングインターフェースの詳細は、「チャンネルボンディングインターフェース」 を参照してください。

HWADDR=*MAC-アドレス*

MAC-アドレス は、*AA:BB:CC:DD:EE:FF* 形式のイーサネットデバイスのハードウェアアドレスです。複数の NIC が格納されたマシンでこの指示文を使用して、各 NIC モジュールの設定されているロード順序に関わらず、正しいデバイス名がインターフェースに割り当てられるようにする必要があります。この指示文は、**MACADDR** と併用されるべき **ではありません**。



注記

- 永続的なデバイス名は **/etc/udev/rules.d/70-persistent-net.rules** が処理します。
- **HWADDR** は、System z ネットワークデバイスと使用しないでください。
- 『Red Hat Enterprise Linux 6 Installation Guide』 のセクション 25.3.3 「サブチャンネルのマッピングとネットワークデバイス名」を参照してください。

IPADDR=*アドレス*

アドレス は IP アドレスです。

LINKDELAY=*時間*

時間 は、デバイス設定の前にリンクが交渉するための待機時間の秒数です。

MACADDR=*MAC-アドレス*

MAC-アドレス は *AA:BB:CC:DD:EE:FF* 形式のイーサネットデバイスのハードウェアアドレスです。

この指示文を使用して、MAC アドレスをインターフェースに割り当て、物理 NIC に割り当てられた MAC アドレスを上書きします。この指示文は **HWADDR** 指示文と併用 **しない** ことが推奨されます。

MASTER=*ボンドインターフェイス*

ボンドインターフェイス はイーサネットインターフェースがリンクされたチャンネルボンディングインターフェースです。

この指示文は、**SLAVE** 指示文と併用されます。

チャンネルボンディングインターフェースに関する詳細は 「チャンネルボンディングインターフェース」 を参照してください。

NETMASK=*マスク*

マスク はネットマスクの値です。

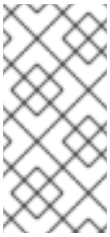
NETWORK=*アドレス*

アドレス はネットワークアドレスです。値は **ipcalc** で自動的に算出されるため、この指示文は廃棄されています。

NM_CONTROLLED=*回答*

回答 が以下のいずれかの場合です:

- **yes** — NetworkManager によるこのデバイスの設定が許可されます。これはデフォルトの動作で、省略できます。
- **no** — NetworkManager によるこのデバイスの設定は許可されません。



注記

NM_CONTROLLED 指示文は、Red Hat Enterpise Linux 6.3 以降で **/etc/sysconfig/network** の **NM_BOND_VLAN_ENABLED** 指示文に依存しています。この指示が存在し、**yes** か **y**、**true** の場合のみ、NetworkManager はボンディングおよび VLAN インターフェースを検出し、管理します。

ONBOOT=*回答*

回答 が以下のいずれかの場合です:

- **yes** — このデバイスはブート時にアクティブにされる必要があります。
- **no** — このデバイスはブート時にアクティブにされる必要はありません。

PEERDNS=*回答*

回答 が以下のいずれかの場合です:

- **yes** — DNS 指示文が設定されている場合は **/etc/resolv.conf** を変更します。DHCP を使用する場合は、**yes** がデフォルトです。
- **no** — **/etc/resolv.conf** を変更しません。

SLAVE=*回答*

回答 が以下のいずれかの場合です:

- **yes** — このデバイスは、**MASTER** 指示文で指定されているチャンネルボンディングインターフェースにより制御されます。
- **no** — このデバイスは、**MASTER** 指示文で指定されているチャンネルボンディングインターフェースにより制御 **されません**。

この指示文は、**MASTER** 指示文と併用されます。

チャンネルボンディングインターフェースの詳細は、「チャンネルボンディングインターフェース」 を参照してください。

SRCADDR=*アドレス*

アドレス は、送信パケットのために指定されたソース IP アドレスです。

USERCTL=*回答*

回答 が以下のいずれかの場合です:

- **yes** — **root** でないユーザーもこのデバイスを制御できます。
- **no** — **root** でないユーザーはこのデバイスを制御できません。

9.2.2. System z 上の Linux 特定の ifcfg オプション

SUBCHANNELS=<*read_device_bus_id*>, <*write_device_bus_id*>, <*data_device_bus_id*>

<*read_device_bus_id*><*write_device_bus_id*>, <*data_device_bus_id*> は、ネットワークデバイスを示す 3 つのデバイスバス ID です。

PORTNAME=*myname;*

myname は、オープンシステムアダプター (OSA) の portname または LAN チャンネルステーション (LCS) の portnumber になります。

CTCPROT=*回答*

回答 が以下のいずれかの場合です:

- **0** — 互換モード、仮想マシンの TCP/IP (IBM S/390 や IBM System z オペレーティングシステムではない、Linux 以外のピアで使用)。これはデフォルトモードです。
- **1** — 拡張モード、Linux-to-Linux ピアーズで使用。
- **3** — S/390 および IBM System z オペレーティングシステムの互換モード。

この指示文は、NETTYPE 指示文と合わせて使われます。NETTYPE='ctc' の CTC プロトコルを指定します。デフォルト値は 0 です。

OPTION='*回答*'

'*回答*' は、有効な sysfs 属性とその値の引用符で囲まれた文字列です。Red Hat Enterprise Linux インストーラーは現在、これを使用して QETH デバイスのレイヤーモード (layer2) と相対ポート番号 (portno) を設定します。以下に例を挙げます。

```
OPTIONS='layer2=1 portno=0'
```

9.2.3. System z 上の Linux の必須 ifcfg オプション

NETTYPE=*回答*

回答 が以下のいずれかの場合です:

- **ctc** —チャンネルツーチャンネル通信。ポイントツーポイントの場合は、TCP/IP または TTY 。
- **lcs** — LAN チャンネルステーション (LCS) 。
- **qeth** — QETH (QDIO イーサネット) 。これはデフォルトのネットワークインターフェースです。実機または仮想の OSA カードおよび HiperSockets デバイスのサポートで推奨されるインストール方法です。

9.2.4. Ethtool

Ethtool は、*Network Interface Cards* (NICs) 設定の際のユーティリティです。このユーティリティを使うと、多くのネットワークデバイス、特にイーサネットデバイスのスピードやポート、オートネゴシエーション、PCI の場所、チェックサムオフロードといった設定のクエリや変更が可能になります。

以下に、使用頻度の高い Ethtool コマンドとあまり知られていない有用なコマンドを挙げておきます。コマンドの全一覧は、**ethtool -h** で表示されます。詳細なリストと説明は、man ページ **ethtool(8)** を参照してください。以下の最初の 2 例は情報クエリで、コマンドの異なるフォーマットの使い方を示しています。

まず、コマンド構文を見ていきます。

```
ethtool [option...] devname
```

option はなしかさらなるオプションで、*devname* は、eth0 や em1 などの使用中の Network Interface Card (NIC) です。

ethtool

オプションがデバイス名のみの **ethtool** コマンドは、ネットワークデバイスに NIC やドライバー統計をクエリする際に使用されます。以下の形式になります。

```
ethtool -S, --statistics devname
```

where *devname* は使用中の NIC です。例えば、eth0 や em1 などです。

値の中には root でコマンドを実行しないと取得できないものもあります。以下は、root でコマンドを実行した際の出力例です。

```
~]# ethtool em1
Settings for em1:
 Supported ports: [ TP ]
 Supported link modes:   10baseT/Half 10baseT/Full
                         100baseT/Half 100baseT/Full
                         1000baseT/Full
 Supported pause frame use: No
 Supports auto-negotiation: Yes
 Advertised link modes:  10baseT/Half 10baseT/Full
```

```
                          100baseT/Half 100baseT/Full
                          1000baseT/Full
 Advertised pause frame use: No
 Advertised auto-negotiation: Yes
 Speed: 1000Mb/s
 Duplex: Full
 Port: Twisted Pair
 PHYAD: 2
 Transceiver: internal
 Auto-negotiation: on
 MDI-X: on
 Supports Wake-on: pumbg
 Wake-on: g
 Current message level: 0x00000007 (7)
          drv probe link
 Link detected: yes
```

長短どちらかの形式の引数を使用して以下のコマンドを発行し、特定のネットワークデバイスに関連するドライバー情報をクエリします。

```
ethtool -i, --driver devname
```

devname は、eth0 や em1 などの使用中の Network Interface Card (NIC) です。

以下は出力例です。

```
~]$ ethtool -i em1
driver: e1000e
version: 2.0.0-k
firmware-version: 0.13-3
bus-info: 0000:00:19.0
supports-statistics: yes
supports-test: yes
supports-eeprom-access: yes
supports-register-dump: yes
```

ここではデバイスのクエリや識別、リセットに使用するコマンドリストを説明します。形式は通常の - 短いものと -- 長いものになります。

--statistics

--statistics または **-S** は、特定のネットワークデバイスに NIC やドライバー統計をクエリする際に使用されます。以下の形式になります。

```
ethtool -S, --statistics devname
```

devname は NIC です。

--identify

--identify または **-p** オプションは、オペレーターが目視でアダプターを簡単に特定できるようにする目的のアダプター固有のアクションを開始します。通常これで行われるのは、特定のネットワークポート上で 1 つ以上の LED の点滅です。以下の形式になります。

```
-p, --identify devname integer
```

integer アクションを実行する時間の長さです (秒単位) 。

devname は NIC です。

--test

--test または **-t** オプションは、ネットワークインターフェースカード上でテストを実行する際に使用します。以下の形式になります。

```
-t, --test word devname
```

word は以下のいずれかになります。

- **offline** — 総合的なテストを実行します。サービスが一時中断されます。
- **online** — 短縮されたテストを実行します。サービスは中断されません。
- **external_lb** — ループバックケーブルが装着されていると、ループバックテストを含む総合的なテストを実行します。

devname は NIC です。

指定されたネットワークデバイスの一部またはすべての設定を変更するには、**-s** または **--change** オプションが必要になります。以下のすべてのオプションは、**-s** または **--change** オプションが指定された場合のみ、適用されます。明確にするために、ここでは省略します。

これらの設定を永続的なものとするために、ETHTOOL_OPTS 指示文を使うことができます。ネットワークインターフェースがアクティベートされる際にインターフェース設定ファイルで好みのオプションの設定に使用できます。この指示文の使用方法については、「イーサネットインターフェース」 を参照してください。

--speed

--speed は、スピードを毎秒メガビット単位 (Mb/s) で設定するオプションです。スピード値を省略すると、対応するデバイスのスピードが表示されます。以下の形式になります。

```
--speed number devname
```

number 毎秒メガビット (Mb/s) 単位でのスピードです。

devname は NIC です。

--duplex

--duplex オペレーションの送信および受信モードを設定するオプションです。以下の形式になります。

```
--duplex word devname
```

word は以下のいずれかになります。

- **half** — 半二重モードを設定します。通常、ハブに接続している際に使用します。
- **full** — 全二重モードを設定します。通常、スイッチまたは別のホストに接続している際に使用します。

devname は NIC です。

--port

--port は、デバイスポートを選択する際に使用するオプションです。以下の形式になります。

```
--port value devname
```

value は、以下のいずれかです。

- **tp** — ツイストペアケーブルを媒介として使用するイーサネットインターフェースです。
- **aui** — Attachment Unit Interface (AUI) 。通常はハブで使用します。
- **bnc** — BNC コネクターおよび同軸ケーブルを使用するイーサネットインターフェースです。
- **mii** — Media Independent Interface (MII) を使用するイーサネットインターフェースです。
- **fibre** — 光ファイバーを媒介として使用するイーサネットインターフェースです。

devname は NIC です。

--autoneg

--autoneg は、ネットワークスピードとオペレーションモード (全二重もしくは半二重モード) のオートネゴシエーションを制御するオプションです。オートネゴシエーションが有効な場合、**-r, --negotiate** オプションを使ってネットワークスピードとオペレーションモードの再ネゴシエーションを開始できます。**--a, --show-pause** オプションを使うとオートネゴシエーションのステータスを表示できます。

以下の形式になります。

```
--autoneg value devname
```

value は、以下のいずれかです。

- **yes** — ネットワークスピードとオペレーションモードのネゴシエーションを許可します。
- **no** — ネットワークスピードとオペレーションモードのネゴシエーションを許可しません。

devname は NIC です。

--advertise

--advertise は、オートネゴシエーションに通知されるスピードとオペレーションモード (二重モード) を設定するオプションです。引数は、表9.1「Ethtool 通知オプション: スピードおよびオペレーションモード」 からの 1 つ以上の十六進法の値になります。

以下の形式になります。

```
--advertise HEX-VALUE devname
```

HEX-VALUE は、以下の表からの 1 つ以上の十六進法の値で、*devname* は NIC です。

表9.1 Ethtool 通知オプション: スピードおよびオペレーションモード

十六進法の値	スピード	二重モード	IEEE 基準?
0x001	10	半	はい
0x002	10	全	はい
0x004	100	半	はい
0x008	100	全	はい
0x010	1000	半	いいえ
0x020	1000	全	はい
0x8000	2500	全	はい
0x1000	10000	全	はい
0x20000	20000MLD2	全	いいえ
0x20000	20000MLD2	全	いいえ
0x40000	20000KR2	全	いいえ

--phyad

--phyad は、物理アドレスを変更するオプションです。MAC もしくはハードウェアアドレスと呼ばれることが多くありますが、ここでは物理アドレスと呼びます。

以下の形式になります。

```
--phyad HEX-VALUE devname
```

HEX-VALUE は十六進法形式での物理アドレスで、*devname* は、NIC です。

--xcvr

--xcvr は、トランシーバーのタイプを選択するオプションです。現在は、「内部」 および 「外部」 のみが指定できます。今後は他のタイプが追加される可能性があります。

以下の形式になります。

```
--xcvr word devname
```

word は以下のいずれかになります。

- **internal** — 内部トランシーバーを使用。
- **external** — 外部トランシーバーを使用。

devname は NIC です。

--wol

--wol は、「Wake-on-LAN」 オプションを設定するオプションです。すべてのデバイスが対応しているわけではありません。このオプションの引数は、どのオプションを有効にするかを指定する文字列です。

以下の形式になります。

```
--wol value devname
```

value は、以下の 1 つまたは 複数のものです。

- **p** — PHY アクティビティーでウェイク
- **u** — ユニキャストメッセージでウェイク
- **m** — マルチキャストメッセージでウェイク
- **b** — ブロードキャストメッセージでウェイク
- **g** — Wake-on-Lan; "magic packet" の受信でウェイク。
- **s** — Wake-on-Lan のパスワードを使用してセキュリティ機能を有効にします。
- **d** — Wake-on-Lan を無効にし、すべての設定をクリアにします。

devname は NIC です。

--sopass

--sopass は、「SecureOn」 パスワードを設定するオプションです。このオプションの引数は、イーサネット MAC 十六進法形式 (xx:yy:zz:aa:bb:cc) の 6 バイトである必要があります。

以下の形式になります。

```
--sopass xx:yy:zz:aa:bb:cc devname
```

xx:yy:zz:aa:bb:cc は MAC アドレスと同一形式のパスワードで、*devname* は NIC です。

--msglvl

--msglvl は、ドライバーのメッセージタイプフラグを名前もしくは番号で設定するオプションです。これらのタイプフラグの正確な意味は、ドライバーによって異なります。

以下の形式になります。

```
--msglvl value devname
```

value は以下のいずれかです。

- **HEX-VALUE** — メッセージタイプを示す十六進法の値です。
- **message-type** — プレーンテキストのメッセージタイプ名です。

devname は NIC です。

定義済みのメッセージタイプ名および番号は、以下の表に示すとおりです。

表9.2 ドライバーメッセージタイプ

メッセージタイプ	十六進法の値	詳細
drv	0x0001	一般的なドライバーステータス
probe	0x0002	ハードウェアのプローブ
link	0x0004	リンクのステータス
timer	0x0008	定期的なステータスチェック
ifdown	0x0010	非アクティベートにしているインターフェース
ifup	0x0020	アクティベートにしているインターフェース
rx_err	0x0040	エラーを受信
tx_err	0x0080	エラー送信
intr	0x0200	割り込み処理
tx_done	0x0400	送信完了
rx_status	0x0800	受信完了
pktdata	0x1000	パケットコンテンツ
hw	0x2000	ハードウェアのステータス
wol	0x4000	Wake-on-LAN ステータス

9.2.5. チャンネルボンディングインターフェース

Red Hat Enterprise Linux では、**bonding** カーネルモジュールと、*チャンネルボンディングインターフェース* と呼ばれる特殊なネットワークインターフェースを使用して、管理者が複数のネットワークインターフェースを単一のチャンネルにまとめてバインドすることが可能です。チャンネルボンディングにより、2 つ以上のネットワークインターフェースが 1 つとして機能できるようになると同時に、帯域幅が増加し、冗長性を実現します。

チャンネルボンディングインターフェースを作成するには、**/etc/sysconfig/network-scripts/** ディレクトリに **ifcfg-bond*N*** という名前のファイルを作成し、*N* をそのインターフェースの番号 **0** などに置き換えます。

ファイルのコンテンツは、イーサネットインターフェースなどボンディングされるインターフェースがどのようなタイプでも同一のものになります。唯一異なる点は、**DEVICE** 指示文が **bond*N*** で、 *N* はインターフェースの番号に置き換えられます。

チャンネルボンディング設定ファイルのサンプルは以下のとおりです:

例9.1 ifcfg-bond0 インターフェース設定ファイルのサンプル

```
DEVICE=bond0
IPADDR=192.168.1.1
NETMASK=255.255.255.0
ONBOOT=yes
BOOTPROTO=none
USERCTL=no
BONDING_OPTS="bonding parameters separated by spaces"
```

チャンネルボンディングインターフェースを作成した後に、バインドされるネットワークインターフェースを設定するには、その設定ファイルに **MASTER** 指示文と **SLAVE** 指示文を追加する必要があります。チャンネルボンディングされたインターフェースの各設定ファイルは、ほぼ同一となる場合があります。

例えば、2 つのイーサネットインターフェースがチャンネルボンディングされる場合、**eth0** と **eth1** は以下の例のようになる場合があります。

```
DEVICE=ethN
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

この例では、*N* をインターフェースの数値に置き換えます。

チャンネルボンディングインターフェースを有効にするには、カーネルモジュールがロードされている必要があります。チャンネルボンディングインターフェースがアクティブになった時にモジュールが確実にロードされるようにするには、**/etc/modprobe.d/** ディレクトリに ***bonding*.conf** という名前の新規ファイルを root で作成します。ファイル名は、末尾に **.conf** の拡張子が付いていれば、どのような名前にすることもできます。新規ファイルに以下の行を挿入します:

```
alias bondN bonding
```

N を **0** などのインターフェースの番号に置き換えます。設定済みの各チャンネルボンディングインターフェースには、新規 **/etc/modprobe.d/*bonding*.conf** ファイルに対応するエントリがなければなりません。

重要

ボンディングカーネルモジュールのパラメータは、**ifcfg-bond*N*** インターフェースファイル内の **BONDING_OPTS="*bonding parameters*"** 指示文でスペースで区切られた一覧として指定する必要があります。**/etc/modprobe.d/*bonding*.conf**、または廃棄になっている **/etc/modprobe.conf** ファイルのボンディングデバイスでは、オプションを指定**しないでください**。ボンディングモジュール設定の詳しい説明とアドバイス、及びボンディングパラメータの一覧を表示するには、「チャンネルボンディングの使用」 を参照してください。

9.2.6. ネットワークブリッジ

ネットワークブリッジは、ネットワーク間のトラフィックを MAC アドレスに基づいて転送するリンク層デバイスで、このためレイヤー 2 デバイスとも呼ばれます。転送の決定は、MAC アドレスの表に基づいて行われ、この表はどのホストがどのネットワーク接続しているかをネットワークブリッジが学習することで構築されます。Linux ホスト内では、ソフトウェアブリッジを使ってハードウェアをエミュレートすることができます。例えば、仮想化アプリケーション内で NIC を 1 つ以上の仮想 NIC と共有するなどです。ここではこのケースを例として説明します。

ネットワークブリッジを作成するには、**/etc/sysconfig/network-scripts/** ディレクトリに **ifcfg-br*N*** という名前のファイルを作成し、*N* をそのインターフェースの番号 **0** などに置き換えます。

ファイルのコンテンツは、イーサネットインターフェースなどブリッジされるインターフェースがどのようなタイプでも類似したものになります。相違点は、以下のとおりです。

- **DEVICE** 指示文は **br*N*** 形式でその引数としてインターフェース名を与えられ、この *N* はインターフェースの番号に置き換えられます。
- **TYPE** 指示文には、引数 **Bridge** または **Ethernet** が与えられます。この指示文は、デバイスタイプと、引数が大文字/小文字を区別するかを決定します。
- ブリッジインターフェース設定ファイルには IP アドレスがあり、物理インターフェースには MAC アドレスしかありません。
- 追加の指示文 **DELAY=0** が加わり、ブリッジがトラフィックを監視し、ホストの位置を学習し、フィルタリング機能の基になる MAC アドレス表を構築する間に、ブリッジが待機することを回避します。ルーティングループが可能でない場合は、デフォルトの 30 秒遅延は不要です。
- **NM_CONTROLLED=no** をイーサネットインターフェースに追加して、**NetworkManager** によるファイル変更を回避することが推奨されます。**NetworkManager** の今後のバージョンがブリッジ設定に対応することに備えて、これをブリッジ設定ファイルに追加することもできます。

以下は、静的 IP アドレスを使用したブリッジインターフェース設定ファイルの例です。

例9.2 ifcfg-br0 インターフェース設定ファイルの例

```
DEVICE=br0
TYPE=Bridge
IPADDR=192.168.1.1
NETMASK=255.255.255.0
ONBOOT=yes
```

```
BOOTPROTO=static
NM_CONTROLLED=no
DELAY=0
```

ブリッジを完成するには、別のインターフェースを作成するか既存のインターフェースを修正して、これをブリッジインターフェースに向けます。以下の例は、イーサネットインターフェース設定ファイルをブリッジインターフェースに向けたものです。**/etc/sysconfig/network-scripts/ifcfg-eth*X*** の物理インターフェースを設定します。ここでは *X* は、特定のインターフェースに対応する一意の番号です。

例9.3 ifcfg-ethX インターフェース設定ファイルの例

```
DEVICE=ethX
TYPE=Ethernet
HWADDR=AA:BB:CC:DD:EE:FF
BOOTPROTO=none
ONBOOT=yes
NM_CONTROLLED=no
BRIDGE=br0
```

注記

DEVICE 指示文では、インターフェース名はデバイスタイプを決定するわけではないので、ほぼいかなる名前でも使用できます。一般的に使われる名前の例は、**tap**、**dummy**、**bond** などです。**TYPE=Ethernet** 絶対に必要なわけではありません。**TYPE** 指示文が設定されていない場合、(名前が明確に異なるインターフェース設定ファイルと合致していなければ) そのデバイスはイーサネットデバイスとして扱われます。

指示文とネットワークインターフェース設定ファイルで使われているオプションの確認は、「インターフェース設定ファイル」 を参照してください。

警告

リモートホスト上でブリッジ設定をしていて、そのホストへの接続に設定中の物理 NIC を使用している場合、この先に進む前に接続が切断された場合の影響を検討してください。サービスを再起動する際には接続が失われ、エラーが発生すると接続を再確立することができない場合があります。コンソールもしくは帯域外のアクセスが推奨されます。

以下のようにネゴシエーションサービスを再起動して変更を反映させます

```
service network restart
```

2 つ以上のボンディングされたイーサネットインターフェースから作られたネットワークブリッジの例

を示します。これは、仮想化環境でよくあるアプリケーションです。ボンディングされたインターフェースのの設定ファイルに詳しくない場合は、「チャンネルボンディングインターフェース」 を参照してください。

2 つ以上のイーサネットインターフェース設定ファイルを作成もしくは編集し、以下のようにボンディングします。

```
DEVICE=ethX
TYPE=Ethernet
USERCTL=no
SLAVE=yes
MASTER=bond0
BOOTPROTO=none
HWADDR=AA:BB:CC:DD:EE:FF
NM_CONTROLLED=no
```

注記

eth*X* をインターフェース名に使うのは一般的な方法ですが、ほとんどどんな名前でも使用できます。よくある例は、**tap**、**dummy**、**bond** などです。

インターフェース設定ファイル **/etc/sysconfig/network-scripts/ifcfg-bond0** を以下のように作成もしくは編集します。

```
DEVICE=bond0
ONBOOT=yes
BONDING_OPTS='mode=1 miimon=100'
BRIDGE=brbond0
NM_CONTROLLED=no
```

ボンディングモジュールの設定に関する指示およびアドバイスと、ボンディングパラメーターの一覧については、「チャンネルボンディングの使用」 を参照してください。

インターフェース設定ファイル **/etc/sysconfig/network-scripts/ifcfg-brbond0** を以下のように作成もしくは編集します。

```
DEVICE=brbond0
ONBOOT=yes
TYPE=Bridge
IPADDR=192.168.1.1
NETMASK=255.255.255.0
NM_CONTROLLED=no
```

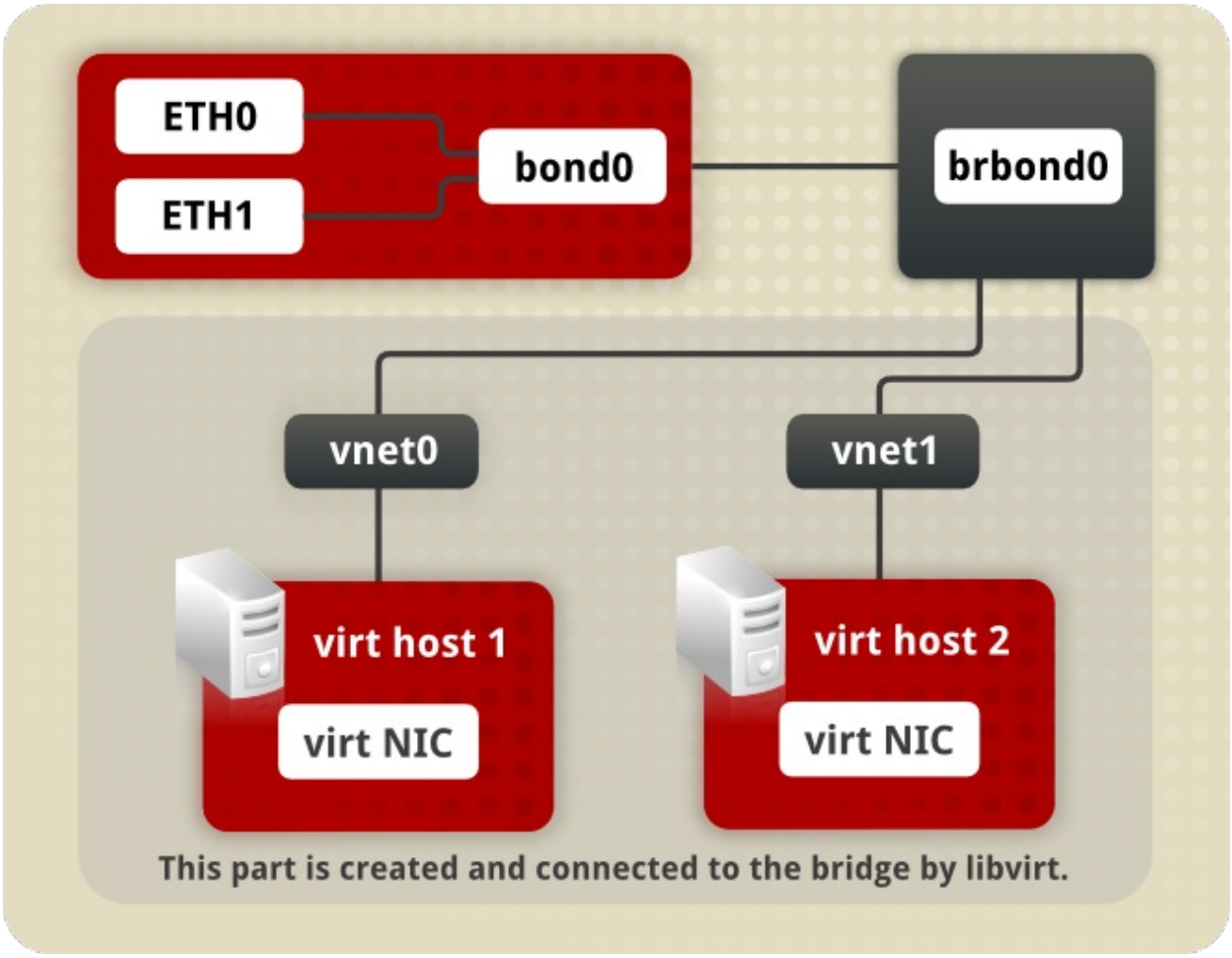


図9.1 2 つのボンディングされたインターフェースインターフェースで構成されるネットワークブリッジ

これで **MASTER=bond0** 指示文のある 2 つ以上のインターフェース設定ファイルができました。これらのファイルは、**DEVICE=bond0** 指示文を含む **/etc/sysconfig/network-scripts/ifcfg-bond0** という名前の設定ファイルに向けられています。この **ifcfg-bond0** は、**/etc/sysconfig/network-scripts/ifcfg-brbond0** 設定ファイルに向けられています。これには IP アドバイスが含まれ、ホスト内の仮想ネットワークへのインターフェースとして動作します。

以下のようにネゴシエーションサービスを再起動して変更を反映させます

```
service network restart
```

9.2.7. 802.1q VLAN タグのセットアップ

1. 以下のコマンドを入力して、モジュールがロードされていることを確認します。

   ```
   lsmod | grep 8021q
   ```

2. モジュールがロードされていない場合は、以下のコマンドでロードします。

   ```
   modprobe 8021q
   ```

3. **/etc/sysconfig/network-scripts/ifcfg-eth*X*** の物理インターフェースを以下のように設定します。*X* は特定のインターフェースに対応する一意の番号です。

```
DEVICE=ethX
TYPE=Ethernet
BOOTPROTO=none
ONBOOT=yes
```

4. **/etc/sysconfig/network-scripts** で VLAN インターフェース設定を行います。設定ファイル名は、物理インターフェースにピリオド **.** と VLAN ID 番号を加えたものにします。例えば、VLAN ID が 192 で物理インターフェースが **eth0** ならば、設定ファイル名は **ifcfg-eth0.192** となります。

```
DEVICE=ethX.192
BOOTPROTO=static
ONBOOT=yes
IPADDR=192.168.1.1
NETMASK=255.255.255.0
USERCTL=no
NETWORK=192.168.1.0
VLAN=yes
```

例えば、VLAN ID 193 で同一インターフェースの **eth0** 上に 2 つ目の VLAN を設定する必要がある場合は、VLAN 設定詳細で **eth0.193** の名前の新規ファイルを追加します。

5. 以下のようにネゴシエーションサービスを再起動して変更を反映させます

```
service network restart
```

9.2.8. エイリアスファイルとクローンファイル

使用頻度が低い 2 種類のインターフェース設定ファイルは、*エイリアス* ファイルと *クローン* ファイルです。iproute パッケージの **ip** コマンドが今では同一への複数アドレス割り当てに対応するので、この方法で複数アドレスを同一インターフェースにバインディングする必要はもうありません。

注記

本書執筆の時点では、 **NetworkManager** は、**ifcfg** ファイルの IP アドレスを検出しません。例えば、**ifcfg-eth0** および **ifcfg-eth0:1** ファイルがある場合に、**NetworkManager** が 2 つの接続を作成すると、混乱を引き起こすことになります。

新規インストールでは、ユーザーは **NetworkManager** の IPv4 または IPv6 タブ上で **手動** を選択し、同一インターフェースに複数の IP アドレスを割り当てることをお勧めします。このツールの使用に関する詳細は 8章*NetworkManager* を参照してください。

複数のアドレスを単一のインターフェースにバインドするために使用するエイリアスインターフェース設定ファイルは、**ifcfg-*if-name*:*alias-value*** 命名スキームを使用します。

例えば、**ifcfg-eth0:0** ファイルは **DEVICE=eth0:0** 及び **10.0.0.2** の静的 IP アドレスを指定するように設定でき、DHCP を介して **ifcfg-eth0** の IP 情報を受信するように既に設定されたイーサネットインターフェースのエイリアスとして機能します。こうした設定の下で、**eth0** は動的 IP アドレスに

バインドされますが、同一の物理ネットワークカードで、固定された **10.0.0.2** IP アドレスを介して要求を受信することができます。

警告

エイリアスインターフェースは DHCP に対応していません。

クローンインターフェース設定ファイルは、命名慣習に従って **ifcfg-*if-name-clone-name*** を使用します。エイリアスファイルは既存のインターフェースに複数のアドレスを使用できるようにするのに対し、クローンファイルはインターフェースの追加オプションを指定するのに使用されます。例えば、**eth0** という名前の標準 DHCP イーサネットインターフェースでは、以下のようになります。

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
```

USERCTL 指示文のデフォルト値は **no** であるため、これが指定されていなければ、ユーザーはこのインターフェースをアクティベート、非アクティベートすることはできません。ユーザーがインターフェースを制御できるようにするには、**ifcfg-eth0** を **ifcfg-eth0-user** にコピーしてクローンを作成し、**ifcfg-eth0-user** に以下の行を追加します:

```
USERCTL=yes
```

この方法では、**ifcfg-eth0** と **ifcfg-eth0-user** の設定オプションが統合されるため、ユーザーは **/sbin/ifup eth0-user** コマンドを使用して、**eth0** インターフェースをアクティベートすることができます。これは基本的な例ですが、この方法は様々なオプションやインターフェースで使用できます。

グラフィカルツールを使ってエイリアスやクローンインターフェース設定ファイルを作成することは、できなくなりました。しかし、本セクションの最初で説明したように、同一インターフェースに複数の IP アドレスを直接割り当てることが化膿になったので、この方法はもう必要ありません。新規インストールでは、ユーザーは **NetworkManager** の IPv4 または IPv6 タブ上で **手動** を選択し、同一インターフェースに複数の IP アドレスを割り当てることをお勧めします。このツールの使用に関する詳細は 8 章*NetworkManager* を参照してください。

9.2.9. ダイヤルアップインターフェース

ダイヤルアップ接続でインターネットに接続する場合は、そのインターフェース用の設定ファイルが必要となります。

PPP インターフェースファイルは、以下の形式で命名されます。

ifcfg-ppp*X*

X は特定のインターフェースに対応する一意の番号です。

wvdial または **Kppp** を使用してダイヤルアップアカウントを作成すると、PPP インターフェース設定ファイルが自動的に作成されます。このファイルは手動で作成、編集することもできます。

典型的な **/etc/sysconfig/network-scripts/ifcfg-ppp0** ファイルは以下のとおりです:

```
DEVICE=ppp0
NAME=test
WVDIALSECT=test
MODEMPORT=/dev/modem
LINESPEED=115200
PAPNAME=test
USERCTL=true
ONBOOT=no
PERSIST=no
DEFROUTE=yes
PEERDNS=yes
DEMAND=no
IDLETIMEOUT=600
```

Serial Line Internet Protocol (SLIP) は、別のダイヤルアップインターフェースですが、頻繁には使用されません。SLIP ファイルには、**ifcfg-sl0** のようなインターフェース設定ファイル名が付いています。

これらのファイルで使用される可能性のある他のオプションとして、以下があります:

DEFROUTE=*回答*

回答 が以下のいずれかの場合です:

- **yes** — このインターフェースをデフォルトルートとして設定します。
- **no** — このインターフェースをデフォルトルートとして設定しません。

DEMAND=*回答*

回答 が以下のいずれかの場合です:

- **yes** — このインターフェースは、誰かが使用を試みた時に **pppd** が接続を開始するのを許可します。
- **no** — このインターフェースでは、接続を手動で確立する必要があります。

IDLETIMEOUT=*値*

値 は、インターフェースが自動的に接続を切断するまでのアイドル状態の秒数です。

INITSTRING=*文字列*

文字列 は、モデムデバイスに渡される初期化文字列です。このオプションは主として、SLIP インターフェースと併用されます。

LINESPEED=*値*

値 は、デバイスのボーレートです。取り得る標準値として、**57600**、**38400**、**19200**、**9600** があります。

MODEMPORT=*デバイス*

デバイス は、このインターフェースの接続を確立するために使用されるシリアルデバイス名です。

MTU=*値*

*値*は、このインターフェースの *最大転送単位* (MTU) の設定です。MTU は、1 フレームが伝送できるデータの最大バイト数を指しており、ヘッダー情報は考慮されません。一部のダイヤルアップ状況下では、この値を **576** に設定すると、破棄されるパケットが少なくなり、接続のスループットが若干向上します。

NAME=*名前*

名前 は、ダイヤルアップ接続設定のコレクションに付けられるタイトルを指します。

PAPNAME=*名前*

名前 は、リモートシステムへの接続を許可するために発生する *パスワード認証プロトコル* (PAP) 交換中に付けられたユーザー名です。

PERSIST=*回答*

回答 が以下のいずれかの場合です:

- **yes** — このインターフェースは、モデムの切断後に非アクティブになるとしても常にアクティブな状態に維持する必要があります。
- **no** — このインターフェースは、常にアクティブな状態を維持する必要はありません。

REMIP=*アドレス*

アドレス は、リモートシステムの IP アドレスです。これは通常指定しません。

WVDIALSECT=*名前*

名前 は、このインターフェースを **/etc/wvdial.conf** のダイヤラー設定に関連付けします。このファイルには、ダイヤルする電話番号とインターフェースに関するその他の重要な情報が含まれます。

9.2.10. その他のインターフェース

以下が、その他の一般的なインターフェース設定ファイルです:

ifcfg-lo

ローカルの *ループバックインターフェース* は、テストに使われる場合が多々ある他、同じシステムに回帰接続する IP アドレスを必要とする様々なアプリケーションで使用されます。ループバックデバイスに送信されるデータはいずれもホストのネットワークレイヤーに直ちに返されます。

警告

ループバックインターフェーススクリプト **/etc/sysconfig/network-scripts/ifcfg-lo** は、決して手動では編集しないで下さい。手動で編集すると、システムが正常に動作しなくなる恐れがあります。

ifcfg-irlan0

赤外線インターフェース により、赤外線リンクでラップトップやプリンターなどのデバイス間の情報が流れるようにします。これは、一般的にピアツーピア接続で発生する点を除いては、イーサネットデバイスと同様に機能します。

ifcfg-plip0

パラレル回線インターフェースプロトコル (PLIP) 接続は、パラレルポートを利用する点を除いては、イーサネットデバイスとほぼ同様に機能します。

System z 上の Linux のインターフェース設定ファイルには以下が含まれます。

ifcfg-hsiN

HiperSockets は、IBM System z メインフレーム上の z/VM ゲスト仮想マシンおよび論理パーティション (LPAR) 内およびこれらにまたがる高速 TCP/IP 通信のインターフェースです。

9.3. インターフェース制御スクリプト

インターフェース制御スクリプトは、システムインターフェースをアクティベート及び、非アクティベートします。制御スクリプトを呼び出す主要なインターフェース制御スクリプトは、**/sbin/ifdown** と **/sbin/ifup** の 2 つあり、**/etc/sysconfig/network-scripts/** ディレクトリにあります。

ifup と **ifdown** のインターフェーススクリプトは、**/sbin/** ディレクトリのスクリプトへのシンボリックリンクです。いずれかのスクリプトが呼び出されると、インターフェースの値を以下のように指定する必要があります:

```
ifup eth0
```

警告

ifup と **ifdown** のインターフェーススクリプトが、ユーザーがネットワークインターフェースをアクティベート、非アクティベートするために使用すべき唯一のスクリプトです。

以下のスクリプトは、参照目的のみのために記載しています。

ネットワークインターフェースを立ち上げるプロセスで様々なネットワーク初期化タスクを実行するために使用する 2 つのファイルは、**/etc/rc.d/init.d/functions** と **/etc/sysconfig/network-scripts/network-functions** です。詳細は 「ネットワーク関数ファイル」 を参照してください。

インターフェースが指定され、要求を実行するユーザーがインターフェースの制御を許可されたことが確認されると、適切なスクリプトによってインターフェースはアクティベート、非アクティベートされます。以下は **/etc/sysconfig/network-scripts/** ディレクトリにある一般的なインターフェース制御スクリプトです:

ifup-aliases

1 つのインターフェースに複数の IP アドレスが関連付けされる場合に、インターフェース設定ファイルから IP エイリアスを設定します。

ifup-ippp 及び ifdown-ippp

ISDN インターフェースをアクティベート、非アクティベートします。

ifup-ipv6 及び ifdown-ipv6

IPv6 インターフェースをアクティベート、非アクティベートします。

ifup-plip

PLIP インターフェースをアクティベートします。

ifup-plusb

ネットワーク接続のため USB インターフェースをアクティベートします。

ifup-post 及び ifdown-post

インターフェースがアクティベート、非アクティベートされた後に実行されるコマンドが含まれます。

ifup-ppp 及び ifdown-ppp

PPP インターフェースをアクティベート、非アクティベートします。

ifup-routes

インターフェースがアクティベートされると、デバイスに対する静的ルートを追加します。

ifdown-sit 及び ifup-sit

IPv4 接続内の IPv6 トンネルをアクティベート、非アクティベートすることに関連する関数呼び出しが含まれます。

ifup-wireless

ワイヤレスインターフェースをアクティベートします。

警告

/etc/sysconfig/network-scripts/ ディレクトリのスクリプトを削除または変更すると、インターフェース接続の動作が不規則になる、または障害が発生する場合があります。そのため、ネットワークインターフェースに関連するスクリプトの修正は、上級ユーザーのみが行うことをお勧めします。

すべてのネットワークスクリプトを同時に操作する最も簡単な方法は、以下のコマンドで例示しているように、ネットワークサービス (**/etc/rc.d/init.d/network**) で **/sbin/service** コマンドを使用することです。

```
/sbin/service network action
```

ここでは、*action* は **start**、**stop**、または **restart** のいずれかになります。

以下のコマンドを使用すると、設定済みのデバイスと現在アクティブなネットワークインターフェースの一覧を表示できます:

```
/sbin/service network status
```

9.4. 静的ルートおよびデフォルトゲートウェイ

静的ルートは、デフォルトゲートウェイを通過すべきでない、通過してはいけないトラフィックのためのものです。ルーティングは通常、ルーティングデバイスが処理するので、Red Hat Enterprise Linux サーバーやクライアントでは静的ルートの設定は多くの場合、必要ありません。例外は、暗号化 VPN トンネルの通過が必要なトラフィックや、より低コストのルートを通過すべきトラフィックなどです。デフォルトゲートウェイは、ローカルネットワークが宛先となっていないすべてのトラフィックのためのもので、これらのトラフィックのルーティングテーブルには優先ルートが指定されていません。デフォルトゲートウェイはこれまで、専用ネットワークルーターでした。

静的ルート

ip route コマンドを使って IP ルーティングテーブルを表示します。静的ルートが必要な場合は、**ip route add** コマンドでルーティングテーブルに追加でき、削除したい時は **ip route del** コマンドを使います。単一の IP アドレスなどのホストアドレスに静的ルートを追加するには、**root** で以下のコマンドを発行します。

```
ip route add X.X.X.X
```

X.X.X.X は、ドッド区切り表記のホストの IP アドレスになります。多くの IP アドレスを意味するネットワークに静的ルートを追加するには、**root** で以下のコマンドを発行します。

```
ip route add X.X.X.X/Y
```

X.X.X.X は、ドッド区切り表記のネットワークの IP アドレスになり、Y はネットワークのプレフィックスになります。ネットワークプレフィックスは、サブネットマスクで有効なビットの数です。ネットワークアドレスとスラッシュ、プレフィックスというこの長さのフォーマットは、CIDR 表記と呼ばれます。

静的ルート設定はインターフェースごとに、**/etc/sysconfig/network-scripts/route-*interface*** ファイルに格納されています。例えば、**eth0** インターフェースの静的ルートは、**/etc/sysconfig/network-scripts/route-eth0** ファイルに格納されています。**route-*interface*** ファイルには IP コマンド引数、およびネットワーク/ネットマスク指示文の 2 つの形式があり、これらは下記で説明されています。

デフォルトゲートウェイ

デフォルトゲートウェイは、GATEWAY 指示文で指定されており、グローバルかインターフェース特定の設定ファイルのどちらかで指定できます。ネットワークインターフェースが 1 つ以上ある場合は特に、デフォルトゲートウェイをグローバルで指定すると有利な点があります。継続的に適用された場合、障害発見を容易にすることができます。GATEWAYDEV 指示文もあり、これはグローバルのオプションです。複数のデバイスが GATEWAY を指定し、1 つのインターフェースが GATEWAYDEV 指示文を使用する場合、その指示文が優先することになります。このオプションは、インターフェースがダウンすると予想外の結果をもたらすことがあり、障害発見を複雑にしかねないので、推奨されません。

グローバルのデフォルトゲートウェイ設定は、**/etc/sysconfig/network** ファイルに格納されてい

ます。このファイルは、すべてのネットワークインターフェースのゲートウェイとホストの情報を指定します。このファイルとそれが使用できる指示文の詳細は、「/etc/sysconfig/network」 を参照してください。

IP コマンド引数形式

インターフェースごとの設定ファイルで必要な場合は、1 行目にデフォルトゲートウェイを定義します。これは、デフォルトゲートウェイが DHCP を介して設定されておらず、上記のようにグローバルに設定されていない場合のみに必要です。

```
default via X.X.X.X dev interface
```

X.X.X.X は、デフォルトゲートウェイの IP アドレスです。*interface* は、デフォルトゲートウェイに接続されているインターフェース、または到達可能なインターフェースです。 **dev** オプションは選択可能で、省略することもできます。

静的ルートを定義します。各行は個別のルートとして解析されます。

```
X.X.X.X/Y via X.X.X.X dev interface
```

X.X.X.X/Y は、静的ルートのネットワークアドレスとネットマスクです。*X.X.X.X* と *interface* はそれぞれ、デフォルトゲートウェイの IP アドレスとインターフェースです。*X.X.X.X* アドレスは、デフォルトゲートウェイの IP アドレスである必要はありません。大半の場合、*X.X.X.X* は異なるサブネットの IP アドレスとなり、*interface* はそのサブネットに接続されたインターフェース、または到達可能なインターフェースです。必要な数の静的ルートを追加します。

以下は、IP コマンド引数形式を用いた **route-eth0** ファイルのサンプルです。デフォルトゲートウェイは、192.168.0.1 でインターフェースは eth0 です。2 つの静的ルートは、10.10.10.0/24 と 172.16.1.0/24 の両ネットワークに対するものです

```
default via 192.168.0.1 dev eth0
10.10.10.0/24 via 192.168.0.1 dev eth0
172.16.1.0/24 via 192.168.0.1 dev eth0
```

静的ルートは、他のサブネットワークに対してのみ設定することをお勧めします。10.10.10.0/24 と 172.16.1.0/24 のネットワークに向かうパケットは、いずれにしてもデフォルトゲートウェイを使用するため、上記の例は必要ではありません。下記の例では、192.168.0.0/24 サブネットのマシン上での異なるサブネットへの静的ルートを設定しています。この例のマシンには、192.168.0.0/24 サブネットの **eth0** インターフェースと 10.10.10.0/24 サブネットの **eth1** インターフェース (10.10.10.1) があります。

```
10.10.10.0/24 via 10.10.10.1 dev eth1
```

終了インターフェースの特定は、オプションです。特定のインターフェースからトラフィックを強制的に締め出したい場合は、これが便利です。例えば VPN の場合、tun0 インターフェースが宛先ネットワークへの異なるサブネットにあったとしても、リモートネットワークへのトラフィックを強制的にこのインターフェース通過とすることができます。

重要

デフォルトゲートウェイが DHCP から既に割り当てられている場合は、IP コマンド引数形式によっては、起動中、または **ifup** コマンドを使用してインターフェースを非アクティブの状態からアクティブにする時に、次の 2 つのエラーのどちらかが発生する可能性があります：「RTNETLINK answers: File existss (RTNETLINK の応答: ファイルが存在)」 または 「Error: either "to" is a duplicate, or "*X.X.X.X*" is a garbage (エラー: "to"が重複、または "*X.X.X.X*" が不要)」。*X.X.X.X* はゲートウェイか、または異なる IP アドレスです。こうしたエラーは、デフォルトゲートウェイを使用した別のネットワークへのルートが別にある場合にも発生することがあります。こうしたエラーは無視しても安全です。

ネットワーク/ネットマスク指示文の形式

route-*interface* ファイルに、ネットワーク/ネットマスク指示文の形式を使用することも可能です。以下は、ネットワーク/ネットマスク形式のテンプレートで、説明はその次をご覧ください:

```
ADDRESS0=X.X.X.X NETMASK0=X.X.X.X GATEWAY0=X.X.X.X
```

- **ADDRESS0=*X.X.X.X*** は、静的ルートのネットワークアドレスです。
- **NETMASK0=*X.X.X.X*** は、**ADDRESS0=*X.X.X.X*** で定義されているネットワークアドレス用のネットマスクです。
- **GATEWAY0=*X.X.X.X*** はデフォルトゲートウェイ、または **ADDRESS0=*X.X.X.X*** に到達するために使用可能な IP アドレスです。

以下は、ネットワーク/ネットマスク指示文の形式を使用した **route-eth0** ファイルのサンプルです。デフォルトゲートウェイは 192.168.0.1、インターフェースは **eth0** です。2 つの静的ルートは、10.10.10.0/24 及び 172.16.1.0/24 のネットワークに対するものです。しかし、前述したとおり 10.10.10.0/24 及び 172.16.1.0/24 のネットワークはいずれにしてもデフォルトゲートウェイを使用するため、この例は必要ではありません:

```
ADDRESS0=10.10.10.0
NETMASK0=255.255.255.0
GATEWAY0=192.168.0.1
ADDRESS1=172.16.1.0
NETMASK1=255.255.255.0
GATEWAY1=192.168.0.1
```

後続する静的ルートは、順番に番号付けされる必要があり、いずれの値もスキップしてはいけません。例えば、**ADDRESS0**、**ADDRESS1**、**ADDRESS2** というようになります。

以下は、192.168.0.0/24 サブネットのマシン上で異なるサブネットへの静的ルートを設定する例です。この例のマシンには、192.168.0.0/24 サブセットの **eth0** インターフェースと、10.10.10.0/24 サブネットの **eth1** インターフェース (10.10.10.1) があります。

```
ADDRESS0=10.10.10.0
NETMASK0=255.255.255.0
GATEWAY0=10.10.10.1
```

DHCP を使用すると、これらの設定を自動的に割り当てることに注意してください。

9.5. ネットワーク関数ファイル

Red Hat Enterprise Linux は、インターフェースをアクティベート、非アクティベートするために使用される重要な一般的関数を格納した複数のファイルを活用します。これらの関数を各インターフェース制御ファイルに強制的に格納する代わりに、必要に応じて呼び出される数種のファイルにグループ化しています。

/etc/sysconfig/network-scripts/network-functions ファイルには、最も一般的に使用される IPv4 関数が格納されており、数多くのインターフェース制御スクリプトに役立ちます。関数の例として、インターフェースのステータス変更に関する情報を要求した実行プログラムへのコンタクト、ホストネームの設定、ゲートウェイデバイスの検索、特定のデバイスが非アクティブかどうかの検証、デフォルトルートの追加があります。

IPv6 インターフェースに必要な関数は IPv4 インターフェースとは異なるため、**/etc/sysconfig/network-scripts/network-functions-ipv6** ファイルはこの情報の保持専用に存在します。このファイルの関数は、静的 IPv6 ルートの設定と削除、トンネルの作成と削除、インターフェースへの IPv6 アドレスの追加と削除、インターフェース上の IPv6 アドレスの有無をテストします。

9.6. その他のリソース

以下は、ネットワークインターフェースについてさらに詳しく説明している資料です。

9.6.1. インストールされているドキュメント

/usr/share/doc/initscripts-*version*/sysconfig.txt

本章では触れていない IPv6 オプションなど、ネットワーク設定ファイル用の利用可能なオプションについて説明しています。

9.6.2. 役立つ Web サイト

http://linux-ip.net/gl/ip-cref/

このファイルには、ルーティングテーブルの操作などに使用できる ip コマンドに関する豊富な情報が含まれています。この情報は、オプションのコンテンツチャンネルから iproute-doc サブパッケージをインストールすると、**ip-cref.ps** ファイルでも閲覧できます。

パート IV. インフラストラクチャのサービス

ここでは、サービスとデーモンの設定、認証設定、リモートログインの有効方法について解説します。

第10章 サービスとデーモン

システムのセキュリティ維持管理は極めて重要なタスクです。そのアプローチの一つとして、システムサービスへのアクセスを慎重に管理することがあげられます。システムが特定のサービスにオープンアクセスを提供する必要がある場合があります (例えば Web サーバーを実行している場合は **httpd** など)。但し、サービスを提供する必要がない場合は、バグが悪用される可能性を最小限に抑えるためオフにしておくべきです。

本章では、ランレベルの概念、及びデフォルトのランレベルを設定する方法について説明します。又、ランレベルで実行するサービスの設定についても取り上げ、コマンドラインで **service** コマンドを使用したサービスの起動、停止、再起動の方法についても説明します。

重要

新たなサービス用にアクセスを許可する場合は、ファイアウォールと **SELinux** の両方を設定する必要があることを常に念頭に置いて下さい。新規サービスの設定時に、最もよくあるミスの 1 つは、そのサービスへのアクセスを許可するのに必要なファイアウォール設定と SELinux ポリシーの実装を怠ってしまうことです。詳細は、Red Hat Enterprise Linux 6 『セキュリティガイド』 を参照して下さい。

10.1. デフォルトのランレベルの設定

ランレベル とは、状態または *モード* のことで、そのランレベルが選択された場合に実行されるサービスによって定義されます。番号が付いた 7 つのランレベルがあります (**0** から始まります)。

表10.1 Red Hat Enterprise Linux でのランレベル

ランレベル	詳細
0	システムの停止に使用します。このランレベルは保持され、変更できません。
1	シングルユーザーモードでの実行に使用します。このランレベルは保持され、変更できません。
2	デフォルトでは使用されません。自由に定義できます。
3	コマンドラインユーザーインターフェースを使用するフルマルチユーザーモードでの実行に使用します。
4	デフォルトでは使用されません。自由に定義できます。
5	グラフィカルユーザーインターフェースを使用するフルマルチユーザーモードでの実行に使用します。
6	システムの再起動に使用します。このランレベルは保持され、変更できません。

どのランレベルで稼働しているかを確認するには、以下のコマンドを入力します:

```
~]$ runlevel
N 5
```

runlevel コマンドにより、以前と現在のランレベルが表示されます。上記の例では、数字の **5** が表示されており、システムはグラフィカルユーザーインターフェースを使用するフルマルチユーザーモードで実行されていることになります。

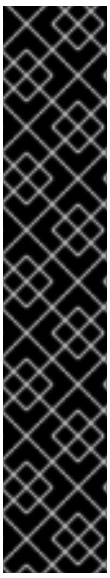
デフォルトのランレベルを変更するには、**/etc/inittab** ファイルを修正します。このファイルの終端の近くに、次のような行があります。

```
id:5:initdefault:
```

このファイルを **root** で編集し、この行の数字を、希望するランレベルに変更します。システムを再起動すると変更内容が反映されます。

10.2. サービスの設定

ブート時に起動するサービスを設定できるように、Red Hat Enterprise Linux には次のユーティリティが同梱されています: **サービスの設定** グラフィカルアプリケーション、**ntsysv** テキストユーザーインターフェース、**chkconfig** コマンドラインツール



重要

POWER アーキテクチャーで最適なパフォーマンスを実現するためには、**irqbalance** サービスを有効にすることを推奨します。大半の場合、このサービスは Red Hat Enterprise Linux 6 のインストール時にインストールされ、実行されるよう設定されています。**irqbalance** が実行されているかどうか確認するには、シェルプロンプトで **root** として以下のコマンドを入力します:

```
~]# service irqbalance status
irqbalance (pid  1234) is running...
```

グラフィカルユーザーインターフェースを使用したサービスの有効化及び実行の方法については 「サービス設定ユーティリティの使用」 を参照して下さい。コマンドラインでのタスクの実行方法については 「chkconfig ユーティリティの使用」 と 「サービスの実行」 をそれぞれ参照して下さい。

10.2.1. サービス設定ユーティリティの使用

サービスの設定 ユーティリティは、メニューから、特定のランレベルで起動するサービスを設定したり、サービスを起動、停止、再起動するために Red Hat が開発したグラフィカルアプリケーションです。このユーティリティを起動するには、パネルから **システム** → **管理** → **サービス** の順に選択するか、シェルプロンプトで **system-config-services** コマンドを入力します。

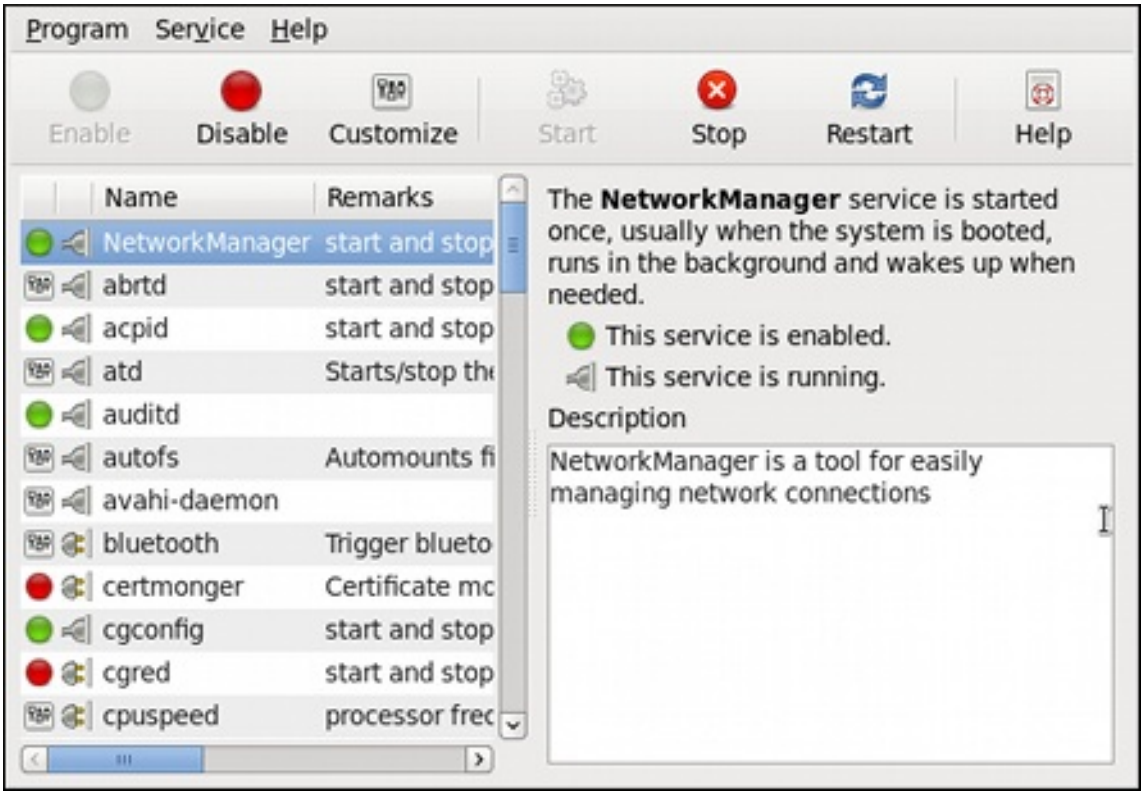


図10.1 サービスの設定ユーティリティ

このユーティリティは、利用可能なすべてのサービス (**/etc/rc.d/init.d/** ディレクトリからのサービスおよび **xinetd** により制御されているサービス) の一覧とともにその詳細と現在の状態を表示します。使用されるアイコンとそれぞれの意味に関する説明の全一覧は、表10.2「サービスの状態」 を参照して下さい。

初めて変更を加える時は、既に認証されていない限り、スーパーユーザーのパスワードを入力するよう求められることに注意してください。

表10.2 サービスの状態

アイコン	詳細
	このサービスは有効です
	このサービスは無効です
	このサービスは、選択されたランレベルで有効です
	このサービスは実行中です
	このサービスは停止しています
	このサービスには、何らかの障害があります

アイコン	詳細
	このサービスの状態は不明です

10.2.1.1. サービスの有効化と無効化

サービスを有効にするには、そのサービスを一覧から選択して、ツールバーの **Enable (有効)** ボタンをクリックするか、メインメニューから **Service (サービス)** → **Enable (有効)** の順に選択します。

サービスを無効にするには、そのサービスを一覧から選択して、ツールバーの **Disable (無効)** ボタンをクリックするか、メインメニューから **Service (サービス)** → **Service (無効)** の順に選択します。

10.2.1.2. サービスの開始、再起動と停止

サービスを開始するには、そのサービスを一覧から選択して、ツールバーの **Start (開始)** ボタンをクリックするか、メインメニューから **Service (サービス)** → **Start (開始)** の順に選択します。**xinetd** によって制御されているサービスは要求に応じて **xinetd** により起動されるため、このオプションは使用できない点に注意して下さい。

実行中のサービスを再起動するには、そのサービスを一覧から選択して、ツールバーの **Restart (再開始)** ボタンをクリックするか、メインメニューから **Service (サービス)** → **Restart (再開始)** の順に選択します。**xinetd** によって制御されているサービスは自動的に **xinetd** により開始/停止されるため、このオプションは使用できない点に注意して下さい。

サービスを停止するには、そのサービスを一覧から選択して、ツールバーの **Stop (停止)** ボタンをクリックするか、メインメニューから **Service (サービス)** → **Stop (停止)** の順に選択します。**xinetd** によって制御されているサービスはジョブが終了した時点で **xinetd** により停止されるため、このオプションは使用できない点に注意して下さい。

10.2.1.3. ランレベルの選択

サービスを特定のランレベルのみに有効にするには、そのサービスを一覧から選択して、ツールバーの **Customize (カスタマイズ)** ボタンをクリックするか、メインメニューから **Service (サービス)** → **Customize (カスタマイズ)** の順に選択します。次に、サービスを実行するランレベルの横にあるチェックボックスにチェックマークを入れます。**xinetd** によって制御されているサービスには、このオプションは使用できない点に注意して下さい。

10.2.2. ntsysv ユーティリティの使用

ntsysv ユーティリティは、選択したランレベルで起動するサービスを設定するために使用する、シンプルなテキストユーザーインターフェースを使用するコマンドラインアプリケーションです。このユーティリティを開始するには、**root** でシェルプロンプトに **ntsysv** と入力します。

図10.2 ntsysv ユーティリティ

このユーティリティにより、利用可能なサービス (**/etc/rc.d/init.d/** ディレクトリからのサービス) の一覧と現在の状態、さらには **F1** キーを押すと説明を表示することができます。使用される記号一覧とその意味の説明については 表10.3「サービスの状態」 を参照して下さい。

表10.3 サービスの状態

記号	詳細
[*]	このサービスは有効です
[ ]	このサービスは無効です

10.2.2.1. サービスの有効化と無効化

サービスを有効にするには、**上向き** と **下向き** の矢印キーを使用して一覧を移動し、対象のサービスを **スペースキー** で選択します。括弧内にアスタリスク (*) が表示されます。

サービスを無効にするには、**上向き** と **下向き** の矢印キーを使用して一覧を移動し、**スペースキー** で状態を切り替えます。括弧内のアスタリスク (*) が表示されなくなります。

手順が完了したら、**Tab** キーを使って **Ok** ボタンまで進み、**Enter** キーを押して変更を確定します。**ntsysv** は、実際にはサービスの開始や停止をしない点に注意して下さい。サービスを即時に起動、停止する必要がある場合は、「サービスの開始」 で説明したように **service** コマンドを使用します。

10.2.2.2. ランレベルの選択

デフォルトでは、**ntsysv** ユーティリティは現在のランレベルにのみ影響を及ぼします。他のランレベルでサービスを有効、無効にするには、**root** でコマンドに **--level** のオプションと、その後に設定する各ランレベルを示す 0 から 6 までの数字を追加して実行します。

```
ntsysv --level runlevels
```

例えばランレベル 3 と 5 を設定するには、以下のように入力します。

```
~]# ntsysv --level 35
```

10.2.3. chkconfig ユーティリティの使用

chkconfig ユーティリティは、選択したサービスをどのランレベルで起動するかを指定できるコマンドラインツールです。このユーティリティを使用すると、すべての利用可能なサービスと現在の設定を表示することもできます。一覧表示以外で、このコマンドを使用するには、スーパーユーザー権限が必要となる点に注意して下さい。

10.2.3.1. サービスの一覧表示

システムサービス (**/etc/rc.d/init.d/** ディレクトリからのサービスおよび **xinetd** により制御されているサービス) の一覧を表示するには、**chkconfig --list** と入力するか、追加の引数なしで **chkconfig** を使用します。これにより、以下の例のような出力が表示されます。

```
~]# chkconfig --list
NetworkManager  0:off   1:off   2:on    3:on    4:on    5:on    6:off
abrtd           0:off   1:off   2:off   3:on    4:off   5:on    6:off
acpid           0:off   1:off   2:on    3:on    4:on    5:on    6:off
anamon          0:off   1:off   2:off   3:off   4:off   5:off   6:off
atd             0:off   1:off   2:off   3:on    4:on    5:on    6:off
auditd          0:off   1:off   2:on    3:on    4:on    5:on    6:off
avahi-daemon    0:off   1:off   2:off   3:on    4:on    5:on    6:off
... several lines omitted ...
wpa_supplicant  0:off   1:off   2:off   3:off   4:off   5:off   6:off

xinetd based services:
        chargen-dgram:  off
        chargen-stream: off
        cvs:            off
        daytime-dgram:  off
        daytime-stream: off
        discard-dgram:  off
... several lines omitted ...
        time-stream:    off
```

各行にはサービス名と 7 つの番号が付いたランレベルそれぞれの状態 (**オン**/**オフ**) が表示されています。例えば、上記の一覧では **NetworkManager** は、ランレベル 2、3、4、5 で有効であり、**abrtd** はランレベル 3 と 5 で実行されます。**xinetd** ベースのサービスは最後に表示され、状態は **on**/**off** のいずれかです。

選択したサービスのみの現在の設定を表示するには、**chkconfig --list** の後にサービス名を指定したコマンドを使用します:

```
chkconfig --list service_name
```

例えば、**sshd** サービスの現在の設定を表示するには、以下のように入力します。

```
~]# chkconfig --list sshd
sshd            0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

このコマンドを使って **xinetd** によって管理されているサービスの状態を表示することもできます。この場合、出力には単にサービスが有効か無効かが表示されます。

```
~]# chkconfig --list rsync
rsync           off
```

10.2.3.2. サービスの有効化

ランレベル 2、3、4、5 でサービスを有効にするには **root** でシェルプロンプトに以下を入力します。

```
chkconfig service_name on
```

例えば、これら 4 つのランレベルで **httpd** サービスを有効にするには、以下を入力します。

```
~]# chkconfig httpd on
```

特定のランレベルのみでサービスを有効にしたい場合は、サービスを実行する各ランレベルを示す 0 から 6 までの数字を **--level** のオプションの後に付けて追加します。

```
chkconfig service_name on --level runlevels
```

例えば **abrtd** をランレベル 3 と 5 で有効にするには、以下のように入力します。

```
~]# chkconfig abrtd on --level 35
```

次回にこれらのランレベルのいずれかに入った時に、該当するサービスが起動します。サービスを即時に起動する必要がある場合は、「サービスの開始」 で説明したように **service** コマンドを使用します。

xinetd により管理されているサービスはサポートされていないため、これと **--level** オプションを合わせて使用 **しない** でください。例えば、**rsync** サービスを有効にするには、以下のように入力します。

```
~]# chkconfig rsync on
```

xinetd デーモンが実行中の場合、手動でデーモンを再起動しなくても、サービスは即座に有効になります。

10.2.3.3. サービスの無効化

ランレベル 2、3、4、5 でサービスを無効にするには **root** でシェルプロンプトに以下を入力します。

```
chkconfig service_name off
```

例えば、これら 4 つのランレベルで **httpd** サービスを無効にするには、以下を入力します。

```
~]# chkconfig httpd off
```

特定のランレベルのみでサービスを無効にしたい場合は、サービスを実行 **しない** 各ランレベルを示す 0 から 6 までの数字を **--level** のオプションの後に付けて追加します。

```
chkconfig service_name off --level runlevels
```

例えば **abrtd** をランレベル 2 と 4 で無効にするには、以下のように入力します。

```
~]# chkconfig abrtd off --level 24
```

次回にこれらのランレベルのいずれかに入った時に、該当するサービスが停止します。サービスを即時に停止する必要がある場合は、「サービスの停止」 で説明したように **service** コマンドを使用します。

xinetd により管理されているサービスはサポートされていないため、これと **--level** オプションを合わせて使用 **しない** でください。例えば、**rsync** サービスを無効にするには、以下のように入力します。

```
~]# chkconfig rsync off
```

xinetd デーモンが実行中の場合、手動でデーモンを再起動しなくても、サービスは即座に無効になります。

10.3. サービスの実行

service ユーティリティにより、**/etc/init.d/** ディレクトリからのサービスを起動、停止、再起動することができます。

10.3.1. サービスステータスの確認

現在のサービスの状態を確認するには、シェルプロンプトで次のコマンドを入力します。

```
service service_name status
```

例えば、**httpd** サービスの状態を確認するには、以下のように入力します。

```
~]# service httpd status
httpd (pid  7474) is running...
```

利用可能な全サービスの状態を一度に表示するには、**service** コマンドを **--status-all** オプションで実行します。

```
~]# service --status-all
abrt (pid  1492) is running...
acpid (pid  1305) is running...
atd (pid  1540) is running...
auditd (pid  1103) is running...
automount (pid 1315) is running...
Avahi daemon is running
cpuspeed is stopped
... several lines omitted ...
wpa_supplicant (pid  1227) is running...
```

「サービス設定ユーティリティの使用」 で説明のとおり、**サービスの設定** ユーティリティを使用することもできます。

10.3.2. サービスの開始

サービスを開始するには、**root** でシェルプロンプトに以下を入力します。

```
service service_name start
```

例えば、**httpd** サービスを開始するには、以下のように入力します。

```
~]# service httpd start
Starting httpd:                                            [  OK  ]
```

10.3.3. サービスの停止

実行中のサービスをていしするには、**root** でシェルプロンプトに以下を入力します。

```
service service_name stop
```

例えば、**httpd** サービスを停止するには、以下のように入力します。

```
~]# service httpd stop
Stopping httpd:                                            [  OK  ]
```

10.3.4. サービスの再開

サービスを再開するには、**root** でシェルプロンプトに以下を入力します。

```
service service_name restart
```

例えば、**httpd** サービスを再開するには、以下のように入力します。

```
~]# service httpd restart
Stopping httpd:                                            [  OK  ]
Starting httpd:                                            [  OK  ]
```

10.4. その他のリソース

10.4.1. インストールされているドキュメント

- **chkconfig**(8) — **chkconfig** ユーティリティの man ページです。
- **ntsysv**(8) — **ntsysv** ユーティリティの man ページです。
- **service**(8) — **service** ユーティリティの man ページです。
- **system-config-services**(8) — **system-config-services** ユーティリティの man ページです。

10.4.2. 関連資料

『Red Hat Enterprise Linux 6 セキュリティガイド』

Red Hat Enterprise Linux 6 のセキュリティ保護についてのガイドです。ファイアウォールの設定や **SELinux** の設定に関する有益な情報が記載されています。

第11章 認証の設定

認証 とは、ユーザーがシステムによって識別されて確認される手段です。認証プロセスはユーザー名とパスワードなど、なんらかの識別子と認証情報を必要とします。この認証情報はそれからシステムのデータストアに保存してある情報と比較されます。Red Hat Enterprise Linuxでは認証設定ツール (Authentication Configuration Tool) が、LDAP など、ユーザー認証情報に使用するデータストアの設定を手伝います。

利便性とシングルサインオンのために、Red Hat Enterprise Linux は中心となるデーモンを使用して多くの異なるデータストアのユーザー認証情報を保存することができます。System Security Services Daemon (システムセキュリティサービスデーモン) (SSSD) は、LDAP や Kerberos、外部のアプリケーショと対話して、ユーザー認証情報を確認できます。認証設定ツールは SSSD、NIS、Winbind、LDAP を設定し、認証プロセスとキャッシングを組み合わせることができます。

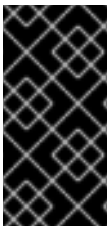
11.1. システム認証の設定

ユーザーが Red Hat Enterprise Linux システムにログインする時には、ユーザーはなんらかの **認証情報** を提示してユーザーの ID を確立します。システムはその認証情報を設定された認証サービスに照らし合わせてチェックします。認証情報が一致してユーザーのアカウントがアクティブだと、ユーザーは**認証され**ます。 (ユーザーが認証されると、その情報はアクセス制御サービスに渡され、そのユーザーに何が許可されているかを確認します。これが、ユーザーがアクセスする **権限のある** リソースです。)

ユーザーを確認するための情報はローカルシステム上に存在するか、ローカルシステムがリモートシステム上のユーザーデータベースを参照することもできます。

システムはユーザー認証をチェックする為に有効なアカウントデータベースの設定済み一覧を保有していなければなりません。Red Hat Enterprise Linuxでは、認証設定ツールが、どのようなユーザーデータベースストアでも設定するために GUI とコマンドラインの両方のオプションを保有しています。

ローカルシステムはユーザー情報に様々な異なるデータストアを使用することができます。それには Lightweight Directory Access Protocol (LDAP)、Network Information Service (NIS)、および Winbind が含まれます。さらに LDAP と NIS の両データストアは Kerberos を使ってユーザーを認証できます。

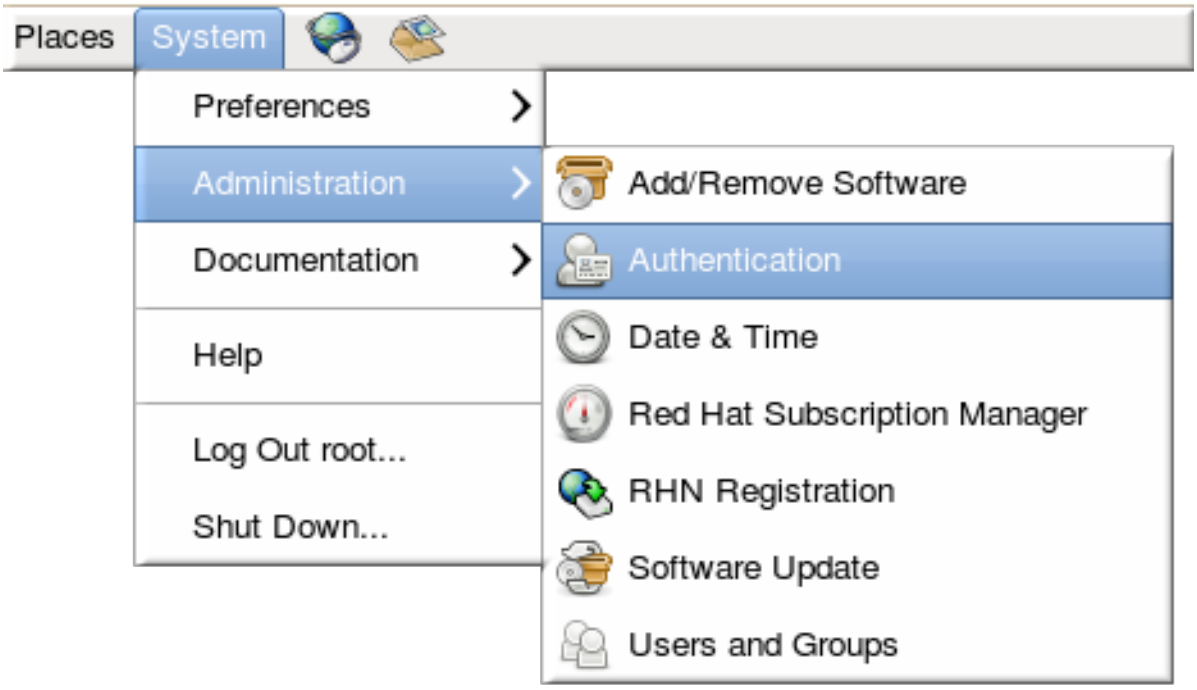


重要

インストール中にまたはセキュリティレベル設定ツールの使用で中レベルまたは高レベルのセキュリティがセットされると、ファイヤーウォールは NIS 認証を妨げます。ファイヤーウォールの詳細については 『Security Guide (セキュリティガイド)』 の「ファイヤーウォール」セクションをご覧下さい。

11.1.1. 認証設定ツール **UI** の起動

1. root でシステムにログインします。
2. **システム** を開きます
3. **管理** メニューを開きます
4. **認証** 項目を選択します



別の方法としては、**system-config-authentication** コマンドを実行します

重要

認証設定ツール UI を閉じると、すぐに変更が反映されます。

Authentication ダイアログボックスには 2 つの設定タブがあります:

- **識別と認証**、これは ID ストア (ユーザー ID と対応する認証情報が保存されているデータレポジトリ) として使用されるリソースを設定します。
- **高度なオプション**、これによりスマートカードや指紋などのパスワードや認証情報以外の認証方法が可能になります。

11.1.2. 認証目的での ID ストアの選択

識別と認証 のタブは、ユーザーに対する認証法をセットします。デフォルトではローカルシステムの認証を使用しますが、これはユーザーとそのパスワードがローカルシステムのアカウントに対してチェックされると言う意味です。Red Hat Enterprise Linux のマシンはユーザーと認証情報を含んだ LDAP、NIS、そして Winbind などの外部リソースを使用することも出来ます。

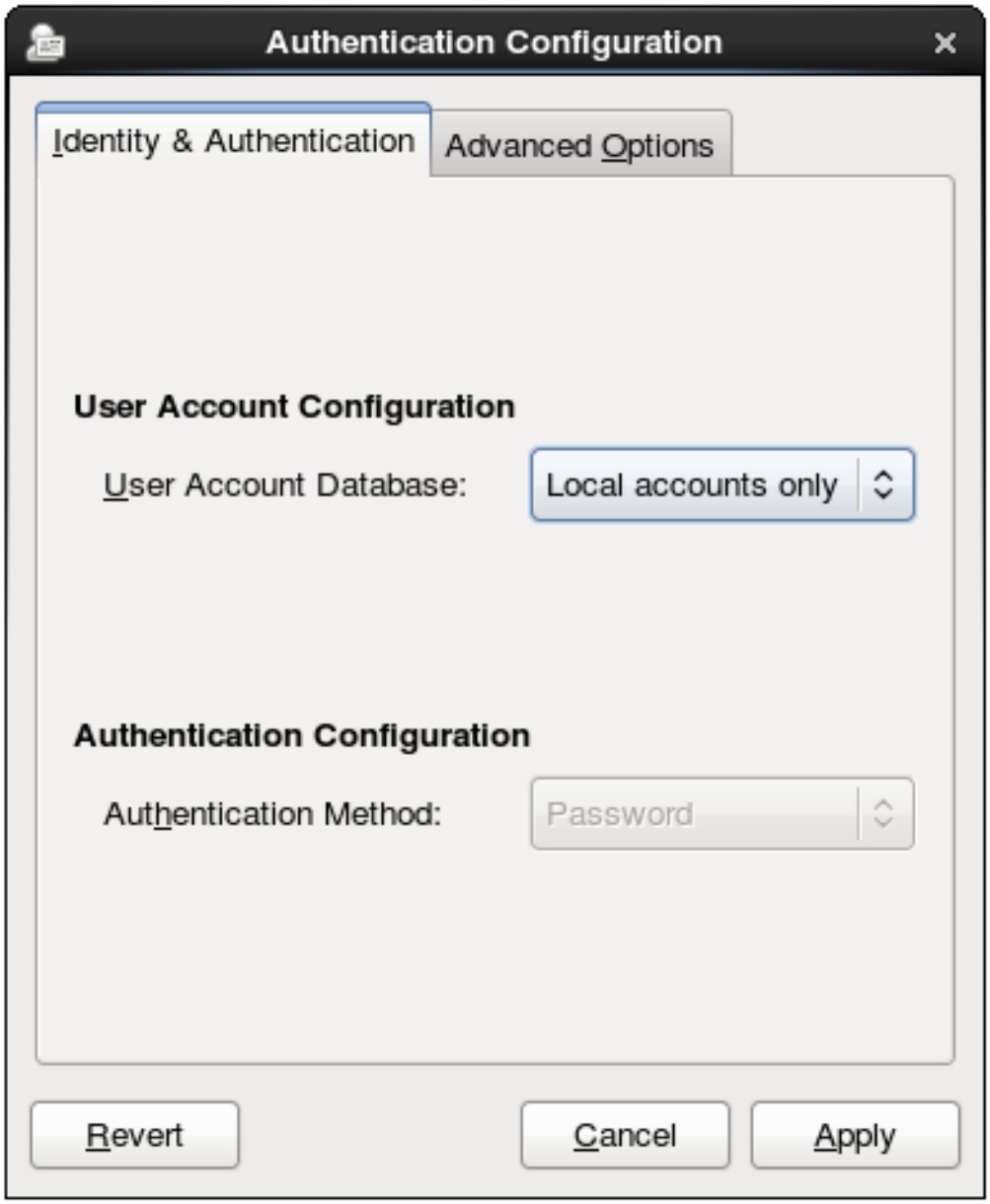


図11.1 ローカル認証

11.1.2.1. LDAP 認証の設定

ユーザーデータベースの LDAP サーバー設定には、**openldap-clients** パッケージか **sssd** パッケージを使用します。両パッケージともデフォルトでインストールされています。

1. 「認証設定ツール UI の起動 」 に説明してあるように、認証設定ツールを開きます。
2. **User Account Database (ユーザーアカウントデータベース)** ドロップダウンメニューから **LDAP** を選択します。

3. LDAP サーバーに接続するために必要となる情報をセットします。

   - **`LDAP Search Base DN (LDAP 検索データ DN)`** は root 接尾辞、またはユーザーディレクトリ用の**distinguished name (識別名)** (DN) を与えてくれます。識別/認証に使用されるすべてのユーザーエントリは、この親エントリの下に存在します。例えば、**ou=people,dc=example,dc=com**

     このフィールドはオプションです。特定されない場合は、LDAP サーバーの設定エントリーにある ***`namingContexts`*** および ***`defaultNamingContext`*** 属性を使って System Security Services Daemon (SSSD) が検索ベースの検出を図ります。

   - **`LDAP Server`** は、LDAP サーバーの URL を提供します。これには通常、**ldap://ldap.example.com:389** のように、LDAP サーバーのホスト名とポート番号の両方が必要になります。

     安全なプロトコルを **`ldaps://`** に入力することで、**`CA 証明書をダウンロードする`** ボタンが有効になります。

- **TLS を使用して接続を暗号化する** は、LDAP サーバーに対して接続を暗号化するために「Start TLS (TLS の開始)」を使用するかどうかをセットします。これにより、標準ポート上での安全な接続を可能にします。

  TLS を選択すると **CA 証明書をダウンロードする** ボタンを有効にするため、発行元の証明書権限から LDAP サーバー用の CA 証明書発行を取り込むようになります。CA 証明書は PEM (privacy enhanced mail プライバシー強化メール) 形式でなければなりません。

重要

サーバー URL が安全なプロトコル (**ldaps**) を使用する場合は、「**TLS を使用して接続を暗号化する** 」を選択**しないでください。** このオプションは Start TLS を使用し、これは標準ポートへの安全な接続を開始します。つまり、安全なポートが指定されている場合、Start TLS ではなく、SSL のようなプロトコルを使用する必要があります。

4. 認証のメソッドを選択します。LDAP は、単純なパスワード認証、あるいは Kerberos 認証を受け付けます。

   「LDAP または NIS 認証で Kerberos の使用」 に説明してあるようにして Kerberos の使用

   LDAP password オプションは、PAM アプリケーションを使用してLDAP 認証をします。このオプションは、LDAP サーバーに接続するために、安全な (**ldaps://**) URL か、または TLS オプションを必要とします。

11.1.2.2. NIS 認証の設定

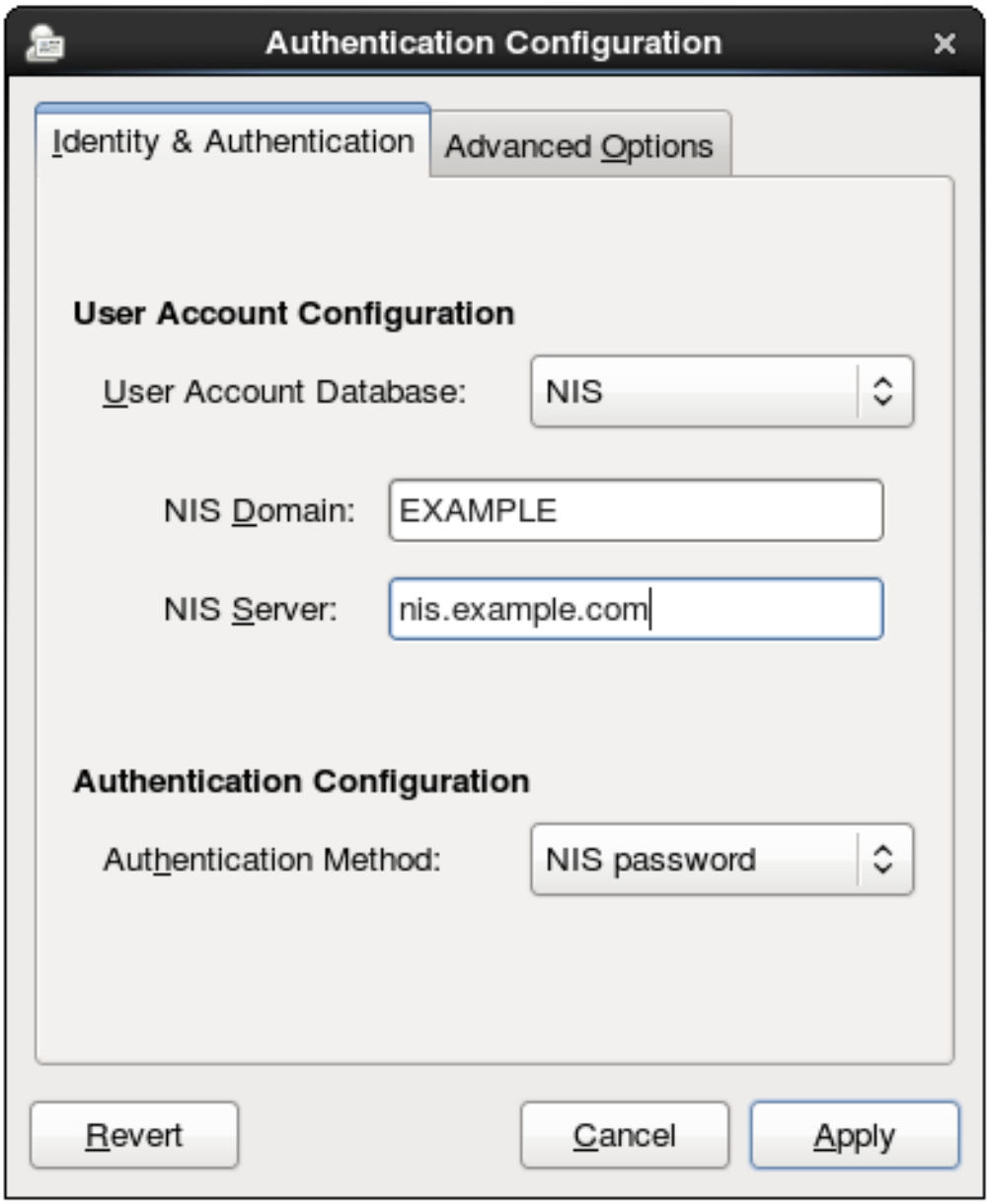
1. **ypbind** パッケージをインストールします。これは NIS サービスに必要なものですが、デフォルトではインストールされていません。

   ```
   [root@server ~]# yum install ypbind
   ```

   ypbind サービスがインストールされると、**portmap** サービスと **ypbind** サービスが開始され、起動時に開始できるようになります。

2. 「認証設定ツール UI の起動 」 に説明してあるように、認証設定ツールを開きます。

3. **User Account Database (ユーザーアカウントデータベース)** のドロップダウンメニュー内の **NIS** を選択します。



4. NIS サーバーに接続するための情報として NIS ドメイン名とサーバーのホスト名を設定します。NIS サーバーが指定されていない場合は **authconfig** デーモンが、NIS サーバーを求めてスキャンします。

5. 認証のメソッドを選択します。NIS は単純なパスワード認証、あるいは Kerberos 認証を受け付けます。

「LDAP または NIS 認証で Kerberos の使用」 に説明してあるようにして Kerberos の使用

NIS の詳細については、『セキュリティガイド』 の 「Securing NIS (NIS の保護)」 のセクションを参照してください。

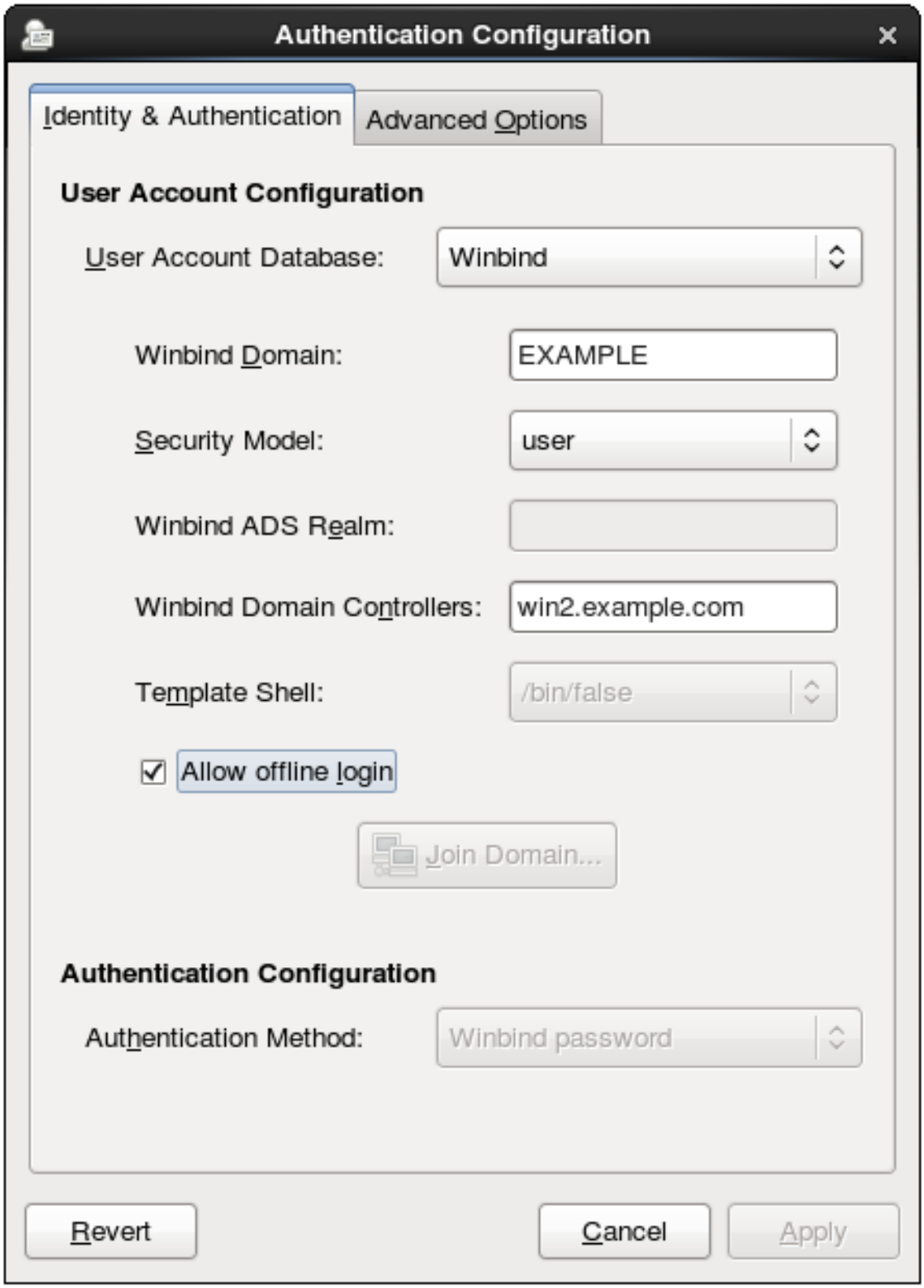
11.1.2.3. Winbind 認証の設定

1. **samba-winbind** パッケージをインストールします。これは、Samba サービスの Windows 統合機能に必要なものですが、デフォルトではインストールされていません。

```
[root@server ~]# yum install samba-winbind
```

2. 「認証設定ツール UI の起動 」 に説明してあるように、認証設定ツールを開きます。

3. **User Account Database** ドロップダウンメニュー内で **Winbind** を選択します。



4. Microsoft Active Directory ドメインコントローラーへ接続するために必要な情報を設定します。

   - **Winbind Domain (Winbind ドメイン)** は接続先の Windows ドメインを提供します。

     これは、**DOMAIN** のような Windows 2000 形式にします。

   - **セキュリティモデル** は Samba クライアントを使用するセキュリティモデルを設定します。**authconfig** は 4 つのタイプのセキュリティモデルをサポートします。

- **ads** は、Samba が Active Directory Server (ADS) レルムのドメインメンバーとして機能するよう設定します。このモードで操作を行うには、**krb5-server** パッケージがインストールされ、Kerberos が適切に設定されている必要があります。

- **domain** は、Windows Server が行うのと同様の方法で、Windows のプライマリまたはバックアップドメインコントローラを通して認証を行うことで、Samba にユーザー名/パスワードの検証をさせます。

- **server** は、別のサーバー (例えば Windows Server) を通して認証をすることでローカル Samba サーバーにユーザー名/パスワードの検証をさせます。サーバー認証に失敗した場合には、システムは代わりに **user** モードで認証を試みます。

- **user** はクライアントが有効なユーザー名とパスワードでログインすることを要求します。このモードは暗号化したパスワードをサポートしています。

  ユーザー名の形式は、**EXAMPLE\jsmith** のように **ドメイン\ユーザー** とする必要があります。

  注記

  あるユーザーが Windows ドメイン内に存在することを検証する際は、常に Windows 2000 スタイルの形式を使い、バックスラッシュ文字 (\) をエスケープします。例えば、

  ```
  [root@server ~]# getent passwd domain\\user
  DOMAIN\user:*:16777216:16777216:Name
  Surname:/home/DOMAIN/user:/bin/bash
  ```

  これがデフォルトのオプションです。

  - **Winbind ADS Realm** Samba サーバーが参加する Active Directory レルムを提供します。これは **ads** セキュリティモデルと一緒でのみ使用されます。

  - **Winbind ドメインコントローラ** は使用するドメインコントローラを提供します。ドメインコントローラの詳細情報については 「ドメインコントローラー」 を参照して下さい。

  - **テンプレートシェル** Windows のユーザーアカウント設定用に使用するログインシェルを設定します。

  - **オフラインログインを許可** は、ローカルキャッシュ内に認証情報が保存されるようにします。システムがオフラインの時にユーザーがシステムリソースの認証を試みるとキャッシュが参照されます。

winbindd サービスに関する詳細は 「Samba のデーモンと関連サービス」 を参照してください。

11.1.2.4. LDAP または NIS 認証で Kerberos の使用

LDAP と NIS の認証ストアは両方とも Kerberos の認証メソッドをサポートします。Kerberos の利用にはいくつかの利便性があります:

- 標準ポート上での接続を許可しながら、通信にセキュリティレイヤーを使用します。

- オフラインログインを可能にする SSSD を使用した認証情報キャッシングを自動的に使用します。

Kerberos 認証の使用には、**krb5-libs** と **krb5-workstation** の両パッケージが必要になります。

認証の方法 のドロップダウンメニューから **Kerberos パスワード** を選ぶと、自動的に Kerberos レルムへの接続に必要なフィールドが開きます。

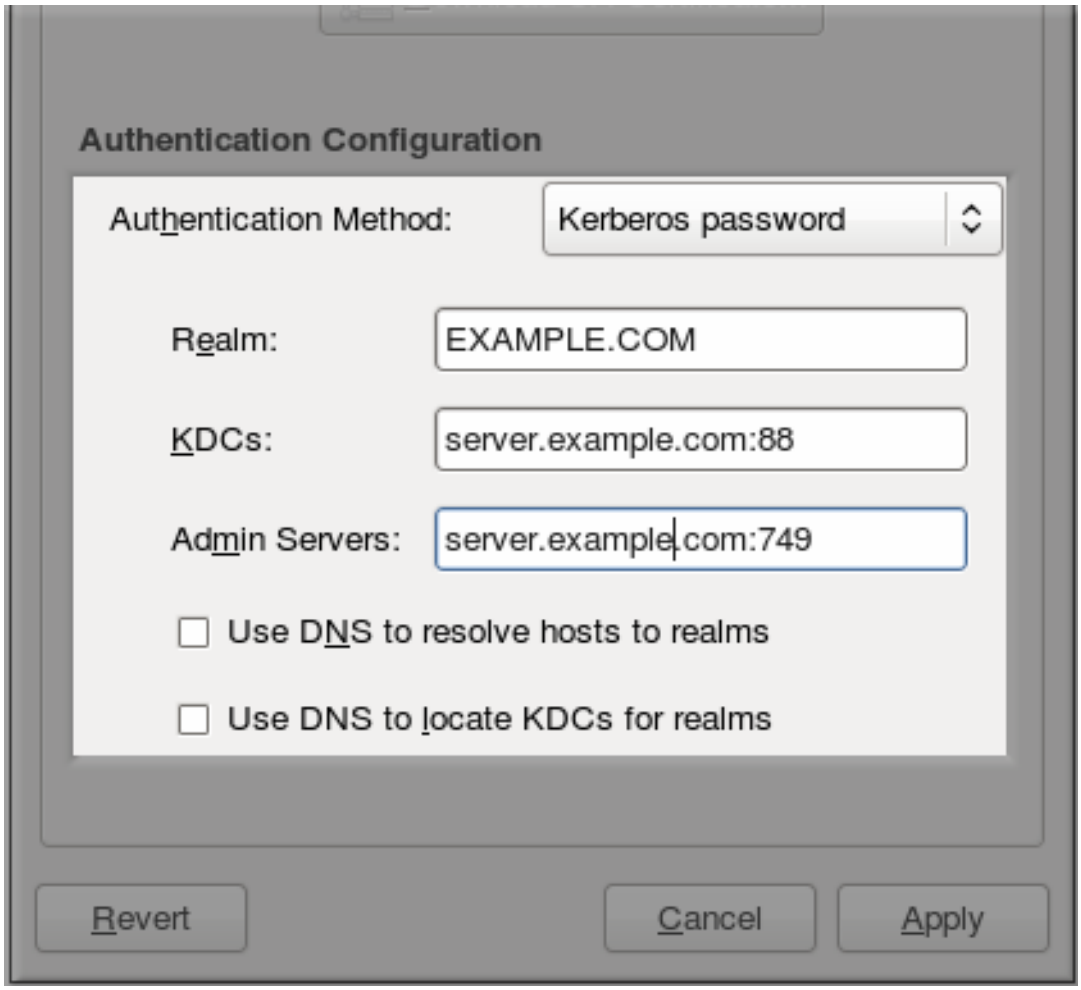


図11.2 Kerberos フィールド

- **レルム** Kerberos サーバー用のレルムに名前を与えます。レルムとは、 1 つまたはそれ以上の KDC と多数のクライアントで構成される、Kerberos を使用するネットワークのことです。
- **KDCs** は、Kerberos チケットを発行するサーバーのコンマ区切りの一覧を提供します。
- **Admin Servers (管理サーバー)** は、レルム内で **kadmind** プロセスを実行している管理サーバーの一覧を提供します。
- オプションとして、DNS を使用してサーバーのホスト名を解決してレルム内で追加の KDC を見つけることができます。

Kerberos の詳細については、Red Hat Enterprise Linux 6 の 『Managing Single Sign-On and Smart Cards (シングルサインオンとスマートカード)』 ガイドにある「Using Kerberos (Kerberos の使用)」のセクションを参照して下さい。

11.1.3. 代替認証機能の設定

認証設定ツールも、ID ストアとは別に認証動作に関連したセッティングを設定します。これには、まったく異なる認証メソッド (指紋スキャンとスマートカード) やローカル認証ルールが含まれます。これらの認証オプションは、**高度なオプション** タブで設定します。

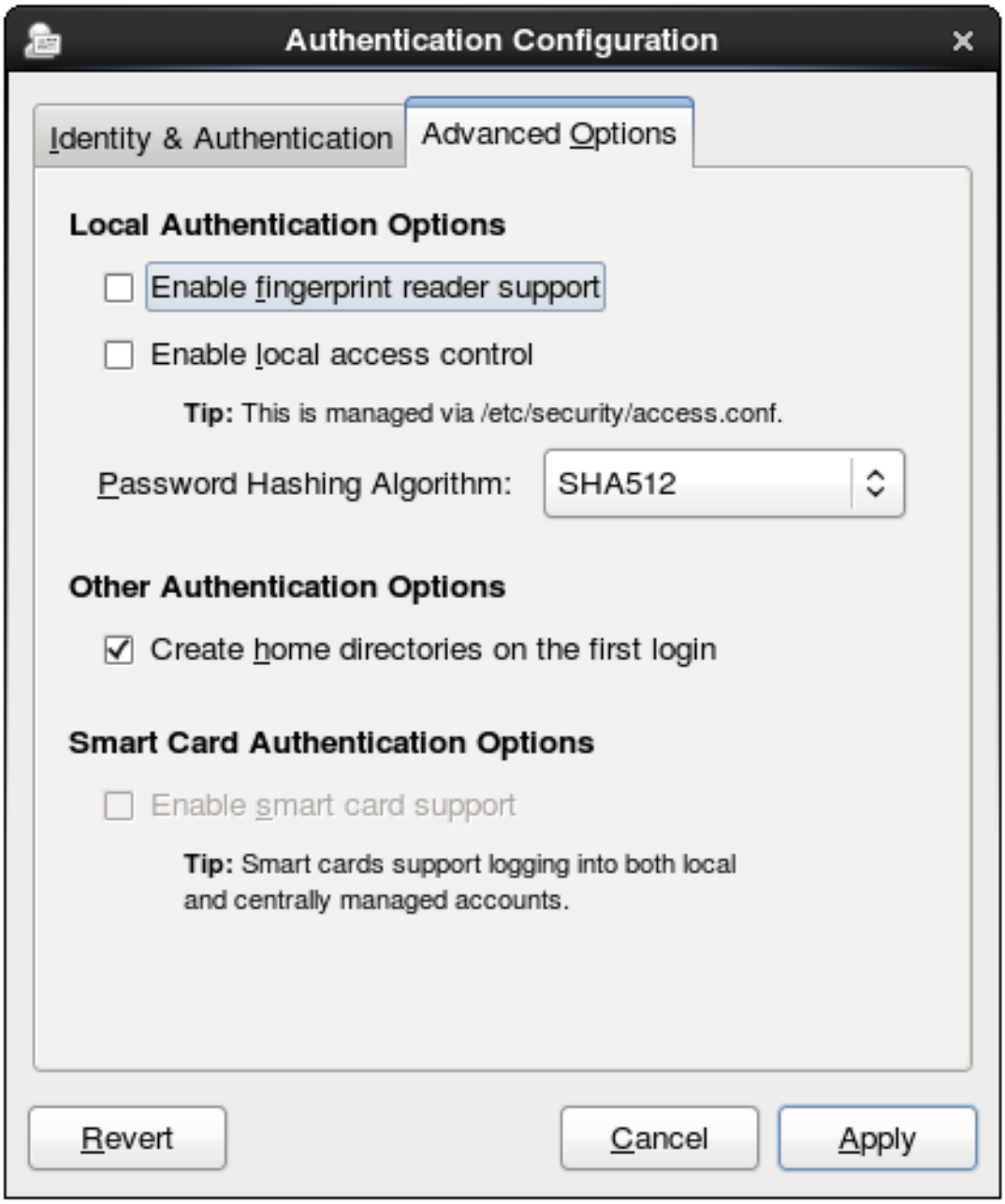


図11.3 高度なオプション

11.1.3.1. 指紋認証の使用

適切なハードウェアが利用可能である場合は、他の認証情報に加えて **指紋読み取りサポートを有効にする** オプションでローカルユーザーの認証に指紋スキャンが使用可能になります。

11.1.3.2. ローカル認証パラメータのセッティング

ローカルシステム上で認証動作を定義する **Local Authentication Options (ローカル認証オプション)** エリアには２つのオプションがあります:

- **Enable local access control (ローカルアクセス制御を有効にする)** は、**/etc/security/access.conf** ファイルに対してローカルユーザー認証ルールをチェックするようの指示します。
- **Password Hashing Algorithm (パスワードハッシュアルゴリズム)** は、ローカル保存のパスワードを暗号化するために使用するハッシュアルゴリズムをセットします。

11.1.3.3. スマートカード認証の有効化

適切なスマートカード読み取りが利用できる時には、システムは認証のために他のユーザー認証情報の代わりにスマートカード (または **トークン**) を受け付けることができます。

Enable smart card support (スマートカードサポートを有効にする) オプションが選択されると、スマートカード認証の動作を定義付けできます:

- **Card Removal Action (カード削除のアクション)** はシステムに対してアクティブセッション中にカード読み取りからカードが取り出された時に対応の方法を伝えます。システムは、取り出しを無視して通常どおりにユーザーにリソースへのアクセスを与えるか、またはスマートカードが再供給されるまでシステムは直ちにロックするかのどちらかです。

- **Require smart card login (スマートカードログインを要求)** は、スマートカードがログインで必要かどうか、または単にログインで許可されるかどうかをセットします。このオプションが選択されていると、認証に於ける他のメソッドはすべてすぐにブロックされます。

警告

スマートカードを使用してシステムに正しく認証をさせるまではこのオプションは選択しないで下さい。

スマートカードの使用には、**pam_pkcs11** パスワードが必要になります。

11.1.3.4. ユーザーホームディレクトリの作成

利用者の最初のログイン時にホームディレクトリを作成する オプションを選択するとユーザーが初めてログインする時に自動的にホームディレクトリが作成されます。

このオプションは、LDAP などアカウントを集中的に管理している場合に便利です。しかし、ユーザーのホームディレクトリー管理に自動マウントのようなシステムを使用している場合は、このオプションを選択しないでください。

11.1.4. コマンドライン上で 認証の設定

authconfig コマンドラインツールは、スクリプトに渡されたセッティングに応じてシステム認証に必要な設定ファイルとサービスのすべてを更新します。UI を通じてセットできるすべての識別子と認証設定のオプションの認可に加えて、**authconfig** ツールはバックアップファイルとキックスタートファイルの作成にも使用できます。

authconfig オプションの総合一覧についてはヘルプの出力と man ページを参照してください。

11.1.4.1. authconfig の使用へのヒント

authconfig を実行する時点で覚えておくことがいくつかあります:

- すべてのコマンドで **--update** か **--test** のオプションを使用します。これらのオプションのいずれかがコマンドを正しく実行するために必要になります。**--update** を使用すると設定の変更を書き込みます。**--test** は標準出力への変更を表示しますが、設定への変更を適用しません。

- それぞれの「enable」 オプションには対応する「disable」 オプションがあります。

11.1.4.2. LDAP ユーザーストアの設定

LDAP 識別子ストアを使用するには、**--enableldap** を使用します。LDAP を認証ソースとして使用するには、**--enableldapauth** を使用して、それから LDAP サーバー名、ユーザー接尾辞用のベース DN、及び (オプションとして) TLS を使用するかどうか、などの必須となる接続情報を使用します。認証設定 UI 経由では利用できないユーザーエントリ用の RFC 2307bis スキーマに対して、**authconfig** コマンドはそれを有効/無効にするオプションも持っています。

確実に完全な形の LDAP URL を使用して下さい。プロトコル (**ldap** または **ldaps**) とポート番号を含みます。**--enableldaptls** オプションと一緒にセキュア LDAP URL (**ldaps**) を使用**しないで下さい。**

```
authconfig --enableldap --enableldapauth --
ldapserver=ldap://ldap.example.com:389,ldap://ldap2.example.com:389 --
ldapbasedn="ou=people,dc=example,dc=com" --enableldaptls --
ldaploadcacert=https://ca.server.example.com/caCert.crt --update
```

LDAP のパスワード認証用に**--ldapauth** を使う代わりに、LDAP ユーザーストアで Kerberos を使用することができます。これらのオプションは、「Kerberos 認証の設定」 で説明してあります。

11.1.4.3. NIS ユーザーストアの設定

NIS 識別子ストアを使用するには、**--enablenis** を使います。Kerberos 認証 (「Kerberos 認証の設定」) を使用するように Kerberos パラメータが明示的にセットされていない限りは、これが自動的に NIS 認証を使用します。唯一のパラメータは NIS サーバーと NIS ドメインを識別することになります。これらが使用されない場合は、**authconfig** サービスが NIS サーバーを求めてネットワークをスキャンします。

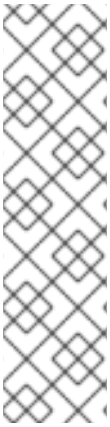
```
authconfig --enablenis --nisdomain=EXAMPLE --nisserver=nis.example.com --
update
```

11.1.4.4. Winbind ユーザーストアの設定

Windows のドメインは数種の異なるセキュリティモデルを持っており、ドメインで使用されるセキュリティモデルがローカルシステムの認証設定を決定します。

ユーザーとサーバーのセキュリティモデルには、Winbind 設定はドメイン (またはワークグループ) の名前とドメインコントローラホスト名のみを必要とします。

```
authconfig --enablewinbind --enablewinbindauth --smbsecurity=user|server
--enablewinbindoffline --smbservers=ad.example.com --smbworkgroup=EXAMPLE
--update
```



注記

ユーザー名の形式は、**EXAMPLE\jsmith** のように **ドメイン\ユーザー** とする必要があります。

あるユーザーが Windows ドメイン内に存在することを検証する際は、常に Windows 2000 スタイルの形式を使い、バックスラッシュ文字 (\) をエスケープします。例えば、

```
[root@server ~]# getent passwd domain\\user
DOMAIN\user:*:16777216:16777216:Name
Surname:/home/DOMAIN/user:/bin/bash
```

ads と domain ドメインのセキュリティモデルには、Winbind 設定はテンプレートシェルとレルム (ads のみ) 用に追加の設定を可能にします。例えば:

```
authconfig --enablewinbind --enablewinbindauth --smbsecurity ads  --
enablewinbindoffline --smbservers=ad.example.com --smbworkgroup=EXAMPLE --
smbrealm EXAMPLE.COM --winbindtemplateshell=/bin/sh --update
```

名前形式、ドメイン名をユーザー名と一緒に要求するかどうか、及び UID の範囲など、Windows ベースの認証の設定と Windows ユーザーアカウントの情報のために他のオプションが数多く存在します。これらのオプションは **authconfig** ヘルプ内に一覧表示してあります。

11.1.4.5. Kerberos 認証の設定

LDAP と NIS は両方ともそれらのネイティブな認証メカニズムの場所で Kerberos 認証の使用を許可します。Kerberos 認証の使用には最低でも、レルム、KDC、及び管理用サーバーの指定を必要とします。クライアント名を解決して追加の管理用サーバーを見つける目的で DNS を使用するオプションもあります。

```
authconfig NIS or LDAP options --enablekrb5 --krb5realm EXAMPLE --krb5kdc
kdc.example.com:88,server.example.com:88 --krb5adminserver
server.example.com:749 --enablekrb5kdcdns --enablekrb5realmdns --update
```

11.1.4.6. ローカル認証セッティングの設定

認証設定ツールは、ホームディレクトリの作成やパスワードアルゴリズムのセッティングや認証などのセキュリティに関連したいくつかのユーザーセッティングも制御できます。これらのセッティングは、識別子/ユーザーストアのセッティングとは独立して実行されます。

例えば、ユーザーホームディレクトリを作成するには:

```
authconfig --enablemkhomedir --update
```

ユーザーパスワードを暗号化するために使用されているハッシュアルゴリズムをセット、または変更するには:

```
authconfig --passalgo=sha512 --update
```

11.1.4.7. 指紋認証の設定

指紋読み取りのサポートを有効にするオプションが 1 つあります。このオプションは単独でも使用できますが、LDAP ユーザーストアのような他の **authconfig** セッティングと一緒でも使用できます。

```
[root@server ~]# authconfig --enablefingerprint --update
```

11.1.4.8. スマートカード認証の設定

システムでスマートカードを使用する際に必要なのは **--enablesmartcard** オプションのセットのみです。

```
[root@server ~]# authconfig --enablesmartcard --update
```

スマートカードには、デフォルトのスマートカードモジュールの変更、スマートカードが取り出された時のシステムの動作のセッティング、ログイン用にスマートカードの要求など、他の設定オプションがあります。

例えば以下のコマンドは、スマートカードが取り出された場合、システムに対してユーザーをすぐに締め出すように指示します (セッティングが 1 の場合は、スマートカードが取り出されても、これを無視します)。

```
[root@server ~]# authconfig --enablesmartcard --smartcardaction=0 --update
```

スマートカード認証が正常に設定され、テストが終わると、単純なパスワードベースの認証ではなく、スマートカード認証をユーザーに要求するようにシステムを設定することができます。

```
[root@server ~]# authconfig --enablerequiresmartcard --update
```

警告

スマートカードの使用でシステムへの認証が正しく終了するまで**--enablerequiresmartcard** オプションは使用しないで下さい。使用するとユーザーはシステムにログインできない可能性があります。

11.1.4.9. キックスタートと設定ファイルの管理

--update オプションは、設定に変更を受けたすべての設定ファイルを更新します。少しだけ異なる動作をする数種の代替オプションが存在します:

- **--kickstart** は、更新した設定をキックスタートファイルに書き込みます。
- **--test** は、変更後の設定全域を標準出力に表示しますが、設定ファイルはどれも編集しません。

更には、**authconfig** はバックアップと以前の設定の復元に使用できます。すべてのアーカイブは **/var/lib/authconfig/** ディレクトリ内の特有のサブディレクトリに保存されます。例えば、**--savebackup** オプションはバックアップディレクトリを**2011-07-01** などとして提供します:

```
[root@server ~]# authconfig --savebackup=2011-07-01
```

これが、**/var/lib/authconfig/backup-2011-07-01** ディレクトリ以下の認証設定ファイルのすべてをバックアップします。

保存したバックアップはどれも **--restorebackup** オプションで手動保存した設定の名前を入力して、設定の復元に使用できます。

```
[root@server ~]# authconfig --restorebackup=2011-07-01
```

また、**authconfig** は、変更を (**--update** オプションで) 適用する前に、自動的に設定のバックアップを作成します。設定は、**--restorelastbackup** オプションを使用することで、正確なバックアップをしていせずに、最新の自動バックアップから復元できます。

11.1.5. カスタムホームディレクトリの使用

LDAP ユーザーのディレクトリが **/home** 以外の場所にあり、かつユーザーの初回ログイン時にシステムがホームディレクトリを作成するように設定されている場合は、これらのディレクトリは間違ったパーミッションで作成されてしまいます。

1. **/home** ディレクトリからの正しい SELinux コンテキストとパーミッションを、ローカルシステムで作成されたホームディレクトリに適用します。例えば:

   ```
   [root@server ~]# semanage fcontext -a -e /home /home/locale
   ```

2. **oddjob-mkhomedir** パッケージをシステムにインストールします。

   このパッケージは、**pam_oddjob_mkhomedir.so** ライブラリを提供します。認証設定ツールは、このライブラリを使用してホームディレクトリを作成します。デフォルトの **pam_mkhomedir.so** ライブラリとは異なり、**pam_oddjob_mkhomedir.so** ライブラリは SELinux ラベルを作成できます。

   認証設定ツールは、**pam_oddjob_mkhomedir.so** ライブラリが利用可能な場合には、自動的に使用します。利用可能でない場合には、デフォルトの **pam_mkhomedir.so** ライブラリを使用します。

3. **oddjobd** サービスが稼働していることを確認して下さい。

4. 「代替認証機能の設定」 に示してあるように、認証設定ツールを再実行して、ホームディレクトリを有効にします。

ホームディレクトリの設定変更前にホームディレクトリが作成されている場合、正しいパーミッションとSELinux コンテキストが存在することになります。例えば:

```
[root@server ~]# semanage fcontext -a -e /home /home/locale
# restorecon -R -v /home/locale
```

11.2. SSSDでの認証情報の使用とキャッシング

System Security Services Daemon (SSSD) は、異なる ID および認証プロバイダーへのアクセスを提供します。

11.2.1. SSSD について

ほとんどのシステム認証はローカルで設定されているので、サービスはユーザーと認証情報の決定にローカルのユーザーストアをチェックする必要があります。SSSD は、ローカルサービスによる SSSD 内のローカルキャッシュのチェックを可能にしますが、このキャッシュは、LDAP ディレクトリーや ID 管理ドメイン、アクティブディレクトリー、さらには Kerberos レルムなど、様々な **リモートの** ID プロバイダーから取得されたものである可能性があります。

SSSD はこれらのユーザーや認証情報もキャッシュするので、ローカルシステム **または** ID プロバイダーがオフラインになっても、サービスはユーザーの認証情報を利用して確認できます。

SSSD は、ローカルのクライアントと設定済みのデータストアとの橋渡し役です。この関係は、管理者に多くの利益をもたらします。

- **識別/認証サーバー上の負荷の低減化。** すべての個別クライアントサービスが認証サービスに直接連絡するのではなく、すべてのローカルクライアントは SSSD に連絡して、SSSD が識別サーバーに接続するか、またはそのキャッシュをチェックします。
- **オフライン認証の許可。** オプションとして、SSSD はリモートサービスから取り込むユーザー識別子と認証情報のキャッシュを維持することができます。これにより、ユーザーは識別サーバーがオフラインであったり、ローカルマシンがオフラインであってもリソースを正しく認証することができます。
- **単独のユーザーアカウントの使用。** リモートユーザーは多くの場合、２つ (またはそれ以上) のユーザーアカウントを持っています。１つが自身のローカルマシン用で、もう１つが組織のシステム用などです。仮想プライベートネットワーク (VPN) に接続するためにそれぞれにアカウントが必要になります。しかし、SSSD がキャッシングとオフライン認証をサポートするため、リモートユーザーは、ローカルマシンに認証させることだけで、SSSD がそのネットワーク認証情報を維持するのでネットワークリソースに接続できます。

その他のリソース

本章では、SSSD のサービスおよびドメインの基本的な設定を説明していますが、包括的なリソースではありません。SSSD の各機能エリアにはそれぞれ利用可能な他の設定オプションがあります。特定の機能エリアの man ページをチェックして、完全なオプション一覧を取得してください。

一般的な man ページの一覧は 表11.1「SSSD Man ページのサンプル」 にあります。SSSD man ページの全一覧は、**sssd(8)** man ページの "See Also" を参照してください。

表11.1 SSSD Man ページのサンプル

機能エリア	Man ページ
全般設定	sssd.conf(8)
sudo サービス	sssd-sudo
LDAP ドメイン	sssd-ldap
アクティブディレクトリードメイン	sssd-ad sssd-ldap

機能エリア	Man ページ
ID 管理 (IdM または IPA) ドメイン	sssd-ipa sssd-ldap
ドメインの Kerberos 認証	sssd-krb5
オープン SSH キー	sss_ssh_authorizedkeys sss_ssh_knownhostsproxy
キャッシュメンテナンス	sss_cache (cleanup) sss_useradd, sss_usermod, sss_userdel, sss_seed (ユーザーキャッシュエントリー管理)

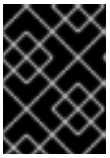
11.2.2. sssd.conf ファイルの設定

SSSD のサービスとドメインは `.conf` ファイル内で設定されています。デフォルトでは `/etc/sssd/sssd.conf` ですが、SSSD はインストール後に設定されないため、このファイルは手動で作成、設定する必要があります。

11.2.2.1. sssd.conf ファイルの作成

SSSD 設定ファイルには、3 つのパートがあります。

- **[sssd]**、これは全般的な SSSD プロセスとオペレーションの設定のためです。基本的に、設定済みサービスとドメイン、それぞれの設定パラメーターを一覧表示します。
- **[service_name]**、これは 「SSSD およびシステムサービス」 にあるように、対応しているシステムサービスの設定オプション用です。
- **[domain_type/DOMAIN_NAME]**、これは設定済み ID プロバイダーの設定オプション用です。



重要

サービスはオプションですが、SSSD サービスの開始前に少なくとも 1 つの ID プロバイダードメインを設定する必要があります。

例11.1 単純な sssd.conf ファイル

```
[sssd]
domains = LOCAL
services = nss
config_file_version = 2
```

```
[nss]
filter_groups = root
filter_users = root

[domain/LOCAL]
id_provider = local
auth_provider = local
access_provider = permit
```

[sssd] セクションには、以下の重要な 3 つのパラメーターがあります。

- **domains** は、**sssd.conf** で設定された全ドメインを一覧表示します。これは、SSSD が ID プロバイダーとして使用するものです。**domains** キーにドメインがない場合、設定セクションがあったとしても、SSSD は使用しません。
- **services** は、**sssd.conf** で設定され、SSSD を使用する全システムサービスを一覧表示します。SSSD が開始されると、対応する SSSD サービスが設定済みの各システムサービス用に開始されます。**services** キーにサービスがない場合、設定セクションがあったとしても、SSSD は使用しません。
- **config_file_version** は、ファイル形式の期待値を設定する設定ファイルのバージョンを設定します。ここではバージョン 2 で、最近の SSSD 全バージョン用になっています。

注記

sssd.conf ファイルでサービスやドメインが設定されても、それらが **[sssd]** セクションの **services** もしくは **domains** パラメーターで表示されていなければ、SSSD はそのサービスやドメインと対話しません。

他の設定パラメーターは、**sssd.conf** man ページに一覧表示されています。

サービスおよびドメインパラメーターは、本章の各設定セクションとそれらの man ページで説明されています。

11.2.2.2. カスタム設定ファイルの使用

デフォルトでは、**sssd** プロセスは設定ファイルが **/etc/sssd/sssd.conf** であることを前提としています。

代わりのファイルは、**-c** オプションの **sssd** コマンドを使って SSSD に渡すことができます。

```
[root@server ~]# sssd -c /etc/sssd/customfile.conf --daemon
```

11.2.3. SSSD の起動と停止

重要

初めて SSSD を開始する前に、ドメインを少なくとも 1 つ設定します。「SSSD および ID プロバイダー (ドメイン)」 をご覧下さい。

service コマンドまたは **/etc/init.d/sssd** スクリプトのいずれかで SSSD を開始できます。例えば:

```
[root@server ~]# service sssd start
```

デフォルトでは、SSSD は自動開始に設定されていません。この動作は、2 つの方法で変更できます。

- **authconfig** コマンドによる SSSD の有効化

  ```
  [root@server ~]# authconfig --enablesssd --enablesssdauth --update
  ```

- **chkconfig** コマンドを使用して、SSSD プロセスをスタートリストに追加する

  ```
  [root@server ~]# chkconfig sssd on
  ```

11.2.4. SSSD およびシステムサービス

SSSD とその関連サービスは **sssd.conf** ファイル内で設定されています。**[sssd]** セクションはアクティブなサービスも一覧表示し、それらは **sssd** が **services** 指示文の中で開始する時に、開始するものです。

SSSD は、いくつかのシステムサービスに認証情報キャッシュを提供できます。

- **sssd_nss** モジュールからのネームサービス要求に応答する Name Service Switch （NSS) プロバイダーサービス。これは、SSSD 設定の **[nss]** セクション内で設定されます。

  「サービスの設定: NSS」 で説明されています。

- **sssd_pam** モジュールを介して PAM 会話を管理する PAM プロバイダーサービス。これは設定の **[pam]** セクション内で設定されます。

  これは、「サービスの設定: PAM」 で説明されています。

- SSSD が **known_hosts** ファイルや他のキー関連の設定を管理する方法を定義する SSH プロバイダーサービス。OpenSSH による SSSD の使用については、「サービスの設定: OpenSSH およびキャッシュされたキー」 で説明されています。

- 設定済みマウントの場所を取得するために LDAP サーバーに接続する **autofs** プロバイダーサービス。これは、設定ファイル内の **[domain/NAME]** セクションで、LDAP ID プロバイダーの一部として設定されます。

  これは、「サービスの設定: autofs」 で説明されています。

- 設定済み **sudo** ポリシーを取得するために LDAP サーバーに接続するプロバイダーサービス。これは、設定ファイル内の **[domain/NAME]** セクションで、LDAP ID プロバイダーの一部として設定されます。

  これは、「サービスの設定: sudo」 で説明されています。

- SSSD が Kerberos と動作してアクティブディレクトリーのユーザーおよびグループを管理する方法を定義する PAC レスポンダーサービス。これは 「ドメインの作成: Active Directory」 で説明されているように、特にドメインのアクティブディレクトリー ID プロバイダー管理の一部です。

11.2.5. サービスの設定: NSS

SSSD は、NSS モジュールの 1 つである **sssd_nss** を提供します。これがSSSD を使用してユーザー情報を取り込むようにシステムに指示します。NSS 設定は SSSD モジュールへの参照を含んでいる必要があり、そして SSSD 設定は SSSD が NSS に対応する方法をセットします。

11.2.5.1. NSS サービスマップおよび SSSD について

Name Service Switch (NSS) は、多くの設定および名前解決サービスをルックアップするサービスの中央設定を提供します。NSSは、設定ソースのシステム ID およびサービスのマッピング方法の一つを提供します。

SSSDは、数種類の NSS マップのプロバイダーサービスとして、NSS と機能します。

- パスワード (**passwd**)
- ユーザーグループ (**shadow**)
- グループ (**groups**)
- ネットグループ (**netgroups**)
- サービス (**services**)

11.2.5.2. SSSD を使用する NSS サービスの設定

NSS は、サービスマップの一部またはすべてにおいて複数の ID および設定プロバイダーを使用できます。デフォルトでは、サービス用のシステムファイルを使用します。SSSD を含めるには、希望するサービスの種類に **nss_sss** モジュールを含める必要があります。

1. 認証設定ツールを使って SSSD を有効にします。これで **nsswitch.conf** ファイルは自動的に SSSD をプロバイダーとして使用するように設定されます。

   ```
   [root@server ~]# authconfig --enablesssd --update
   ```

   これでパスワード、シャドウ、ネットグループサービスマップが SSSD モジュールを使用するように自動的に設定されます。

   ```
   passwd:     files sss
   shadow:     files sss
   group:      files sss

   netgroup:   files sss
   ```

2. SSSD で **authconfig** が有効化されていると、サービスマップはデフォルトでは有効化されません。このマップを含めるには、**nsswitch.conf** ファイルを開いて **services** マップに **sss** モジュールを追加します。

   ```
   [root@server ~]# vim /etc/nsswitch.conf

   ...
   services: file sss
   ...
   ```

11.2.5.3. NSS と機能する SSSD の設定

NSS リクエストに応えるために SSSD が使用するオプションおよび設定は、SSSD 設定ファイルの **[nss]** サービスセクションで設定されます。

1. **sssd.conf** ファイルを開きます。

```
[root@server ~]# vim /etc/sssd/sssd.conf
```

2. NSS が、SSSD と一緒に機能するサービスの 1 つとして一覧表示されていることを確認します。

```
[sssd]
config_file_version = 2
reconnection_retries = 3
sbus_timeout = 30
services = nss, pam
```

3. **[nss]** セクションで NSS パラメータのいずれかを変更します。これらは 表11.2「SSSD [nss] 設定のパラメータ」 に一覧表示されています。

```
[nss]
filter_groups = root
filter_users = root
reconnection_retries = 3
entry_cache_timeout = 300
entry_cache_nowait_percentage = 75
```

4. SSSD を再開始します。

```
[root@server ~]# service sssd restart
```

表11.2 SSSD [nss] 設定のパラメータ

パラメーター	値の形式	詳細
enum_cache_timeout	整数	**sssd_nss** がすべてのユーザーに関する情報 (列挙) の要求をキャッシュすべき時間の長さを秒で指定します。

パラメーター	値の形式	詳細
entry_cache_nowait_percentage	整数	キャッシュをリフレッシュするまでの **sssd_nss** がキャッシュ化したエントリを返す時間の長さを指定します。これをゼロ (**0**) にセットすると、エントリキャッシュのリフレッシュを無効にします。 次回の更新までの時間が次回の待機間隔の一定のパーセンテージである時、要求される場合は、これが背景で自動的にエントリを更新するようにエントリキャッシュを設定します。例えば、待機間隔が300秒であり、キャッシュのパーセンテージが 75であるならば、エントリキャッシュは、この待機間隔の 75%となる 225秒で要求が来る時にリフレッシュを開始します。 このオプションの有効な値は、0-99 で、**entry_cache_timeout** 値を土台にしたパーセンテージをセットします。デフォルト値は、50 %です。
entry_negative_timeout	整数	**sssd_nss** が **ネガティブ** キャッシュヒットをキャッシュすべき時間の長さを秒で指定します。ネガティブキャッシュヒットとは、存在しないエントリも含む無効なデータベースエントリ用のクエリです。
filter_users, filter_groups	文字列	特定のユーザーを NSS データベースからのフェッチ対象から除外するように SSSD に伝えます。これは、特に **root** のようなシステムアカウントに有用です。
filter_users_in_groups	ブール値	グループルックアップを実行している時に、**filter_users** 一覧に表示されているユーザーがグループメンバーシップに表れるかどうかをセットします。**FALSE** にセットすると、グループルックアップは、そのグループのメンバーであるすべてのユーザーを返します。指定されていない場合は、値はデフォルトの **true** となり、グループメンバー一覧をにフィルターをかけます。

パラメーター	値の形式	詳細
debug_level	0 - 9 の整数	デバッグログレベルを設定します。

11.2.6. サービスの設定: PAM

警告

PAM 設定ファイルに誤りがあると、ユーザーはシステムから完全にロックアウトされてしまいます。変更を行う前に、設定ファイルは必ずバックアップし、変更を元に戻すことができるようにセッションはオープンのままにします。

SSSD は PAM モジュールの 1 つである **sssd_pam** を提供します。これはSSSD を使用してユーザー情報を取り込むようにシステムに指示します。PAM 設定は SSSD モジュールへの参照を含んでいる必要があり、そして SSSD 設定は SSSD が PAM に対応する方法をセットします。

PAM サービスを設定するには:

1. **authconfig** を使用してシステム認証のために SSSD を有効にします。

```
# authconfig --update --enablesssd --enablesssdauth
```

これが自動的に PAM 設定を更新して、すべての SSSD モジュールを参照します:

```
#%PAM-1.0
# This file is auto-generated.
# User changes will be destroyed the next time authconfig is run.
auth        required      pam_env.so
auth        sufficient    pam_unix.so nullok try_first_pass
auth        requisite     pam_succeed_if.so uid >= 500 quiet
auth sufficient pam_sss.so use_first_pass
auth        required      pam_deny.so

account     required      pam_unix.so
account     sufficient    pam_localuser.so
account     sufficient    pam_succeed_if.so uid < 500 quiet
account [default=bad success=ok user_unknown=ignore] pam_sss.so
account     required      pam_permit.so

password    requisite     pam_cracklib.so try_first_pass retry=3
password    sufficient    pam_unix.so sha512 shadow nullok
try_first_pass use_authtok
password sufficient pam_sss.so use_authtok
password    required      pam_deny.so

session     optional      pam_keyinit.so revoke
session     required      pam_limits.so
```

```
session     [success=1 default=ignore] pam_succeed_if.so service in
crond quiet use_uid
session sufficient pam_sss.so
session     required      pam_unix.so
```

これらのモジュールは必要に応じて **include** ステートメントにセットできます。

2. **sssd.conf** ファイルを開きます。

```
# vim /etc/sssd/sssd.conf
```

3. PAM が、SSSD と一緒に機能するサービスの 1 つとして一覧表示されていることを確認します。

```
[sssd]
config_file_version = 2
reconnection_retries = 3
sbus_timeout = 30
services = nss, pam
```

4. **[pam]** セクションで、PAM パラメータのいずれかを変更します。それらは、表11.3「SSSD [pam] 設定パラメータ」 内に一覧表示されています。

```
[pam]
reconnection_retries = 3
offline_credentials_expiration = 2
offline_failed_login_attempts = 3
offline_failed_login_delay = 5
```

5. SSSD を再開始します。

```
[root@server ~]# service sssd restart
```

表11.3 SSSD [pam] 設定パラメータ

パラメーター	値の形式	詳細
offline_credentials_expiration	整数	認証プロバイダーがオフラインの場合に、キャッシュ化したログインが許可される時間の長さを日数でセットします。この値は最後の正しいログインから計測されます。指定がない場合は、デフォルト値は ゼロ(**0**) (制限なし) となります。
offline_failed_login_attempts	整数	認証プロバイダーがオフラインの場合に、許可されるログイン試行の失敗回数をセットします。指定されていない場合には、デフォルト値はゼロ (**0**) (制限なし) となります。

パラメーター	値の形式	詳細
offline_failed_login_delay	整数	ユーザーがログイン試行の失敗限度に到達した場合に、ログイン試行を阻止する長さをセットします。ゼロ (**0**) にセットされている場合は、ユーザーが試行の失敗限度に到達するとプロバイダーがオフラインの間は認証ができません。正しいオンラインの認証のみがオフライン認証を再有効化できます。指定されていない場合は、デフォルトは 5 (**5**) になります。

11.2.7. サービスの設定: autofs

11.2.7.1. 自動マウント、LDAP、SSSD について

自動マウントマップは一般的にフラットファイルで、マップとマウントディレクトリー、ファイルサーバーの関係を定義します。(自動マウントは、Storage Administration Guide で説明されています。)

例えば、**nfs.example.com** というファイルサーバーがあったとして、これがディレクトリー **pub** をホストし、自動マウントが **/shares/** ディレクトリーにディレクトリーをマウントするように設定されているとします。すると、マウントする場所は **/shares/pub** になります。すべてのマウントは **auto.master** ファイルに一覧表示されており、これはマウントを設定する様々なマウントディレクトリーとファイルを特定します。さらに **auto.shares** ファイルは、**/shares/** ディレクトリーにつながる各ファイルサーバーとマウントディレクトリーを特定します。この関係を図にすると以下のようになります。

```
          auto.master
    ___________|___________
    |                     |
    |                     |
/shares/            auto.shares
                          |
         |
         |
              nfs.example.com:pub
```

すると、すべてのマウントポイントは、**auto.master** ファイルと **auto.whatever** ファイルという少なくとも 2 つのファイルで定義されることになり、これらのファイルは各ローカルの自動マウントプロセスで利用可能である必要があります。

大規模な環境で管理者がこれを管理する方法の一つは、中央 LDAP ディレクトリーに自動マウント設定を保存し、各ローカルシステムをその LDAP ディレクトリーに向けるように設定することです。これにより、アップデートは単一の場所でのみ必要となり、新しいマップはローカルシステムが自動的に認識することになります。

自動マウント - LDAP 設定では、自動マウントファイルは LDAP エントリーとして保存され、その後に必須の自動マウントファイルに変換されます。それで各要素は、LDAP 属性に変換されます。

LDAP エントリーは以下のようになります。

```
# container entry
dn: cn=automount,dc=example,dc=com
objectClass: nsContainer
objectClass: top
cn: automount

# master map entry
dn: automountMapName=auto.master,cn=automount,dc=example,dc=com
objectClass: automountMap
objectClass: top
automountMapName: auto.master

# shares map entry
dn: automountMapName=auto.shares,cn=automount,dc=example,dc=com
objectClass: automountMap
objectClass: top
automountMapName: auto.shares

# shares mount point
dn:
automountKey=/shares,automountMapName=auto.master,cn=automount,dc=example,
dc=com
objectClass: automount
objectClass: top
automountKey: /shares
automountInformation: auto.shares

# pub mount point
dn:
automountKey=pub,automountMapName=auto.shares,cn=automount,dc=example,dc=c
om
objectClass: automount
objectClass: top
automountKey: pub
automountInformation: filer.example.com:/pub
description: pub
```

スキーマ要素は、以下の図ようにマッチします (RFC 2307 スキーマ) 。

```
               auto.master
                      objectclass: automountMap
                      filename attribute: automountMapName
   ____________________|____________________
   |                                        |
   |                                        |
/shares/                                  auto.shares
objectclass: automount                    objectclass: automountMap
mount point name attribute: automountKey  filename attribute:
automountMapName
map name attribute: automountInformation      |
                                    |
                                              |
                                        nfs.example.com:pub
                                        objectclass: automount
                                        mount point name attribute:
automountKey
```

```
                                                fileserver attribute:
automountInformation
```

autofs は、これらのスキーマ要素を使用して自動マウント設定を取得します。**/etc/sysconfig/autofs** ファイルは、自動マウントエントリーに使用される LDAP サーバー、ディレクトリーの位置、スキーマ要素を特定します。

```
LDAP_URI=ldap://ldap.example.com
SEARCH_BASE="cn=automount,dc=example,dc=com"
MAP_OBJECT_CLASS="automountMap"
ENTRY_OBJECT_CLASS="automount"
MAP_ATTRIBUTE="automountMapName"
ENTRY_ATTRIBUTE="automountKey"
VALUE_ATTRIBUTE="automountInformation"
```

自動マウント設定を LDAP ディレクトリーではなく、SSSD に向けることができます。すると SSSD は自動マウントが必要とする全情報を保存し、ユーザーがディレクトリーをマウントしようとすると、この情報は SSSD にキャッシュされます。これにより、フェイルオーバーやサービスディレクトリー、タイムアウトなどでの設定が有利になるほか、LDAP サーバーへの接続回数が減ることでパフォーマンスが改善されます。最も重要なのは、SSSD を使用することでマウント情報がすべてキャッシュされるので、**LDAP サーバーがオフラインになっても** クライアントはディレクトリーを正常にマウントできます。

11.2.7.2. SSSD での autofs サービスの設定

1. **autofs** パッケージと **libsss_autofs** パッケージがインストールされていることを確認してください。

2. **sssd.conf** ファイルを開きます。

   ```
   [root@server ~]# vim /etc/sssd/sssd.conf
   ```

3. **autofs** サービスを SSSD が管理する一覧に追加します。

   ```
   [sssd]
   services = nss,pam,autofs
   ....
   ```

4. 新たな **[autofs]** サービス設定セクションを作成します。このセクションは空白のままでかまいません。唯一設定可能なオプションは、ネガティブキャッシュヒットのタイムアウトです。

   ただし、このセクションは、SSSD が **autofs** サービスを認識し、デフォルト設定を提供するために必要となります。

   ```
   [autofs]
   ```

5. 自動マウント情報は、SSSD 設定の設定済み LDAP ドメインから読み取られるので、LDAP ドメインが利用可能である必要があります。追加設定がない場合、自動マウント情報には RFC 2307 スキーマと LDAP 検索ベース (**ldap_search_base**) がデフォルト設定となります。これはカスタマイズが可能です。

- ディレクトリーのタイプ、**autofs_provider**; **id_provider** の値がデフォルトです。値が **なし** の場合は、ドメインの autofs を明確に無効とします。
- 検索ベース、**ldap_autofs_search_base**
- マップエントリーの認識に使用するオブジェクトクラス、**ldap_autofs_map_object_class**
- マップ名の認識に使用する属性、**ldap_autofs_map_name**
- マウントポイントエントリーの認識に使用するオブジェクトクラス、**ldap_autofs_entry_object_class**
- マウントポイント名の認識に使用する属性、**ldap_autofs_entry_key**
- マウントポイントの新たな設定情報に使用する属性、**ldap_autofs_entry_value**

例:

```
[domain/LDAP]
...
autofs_provider=ldap
ldap_autofs_search_base="cn=automount,dc=example,dc=com"
ldap_autofs_map_object_class="automountMap"
ldap_autofs_entry_object_class="automount"
ldap_autofs_map_name="automountMapName"
ldap_autofs_entry_key="automountKey"
ldap_autofs_entry_value="automountInformation"
```

6. **sssd.conf** ファイルを保存し、閉じます。
7. **nsswitch.conf** ファイルを編集し、場所を **ldap** から **sss** に変更することで **autofs** を設定して、SSSDの自動マウントマップ情報を探します。

```
[root@server ~]# vim /etc/nsswitch.conf

automount: files sss
```

8. SSSD を再開始します。

```
[root@server ~]# service sssd restart
```

11.2.8. サービスの設定: sudo

11.2.8.1. sudo、LDAP、SSSD について

sudo ルールは **sudoers** ファイルで定義されており、これは一貫性の維持のために各マシンに個別に配布される必要があります。

大規模な環境で管理者がこれを管理する方法の一つは、中央 LDAP ディレクトリーに **sudo** 設定を保存し、各ローカルシステムをその LDAP ディレクトリーに向けるように設定することです。これにより、アップデートは単一の場所でのみ必要となり、新しいルールはローカルシステムが自動的に認識することになります。

sudo-LDAP 設定では、各 **sudo** ルールは、LDAP 属性で定義されている **sudo** ルールとともに LDAP エントリーとして保存されます。

sudoers ルールは以下のようになります。

```
Defaults    env_keep+=SSH_AUTH_SOCK
...
%wheel        ALL=(ALL)       ALL
```

LDAP エントリーは以下のようになります。

```
# sudo defaults
dn: cn=defaults,ou=SUDOers,dc=example,dc=com
objectClass: top
objectClass: sudoRole
cn: defaults
description: Default sudoOptions go here
sudoOption: env_keep+=SSH_AUTH_SOCK

# sudo rule
dn: cn=%wheel,ou=SUDOers,dc=example,dc=com
objectClass: top
objectClass: sudoRole
cn: %wheel
sudoUser: %wheel
sudoHost: ALL
sudoCommand: ALL
```

注記

SSSD は、***sudoHost*** 属性の値により、ローカルシステムに適用される **sudo** ルールのみをキャッシュします。つまり、***sudoHost*** の値は ALL に設定され、ホスト名かシステムネットグループ、システムホスト名か完全修飾ドメイン名または IP アドレスに合致する正規表現を使用します。

sudo サービスは、LDAP サーバーに向けて、それらの LDAP エントリーからルール設定を引き出すように設定が可能です。**sudo** 設定を LDAP ディレクトリーではなく、SSSDに向けることができます。すると SSSD は **sudo** が必要とする全情報を保存し、ユーザーが **sudo** 関連の操作を行おうとすると、最新の **sudo** 設定が LDAP ディレクトリーから SSSD 経由で引き出されます。しかし、SSSD は **sudo** ルールをすべてキャッシュするので、**LDAP サーバーがオフラインになっても、offline** ユーザーは中央化された LDAP 設定を使用してタスクを実行することができます。

11.2.8.2. SSSD を使った sudo の設定

SSSD **sudo** 設定オプションはすべて、**sssd-ldap(5)** man ページに一覧表示されています。

sudo サービスは、以下の手順で設定します。

1. **sssd.conf** ファイルを開きます。

   ```
   [root@server ~]# vim /etc/sssd/sssd.conf
   ```

2. **sudo** サービスを SSSD が管理する一覧に追加します。

```
[sssd]
services = nss,pam,sudo
....
```

3. 新たな **[sudo]** サービス設定セクションを作成します。このセクションは空白のままでかまいません。唯一設定可能なオプションは、sudo not before/after period の評価です。

   ただし、このセクションは、SSSD が **sudo** サービスを認識し、デフォルト設定を提供するために必要となります。

```
[sudo]
```

4. **sudo** 情報は、SSSD 設定の設定済み LDAP ドメインから読み取られるので、LDAP ドメインが利用可能である必要があります。LDAP プロバイダーには、以下のパラメーターが必要です。

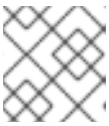
   - ディレクトリーのタイプ、**sudo_provider**; これは常に **ldap** です。
   - 検索ベース、**ldap_sudo_search_base**
   - LDAP サーバーの URI、**ldap_uri**

   例:

```
[domain/LDAP]
id_provider = ldap

sudo_provider = ldap
ldap_uri = ldap://example.com
ldap_sudo_search_base = ou=sudoers,dc=example,dc=com
```

   ID 管理 (IdM または IPA) プロバイダーは、サーバー接続時に Kerberos 認証を行うために追加のパラメーターが必要になります。

```
[domain/IDM]
id_provider = ipa
ipa_domain = example.com
ipa_server = ipa.example.com
ldap_tls_cacert = /etc/ipa/ca.crt

sudo_provider = ldap
ldap_uri = ldap://ipa.example.com
ldap_sudo_search_base = ou=sudoers,dc=example,dc=com
ldap_sasl_mech = GSSAPI
ldap_sasl_authid = host/hostname.example.com
ldap_sasl_realm = EXAMPLE.COM
krb5_server = ipa.example.com
```



注記

ID 管理プロバイダーの **sudo_provider** タイプは **ldap** のままです。

5. **sudo** ルールキャッシュをリフレッシュする間隔を設定します。

   特定のシステムユーザー向け キャッシュは、そのユーザーがタスクを実行すると毎回必ずチェックされ、アップデートされます。しかし、SSSD はローカルシステムに関連するルールをすべてキャッシュします。この完全なキャッシュは、以下の 2 つの方法でアップデートされます。

   - 増分、つまり最後の完全アップデートからルールになされた変更のみ (**ldap_sudo_smart_refresh_interval**、秒数); デフォルト値は 15 分間。
   - 完全、つまりキャッシュ全体をダンプし、LDAP サーバー上の最新ルールすべてをプルします (**ldap_sudo_full_refresh_interval**、秒数); デフォルト値は 6 時間。

   これら 2 つの間隔は、以下のように別個に設定されます。

   ```
   [domain/LDAP]
   ...
   ldap_sudo_full_refresh_interval=86400
   ldap_sudo_smart_refresh_interval=3600
   ```

   注記

   SSSD は、ローカルシステムに適用される **sudo** ルールのみをキャッシュします。つまり、***sudoHost*** の値は ALL に設定され、ホスト名かシステムネットグループ、システムホスト名か完全修飾ドメイン名または IP アドレスに合致する正規表現を使用します。

6. オプションでは、任意の値を設定して、**sudo** ルールに使われるスキーマを変更します。

   スキーマ要素は、**ldap_sudorule_*** 属性で設定されています。デフォルトでは、すべてのスキーマ要素が sudoers.ldap で定義されたスキーマを使用します。これらのデフォルトは、ほとんどすべての導入で使用されます。

7. **sssd.conf** ファイルを保存し、閉じます。

8. **nsswitch.conf** ファイルを編集し、**sss** の場所を加えることで **sudo** が SSSD のルール設定を探すように設定します。

   ```
   [root@server ~]# vim /etc/nsswitch.conf

   sudoers: files sss
   ```

9. SSSD を再開始します。

   ```
   [root@server ~]# service sssd restart
   ```

11.2.9. サービスの設定: OpenSSH およびキャッシュされたキー

OpenSSH は、2 つのシステム間に安全で暗号化された接続を作成します。一方のマシンが他方のマシンを認証してアクセスを許可します。認証は、マシン自体のサーバー接続に対するものである場合と、該当マシン上のユーザーに対する場合があります。OpenSSH の詳細は、12章*OpenSSH* で説明されています。

この認証は、認証を行なっているユーザーもしくはマシンを特定する **公開キーと秘密キーのペア** を通して実行されます。マシンにアクセスしようとしているリモートのマシンもしくはユーザーは、キーのペアを提示します。するとローカルマシンは、そのリモートのエンティティを信頼するかどうかを決定します。信頼できる場合は、そのリモートマシン用の公開キーは **known_hosts** ファイルに、リモートユーザー用の公開キーは **authorized_keys** ファイルに保存されます。このリモートマシンもしくはユーザーが再度認証を試みる際は常に、ローカルシステムは単にこの **known_hosts** か **authorized_keys** ファイルを最初にチェックして、このリモートのエンティティが認証され、信頼できるものかを確かめた後にアクセスを許可します。

最初の問題は、これらの ID 認証を信頼性を持って行うことです。

known_hosts ファイルは、マシン名、IP アドレス、公開キーの 3 つで構成されています。

```
server.example.com,255.255.255.255 ssh-rsa
AbcdEfg1234ZYX098776/AbcdEfg1234ZYX098776/AbcdEfg1234ZYX098776=
```

known_hosts ファイルは、様々な理由ですぐに古いものとなってしまいます。IP アドレス経由の DHCP サイクルを使用しているシステムでは、新たなキーが定期的に再発行されたり、仮想マシンやサービスがオンラインで持ち込まれたり削除されたりする場合があります。このようなことがあると、ホスト名、IP アドレス、キーの構成が変わってしまいます。

管理者は、現行の **known_hosts** ファイルをクリーンに維持して、セキュリティーを保つ必要があります。 (さもなくば、システムユーザーは、単に提示されたマシンやキーをどれでもうけいれるという習慣に陥ってしまい、キーベースのセキュリティーの利点がなくなってしまいます。)

さらには、マシンとユーザーにはスケーラブルな方法でキーを配布するという問題があります。マシンは暗号化セッション確立の一部としてキーを送信できますが、ユーザーは事前にキーを提供する必要があります。単にキーを伝達し、一貫してアップデートするのは、困難な管理タスクです。

最後に、SSH キーおよびマシン情報はローカルでのみ維持されます。**known_hosts**ファイルが一様にアップデートされていないと、ネットワーク上のマシンやユーザーが一部のシステムには認識、信頼され、別のシステムには認識、信頼されないということが起こり得ます。

SSSD のゴールは、認証情報キャッシュとして機能することです。これには、マシンおよびユーザーの SSH 公開キーの認証情報キャッシュとして機能することが含まれます。OpenSSH は、キャッシュされたキーの確認に SSSD を参照するように設定されています。SSSD は、 Red Hat Linux の Identity Management (IPA) ドメインを ID として使用し、Identity Management は実際に公開キーおよびホスト情報を保存します。

注記

SSSD を OpenSSH のキーキャッシュとして使用できるのは、Identity Management ドメインに登録もしくは参加している Linux マシンのみです。他の Unix マシンや Windows マシンは、**known_hosts** ファイルで通常の認証メカニズムを使用する必要があります。

11.2.9.1. ホストキーに SSSD を使用するための OpenSSH 設定

OpenSSH は、ユーザー特定の設定ファイル (**~/.ssh/config**) か、システムワイドの設定ファイル (**/etc/ssh/ssh_config**) で設定されます。ユーザーファイルはシステム設定に優先し、最初に獲得されたパラメーター値が使用されます。このファイルの形式および習慣については、12章*OpenSSH* で説明されています。

SSSD にはホストキーを管理するためのツール、**sss_ssh_knownhostsproxy**、があり、以下の 3 つの操作を実行します。

1. 登録済み Linux システムから公開ホストキーを取得します。
2. カスタムホストファイル **.ssh/sss_known_hosts** にホストキーを保存します。
3. ソケット (デフォルト) もしくは安全な接続でホストマシンとの接続を確立します。

このツールは以下の形式になります。

```
sss_ssh_knownhostsproxy [-d sssd_domain] [-p ssh_port] HOST
[PROXY_COMMAND]
```

表11.4 sss_ssh_knownhostsproxy オプション

短い引数	長い引数	詳細
	HOSTNAME	チェックおよび接続するホストのホスト名を与えます。OpenSSH 設定ファイルでは、**%h** のトークンである場合があります。
	PROXY_COMMAND	SSH クライアント接続に使用するプロキシコマンドを渡します。これは、**ssh -o ProxyCommand=値** の実行に類似したものです。このオプションは、コマンドラインもしくは別のスクリプトで **sss_ssh_knownhostsproxy** を実行する際に使われますが、OpenSSH 設定ファイルでは不要です。
-d **sssd_domain**	--domain **sssd_domain**	指定ドメインのエンティティ内の公開キーのみを検索します。これがない場合は、SSSD はすべての設定済みドメインでキーを検索します。
-p **port**	--port **port**	SSH クライアントへの接続にこのポートを使用します。デフォルトでは、port 22 です。

この SSSD ツールを使用するには、**ssh_config** もしくは **~/.ssh/config** ファイルに 2 つのパラメーターを追加または編集します。

- SSH クライアントに接続するようにコマンドに指定します (**ProxyCommand**)。これは、**sss_ssh_knownhostsproxy** に希望する引数とホスト名を加えたものです。
- デフォルトの **known_hosts** ファイルではなく、SSSD ホストファイルの場所を指定します。(**UserKnownHostsFile**)。SSSD ホストファイルは、**.ssh/sss_known_hosts** です。

例えば、以下では **IPA1** SSSD ドメイン内の公開キーを探して、提供されるポートとホストで接続します。

```
ProxyCommand /usr/bin/sss_ssh_knownhostsproxy -p %p -d IPA1 %h
UserKnownHostsFile2 .ssh/sss_known_hosts
```

11.2.9.2. ユーザーキーに SSSD を使用するための OpenSSH 設定

OpenSSH では、ユーザーキーはローカルシステムの **authorized_keys** ファイルに保存されています。ホストに関してはOpenSSH が参照するために、SSSD がユーザー公開キーの別個のキャッシュを維持し、自動的にアップデートします。これは、**.ssh/sss_authorized_keys** ファイルに保存されます。

OpenSSH は、ユーザー特定の設定ファイル (**~/.ssh/config**) か、システムワイドの設定ファイル (**/etc/ssh/ssh_config**) で設定されます。ユーザーファイルはシステム設定に優先し、最初に獲得されたパラメーター値が使用されます。このファイルの形式および習慣については、12章*OpenSSH* で説明されています。

SSSD にはユーザーキーを管理するためのツール、**sss_ssh_authorizedkeys**、があり、以下の 2 つの操作を実行します。

1. Identity Management (IPA) ドメインのユーザーエンティティからユーザーの公開キーを取得します。
2. 標準的な認証キー形式でユーザーキーをカスタムファイル **.ssh/sss_authorized_keys** に保存します。

このツールは以下の形式になります。

```
sss_ssh_authorizedkeys [-d sssd_domain] USER
```

表11.5 sss_ssh_authorizedkeys オプション

短い引数	長い引数	詳細
	USER	公開キーを取得するユーザー名またはアカウント名を与えます。OpenSSH 設定ファイルでは、**%u** のトークンで示すこともできます。
-d **sssd_domain**	--domain **sssd_domain**	指定ドメインのエンティティ内の公開キーのみを検索します。これがない場合は、SSSD はすべての設定済みドメインでキーを検索します。

ユーザーキーに SSSD を使用するための OpenSSH の設定方法は、SSH 導入により 2 つのオプションがあります。

- 最も一般的なのは、SSH が認証キーコマンドをサポートする場合です。このケースで必要なのは、ユーザーキーの取得で実行するコマンドの指定のみです。以下に例を挙げます。

  ```
  AuthorizedKeysCommand /usr/bin/sss_ssh_authorizedkeys
  ```

- SSH も公開キーエージェントのサポートが可能です。この場合、ユーザー名などの必要な引数のトークンを含むエージェントキーを取得するために使用するコマンドを与えます。

```
PubKeyAgent /usr/bin/sss_ssh_authorizedkeys %u
```

11.2.10. SSSD および ID プロバイダー (ドメイン)

SSSD は異なる ID プロバイダーと関連する設定エントリーである **ドメイン** を認識します。ドメインは ID プロバイダーと認証メソッドの組み合わせです。SSSD は LDAP ID プロバイダー (OpenLDAP, Red Hat Directory Server, 及び Microsoft Active Directory を含む) と共に機能して、ネイティブな LDAP 認証、または Kerberos 認証を使用できます。

ドメインの設定は、 **ID プロバイダー**、**認証プロバイダー**、特定の設定を定義してそれらのプロバイダー内の情報にアクセスします。ID および認証プロバイダーにはいくつかのタイプがあります。

- LDAP、一般的な LDAP サーバー用
- Active Directory (LDAP プロバイダータイプの拡張)
- Identity Management (LDAP プロバイダータイプの拡張)
- Local、ローカルの SSSD データベース用
- Proxy
- Kerberos (認証プロバイダーのみ)

ID および認証プロバイダーは、ドメインエントリー内で異なる組み合わせで設定が可能です。可能な組み合わせは 表11.6「識別ストアと認証タイプの組み合わせ」 に一覧表示してあります。

表11.6 識別ストアと認証タイプの組み合わせ

識別プロバイダー	認証プロバイダー
Identity Management (LDAP)	Identity Management (LDAP)
Active Directory (LDAP)	Active Directory (LDAP)
Active Directory (LDAP)	Kerberos
LDAP	LDAP
LDAP	Kerberos
proxy	LDAP
proxy	Kerberos
proxy	proxy

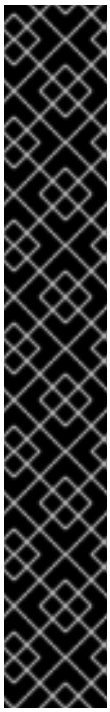
ドメインエントリ自身と共に、SSSD がクエリをするドメインの一覧にドメイン名が追加されなければなりません。例えば:

```
[sssd]
domains = LOCAL,Name
...

[domain/Name]
id_provider = type
auth_provider = type
provider_specific = value
global = value
```

global 属性は、キャッシュやタイムアウトセッティングなどのいかなるタイプのドメインにも利用可能です。各識別と認証のプロバイダーはそれ自身の必須及びオプションの設定パラメータのセットを所有しています。

表11.7 一般的な [domain] 設定パラメータ

パラメーター	値の形式	詳細
id_provider	文字列	このドメイン用に使用するデータバックエンドを指定します。サポートされている識別バックエンドには以下があります: • ldap • ipa (Red Hat Enterprise Linux 内のIdentity Management) • ad (Microsoft Active Directory) • **nss_nis** など、レガシー NSS プロバイダー用の proxy。proxy を使用すると ID プロバイダーは、正しく開始するためにロードするレガシーNSS ライブラリも必要になります。**proxy_lib_name** オプションでセットされます。 • local, SSSD 内部ローカルプロバイダー
auth_provider	文字列	ドメイン用に使用される認証プロバイダーをセットします。このオプションのデフォルト値は **id_provider** の値となります。サポートされている認証プロバイダーは ldap、ipa、ad、krb5 (Kerberos)、proxy、および none です。

パラメーター	値の形式	詳細
min_id,max_id	整数	**オプションです。**ドメイン用に UID と GID の範囲を指定します。ドメインがこの範囲外のエントリーを含んでいる場合、それらは無視されます。**`min_id`** のデフォルト値は **`1`** であり、**`max_id`** のデフォルト値は **`0`** でこれは無制限となります。  **重要** デフォルトの **`min_id`** 値はすべての識別プロバイダーに対して同じです。LDAP ディレクトリが 1 からスタートする UID 番号を使用している場合は、ローカルの **`/etc/passwd`** ファイル内のユーザーと競合原因となる可能性があります。この競合を回避するには、**`min_id`** を **`1000`** かまたはできるだけ高い値にセットします。
enumerate	ブール値	**オプションです。**ドメインのユーザーとグループを一覧表示するかどうかを指定します。Enumeration (列挙) とはリモートソース上のユーザーとグループの全セットがローカルマシン上にキャッシュされていると言う意味です。Enumeration が無効になっている時は、ユーザーとグループは要求がある時にのみキャッシュされます。 **警告** Enumeration が有効になっている時は、クライアントを

パラメーター	値の形式	詳細
		再初期化するとリモートソースからユーザーとグループの使用可能な全セットが完全にリフレッシュされます。同じように、SSSDが新しいサーバーに接続されている時は、リモートソースからユーザーとグループの使用可能な全セットが取り込まれてローカルマシンにキャッシュされます。リモートソースに接続されている多数のクライアントを持つドメイン内では、このリフレッシュのプロセスは、クライアントからの頻繁なクエリによりネットワークパフォーマンスを害することになります。ユーザーとグループの使用可能なセットが充分に大きい場合は、クライアントパフォーマンスを悪化することもあります。 このパラメータのデフォルト値は **`false`** であり、これは enumeration を無効にします。

パラメーター	値の形式	詳細
cache_credentials	ブール値	**オプションです。**ユーザーの認証情報をローカルの SSSD ドメインデータベースキャッシュに保管するかどうかを指定します。このパラメータのデフォルト値は **`false`** です。LOCAL ドメイン以外のドメイン用にこの値を **`true`** にセットすると、オフライン認証が有効になります。
entry_cache_timeout	整数	**オプションです。**SSSD が**ポジティブ**なキャッシュヒットをキャッシュすべき時間の長さを秒で指定します。ポジティブキャッシュヒットとは、成功したクエリのことです。
use_fully_qualified_names	ブール値	**オプションです。**このドメインへの要求が完全修飾型ドメイン名を必要とするかどうかを指定します。**`true`** にセットしてあると、このドメインへのすべての要求は完全修飾型ドメイン名を使用しなければなりません。また、要求からの出力は完全修飾名を表示すると言う意味です。要求を完全修飾のユーザー名のみに限定するとSSSD は、競合するユーザー名を持つユーザーのドメイン間で区別ができるようになります。***`use_fully_qualified_names`*** が **`false`** にセットされている場合、要求で引き続き完全修飾名を使用できますが、出力では簡易バージョンが表示されます。 SSSD は、レルム名ではなくドメイン名を基にした名前だけを構文解析できます。しかし、ドメインとレルムの両方で同一名が使用できます。

11.2.11. ドメインの作成: LDAP

LDAP ドメインとは、単に SSSD が LDAP ディレクトリを識別プロバイダーとして使用すると言う意味です(そして、オプションとして認証プロバイダーとしても)。SSSD は数種の主要なディレクトリサービスをサポートします:

- Red Hat ディレクトリサーバー
- OpenLDAP
- Identity Management (IdM または IPA)
- Microsoft Active Directory 2008 R2

注記

一般的な LDAP ID プロバイダーで利用可能なパラメーターはすべて、このサブセットである Identity Management および Active Directory ID プロバイダーでも利用可能です。

11.2.11.1. LDAP ドメイン設定用のパラメータ

LDAP ディレクトリーは、ID プロバイダーとしても認証プロバイダーとしても機能することができます。この設定には LDAP サーバー内のユーザーディレクトリーを特定し、これに接続するための十分な情報が必要ですが、これらの接続パラメーターの定義方法は柔軟なものです。

LDAP サーバーへの接続に使用するためのユーザーアカウントの指定や、パスワード操作に他の LDAP サーバーの使用などよりきめ細かい制御を提供するために他のオプションも利用可能です。最も一般的なオプションは 表11.8「LDAP ドメイン設定のパラメータ」 に一覧表示してあります。

注記

他にも多くのオプションが LDAP ドメイン設定用の man ページ内に一覧表示されています。**`sssd-ldap(5)`**。

表11.8 LDAP ドメイン設定のパラメータ

パラメーター	詳細
ldap_uri	SSSD の接続先である LDAP サーバーの URI をカンマ区切りの一覧で提供します。この一覧は設定により順序を指定できるため、一覧内の最初のサーバーが最初に試行されるようにできます。サーバーを追加して一覧表示しておくとフェイルオーバー保護を提供できます。これがない場合は、DNS SRV レコードから検出可能です。
ldap_search_base	LDAP ユーザー操作の実行のために使用するベース DN を提供します。
ldap_tls_reqcert	TLS セッションで SSL サーバー証明書のチェックの仕方を指定します。4 つのオプションがあります: • **never** は証明書への要求を無効にします。 • **allow** は証明書を要求しますが、証明書が無くても、または不正な証明書が与えられても通常に進行します。 • **try** は証明書を要求して証明書が与えられなくても通常に進行します。不正な証明書が与えられた場合は、セッションは終了します。 • **demand** と **hard** は同じオプションです。有効な証明書を要求して、それがないとセッションは終了します。 デフォルトは **hard** です。

パラメーター	詳細
ldap_tls_cacert	SSSD が認識するすべての CA 用の CA 証明書を含んでいるファイルに完全なパスとファイル名を渡します。SSSD はこれらの CA によって発行された証明書はどれも受理します。 これが明示的に与えられていない場合は、OpenLDAP システムのデフォルトを使用します。
ldap_referrals	SSSD が　LDAP 参照を使用するかどうかをセットします。このことは 1 つの LDAP データベースから別の LDAP データベースにクエリを転送することを意味します。SSSD はデータベースレベルの参照とサブツリーの参照をサポートします。同一の LDAP サーバー内の参照には、SSSD はクエリされているエントリの DN を調節します。異なる LDAP サーバーに行く参照には、SSSD は DN 上で完全なマッチを行います。この値を **`true`** に設定すると、参照を有効にします。これが、デフォルトです。 参照は、追跡を試みる時間のために、全体的なパフォーマンスにマイナスの影響を与える場合があります。参照のチェックを無効にすると、パフォーマンスは大幅に改善します。

パラメーター	詳細

ldap_schema	ユーザーエントリーの検索の際に使用するスキーマのバージョンを設定します。可能なのは、**rfc2307**、**rfc2307bis**、**ad**、**ipa** です。デフォルトは、**rfc2307** です。
	RFC 2307 では、グループオブジェクトは複数値の属性である ***memberuid*** を使用しますが、これはそのグループに所属するユーザーの名前を一覧表示します。RFC 2307bis では、グループオブジェクトは ***member*** 属性を使用しますが、これはユーザーまたはグループエントリの完全識別名 (DN) を含んでいます。RFC 2307bis は、***member*** 属性を使用している入れ子グループ (nested groups) を可能にします。これらの異なるスキーマがグループメンバーシップについて異なる定義を使用するため、SSSD において間違えた LDAP スキーマの使用は、適切な権限がある場合でもネットワークリソースの表示と管理の両方に影響を及ぼします。
	
	例えば、RFC 2307bis の使用では、すべてのグループは入れ子グループ、あるいはプライマリ/セカンダリグループの使用時に返されます。
	
	`$ id`
	`uid=500(myserver)`
	`gid=500(myserver)`
	`groups=500(myserver),510(myothergroup)`
	
	SSSD が RFC 2307 スキーマを使用している場合は、プライマリグループのみが返されます。
	
	この設定は SSSD がグループメンバーを決定する方法にのみ影響します。実際のユーザーデータは変更しません。

ldap_search_timeout	LDAP 検索がキャンセルになるまでに実行してキャッシュ化した結果が返される時間の長さを秒でセットします。**enumerate** 値が「false 」の時はデフォルトでは 5 であり、**enumerate** が「true」の時はデフォルトは 30 になります。 LDAP 検索がタイムアウトになると、SSSD は自動的にオフラインモードに切り替わります。
ldap_network_timeout	接続試行が失敗した後に SSSD が LDAP サーバーへ投票を試みる時間の長さを秒でセットします。デフォルトは 6 秒です。
ldap_opt_timeout	サーバーから反応が受信されない場合に同期 LDAP 操作が中止されるまでの時間の長さを秒でセットします。このオプションは、SASL バインドの場合に KDC との通信時のタイムアウトも制御します。デフォルトは 5 秒です。

11.2.11.2. LDAP ドメインの例

LDAP 設定は、ユーザー特定の環境と SSSD の動作に応じてかなり柔軟になります。以下は LDAP ドメインの一般的な例の一部ですが、SSSD 設定はこれらの例に限られるものではありません。

注記

ドメインエントリの作成と並行して、SSSD 用のドメインの一覧へ新規ドメインを追加して**sssd.conf** ファイル内でクエリをします。例えば:

```
domains = LOCAL,LDAP1,AD,PROXYNIS
```

例11.2 基本的 LDAP ドメインの設定

LDAP ドメインは３つの項目を要求します:

- LDAP サーバー
- 検索ベース
- 安全な接続を確立する手段

最後の項目は LDAP 環境に左右されます。SSSD は機密情報を取り扱うため安全な接続を必要とします。この接続は専属の TLS/SSL 接続であるか、あるいは Start TLS を使用することができます。

専属の TLS/SSL 接続の使用は単に LDAPS 接続を使用してサーバーに接続します。そのため、**ldap_uri** オプションの一部として設定されています:

```
# An LDAP domain
[domain/LDAP]
enumerate = false
cache_credentials = true

id_provider = ldap
auth_provider = ldap

ldap_uri = ldaps://ldap.example.com:636
ldap_search_base = dc=example,dc=com
```

Start TLS の使用には、安全でないポート上で動的に安全な接続を確立するために証明書情報を入力する手段が必要になります。これは、**ldap_id_use_start_tls** オプションで Start TLS を使用することと、**ldap_tls_cacert** で SSL サーバー証明書を発行した CA 証明書を識別することで達成できます。

```
# An LDAP domain
[domain/LDAP]
enumerate = false
cache_credentials = true

id_provider = ldap
auth_provider = ldap

ldap_uri = ldap://ldap.example.com
ldap_search_base = dc=example,dc=com
```

```
ldap_id_use_start_tls = true
ldap_tls_reqcert = demand
ldap_tls_cacert = /etc/pki/tls/certs/ca-bundle.crt
```

11.2.12. ドメインの作成: Identity Management (IdM)

Identity Management (IdM または IPA) ID プロバイダーは、一般的な LDAP プロバイダーの拡張機能です。LDAP プロバイダーで利用可能な設定オプションはすべて、IdM プロバイダーでも利用可能で、SSSD を IdM ドメインのクライアントとして機能させ、 IdM の機能を拡張することのできるいくつかのパラメーターも利用可能です。

Identity Management は、ID や認証、アクセス制御ルール、パスワードなど、ドメインの ***_provider** パラメーターすべてのプロバイダーとして機能することができます。さらに Identity Management のドメイン内には、SELinux ポリシーや自動マウント情報、ホストベースのアクセス制御を管理する設定オプションがあります。IdM ドメイン内のこれら全機能は SSSD 関連付けが可能で、これらのセキュリティー関連のポリシーがシステムユーザー用に適用され、キャッシュされることを可能にしています。

例11.3 基本的な IdM プロバイダー

IdM プロバイダーは LDAP プロバイダーのように、ID や認証、アクセス制御などの異なるサービスに対応するよう設定することが可能です。

IdM サーバーには、非常に便利な (ただし必須ではない) 2 つの追加設定があります。

- デフォルトの RFC 2307 スキーマではなく、特定の IdM スキーマの利用。
- クライアントが最初に IdM ドメインに接続する際に、このクライアントの IP アドレスで Identity Management ドメインの DNS サーバーをアップデートするように SSSD を設定。

```
[sssd]
domains = local,example.com
...

[domain/example.com]
id_provider = ipa
ipa_server = ipaserver.example.com
ipa_hostname = ipa1.example.com
auth_provider = ipa
access_provider = ipa
chpass_provider = ipa

# set which schema to use
ldap_schema = ipa

# automatically update IdM DNS records
ipa_dyndns_update = true
```

Identity Management は、Linux ドメインにわたってユーザーのセキュリティーポリシーと ID を定義、維持します。これには、アクセス制御ポリシー、SELinux ポリシー、その他のルールが含まれます。IdM ドメインのこれらの要素のいくつかは、SSSD を IdM クライアントとして使用して SSSD と直接対話します。これらの機能は、**sssd.conf** の IdM ドメインエントリーで管理できます。

ほとんどの設定パラメーターは、スキーマ要素の設定 (IdM は固定スキーマを使用するのでほとんどの導入では関係なし) に関連し、変更の必要はありません。実際のところ、IdM の機能でクライアント側の設定を必要とするものは一つもありません。しかし、動作を微調整すると役立つ場合もあります。

例11.4 SELinux での IdM プロバイダー

IdM は、システムユーザーの SELinux ユーザーポリシーを定義できるので、SSSD 向けの SELinux プロバイダーとして機能することができます。これは、**`selinux_provider`** パラメーターで設定されます。プロバイダーは **`id_provider`** 値をデフォルトとしているので、明示的に SELinux ルールを **サポートする** と設定する必要はありません。しかし、SSSD 内 のIdM プロバイダーに対する SELinux サポートを明示的に **無効にする** と便利な場合があります。

```
selinux_provider = ipa
```

例11.5 ホストベースのアクセス制御での IdM プロバイダー

IdM は、ユーザーが接続に使用しているホストやユーザーが接続しようとしているホストがどれかに基づいてサービスやシステム全体へのアクセスを制限することで、ホストベースのアクセス制御を定義できます。このルールは、アクセスプロバイダーの動作の一部として、SSSD による評価と強制が可能です。

ホストベースのアクセス制御に効果を持たせるには、少なくとも Identity Management サーバーがアクセスプロバイダーである必要があります。

SSSD がホストベースのアクセス制御ルールを評価する方法を設定するには、2 つのオプションがあります。

- SSSD は、ユーザーが IdM リソースへの接続に使用しているマシン (ソースホスト) を評価できます。これはデフォルトでは無効なので、ルールの対象ホスト部分のみが評価されます。
- SSSD は、指定された間隔でキャッシュにあるホストベースのアクセス制御ルールを更新できます。

例:

```
access_provider = ipa
ipa_hbac_refresh = 120

# check for source machine rules; disabled by default
ipa_hbac_support_srchost = true
```

例11.6 Cross-Realm Kerberos Trusts での Identity Management

Identity Management (IdM または IPA) は、Active Directory DNS ドメインと Kerberos レルムの信頼できる関係で設定が可能です。これにより、Active Directory ユーザーは、Linux システム上のサービスとホストにアクセスできるようになります。

SSSD には、レルム間の信頼で使われる設定が 2 つあります。

- Kerberos チケットに必要なデータを追加するサービス
- サブドメインをサポートする設定

Kerberos チケットデータ

Microsoft は、**privileged access certificates** または MS-PAC と呼ばれる特別な認証構造を使用しています。PAC は、Windows ドメインの他の Windows クライアントやサーバーへのエンティティを特定する方法として、Kerberos チケットに埋め込まれています。

SSSDには、Kerberos チケットに新たなデータを生成する特別な PAC サービスがあります。Active Directory ドメインを使用している際には、Windows ユーザー用に PAC データを含める必要がある場合があります。その場合には、SSSD で **pac** サービスを有効にします。

```
[sssd]
services = nss, pam, pac
...
```

Windows サブドメイン

通常は、SSSD のドメインエントリーは単一 ID プロバイダーに直接対応します。しかし、IdM のレルム間の信頼があると、IdM ドメインは別のドメインを信頼でき、ドメインはお互いに透明性を持つことに成ります。SSSD はこの信頼できる関係に従うことが可能になり、IdM ドメインが設定されればいかなる Windows ドメインも SSSDのドメインセクションで設定することなく、SSSD が検索し、サポートすることになります。

これは、**subdomains_provider** パラメーターを IdM ドメインセクションに加えることで設定されます。これは実際には、オプションのパラメーターです。サブドメインが発見されれば、SSSD はデフォルトで **ipa** プロバイダータイプを使用します。しかしこのパラメーターは、値を **none** に設定することで、サブドメインフェッチを無効にする際にも使用できます。

```
[domain/IDM]
...
subdomains_provider = ipa
get_domains_timeout = 300
```

11.2.13. ドメインの作成: Active Directory

Active Directory ID プロバイダーは、一般的な LDAP プロバイダーの拡張機能です。LDAP プロバイダーで利用可能な設定オプションはすべて、Active Directory プロバイダーでも利用可能で、ユーザーアカウントや Active Directory とシステムユーザー間の ID マッピングに関連するいくつかのパラメーターも利用可能です。

11.2.13.1. SSSD と Active Directory

標準的な LDAP サーバーと Active Directory サーバーには根本的な違いがいくつかあります。そのため、Active Directory プロバイダーの設定の際には、特別な設定を必要とする設定領域がいくつかあります。

- Winows セキュリティー ID を使用している ID は、対応する Linux システムユーザー ID にマッピングする必要があります。
- 検索は、範囲取得拡張の説明をする必要があります。
- LDAP 紹介にパフォーマンスの問題がある可能性がある。

11.2.13.1.1. Active Directory セキュリティー ID と Linux ユーザー ID のマッピング

Windows と Linux がシステムユーザーに対応する方法と、Active Directory および標準 LDAPv3 ディレクトリーサービスで使用されるユーザースキーマには、内包的な構造の違いがあります。SSSD でシステムユーザーの管理に Active Directory ID プロバイダーを使用する際は、Active Directory スタイルのユーザーを新たな SSSD ユーザーに調和させる必要があります。これには以下の 2 つの方法があります。

- Services for Unix を使用して Windows ユーザーおよびグループエントリー上に POSIX 属性を挿入し、これらの属性を PAM/NSS にプルする。
- SSSD 上の ID マッピングを使用して、Active Directory 説明 ID (SID) と Linux 上の生成済み UID との間のマップを作成する。

ID マッピングは、新たなパッケージや Active Directory 上での設定を必要としないため、ほとんどの環境で最も簡単なオプションです。

11.2.13.1.1.1. ID マッピングのメカニズム

Linux/Unix システムはローカルのユーザー ID 番号都グループ ID 番号を使ってシステム上のユーザーを識別します。これらの UID:GID 番号は、**501:501** といった単純な整数です。これらの番号は、大規模な Linux/Unix ドメインの一部のシステムでも、常にローカルで作成、管理されるので、単純なものになります。

Microsoft Windows と Active Directory は異なるユーザー ID 構造を使ってユーザーやグループ、マシンを識別します。各 ID は、セキュリティーバージョンや発行権限のタイプ、マシン、ID 自体を識別する異なるセグメントで構築されています。例えば、以下のようになります。

```
S-1-5-21-3623811015-3361044348-30300820-1013
```

3 つ目から 6 つ目のブロックは、マシン識別子です。

```
S-1-5-21-3623811015-3361044348-30300820-1013
```

最後のブロックは、特別エンティティを識別する **相対識別子** (RID) です。

```
S-1-5-21-3623811015-3361044348-30300820-1013
```

可能な ID 番号の範囲は、常に SSSD に割り当てられます。 (これはローカルの範囲なので、すべてのマシンで同じものになります。)

```
|_____________________________________|
|                                     |
minimum ID                          max ID
```

この範囲は (デフォルトで) 1 万セクションに分けられ、各セクションには 20 万 ID が割り当てられます。

```
| slice 1 | slice 2 |   ...   |
|_________|_________|_________|
|         |         |         |
minimum ID                          max ID
```

新たな Active Directory ドメインが検出されると、SID がハッシュ化されます。すると SSSD はこのハッシュのモジュールと利用可能なセクションの数で、Active Directory ドメインにどの ID セクションを割り当てるかを決めます。これは、ID セクション割り当て方法としては信頼性と一貫性のあるもの

で、ほとんどのクライアントマシン上で同一の Active Directory ドメインに同一の ID 範囲が割り当てられます。

```
| Active  | Active  |         |
|Directory|Directory|         |
|domain 1 |domain 2 |   ...   |
|         |         |         |
| slice 1 | slice 2 |   ...   |
|_________|_________|_________|
|         |         |         |
minimum ID                   max ID
```

注記

ID セクションの割り当て方法には一貫性がありますが、**ID マッピングは Active Directory ドメインがクライアントマシン上で発生する順序に基づいています**。このため、すべての Linux クライアントマシン上で ID 範囲の割り当てに一貫性が保たれない場合もあります。一貫性が必要な場合は、ID マッピングを無効にし、明示的な POSIX 属性の使用を検討してください。

11.2.13.1.1.2. ID マッピングパラメーター

ID マッピングは 2 つのパラメーターで有効化されます。 1 つ目でマッピングを有効にし、2 つ目で適切な Active Directory ユーザースキーマをロードします。

```
ldap_id_mapping = True
ldap_schema = ad
```

注記

ID マッピングが有効になると、***uidNumber*** および ***gidNumber*** 属性は無視されます。これにより、手動で割り当てられた値はいかなるものでも止められます。**何らかの** 値が手動で割り当てられる必要がある場合は、**すべての** 値が手動で割り当てられ、ID マッピングを無効にする必要があります。

11.2.13.1.1.3. マッピングユーザー

Active Directory ユーザーが初めてローカルシステムサービスにログイン使用とする際には、そのユーザーのエントリーが SSSD キャッシュに作成されます。リモートユーザーはシステムユーザーのように設定されます。

- ユーザーの SID およびそのドメインの ID 範囲に基づいてシステム UID が作成されます。
- ユーザーに GID が作成されます。これは UID と同一のものです。
- ユーザーにプライベートグループが作成されます。
- **sssd.conf** ファイルのホームディレクトリー形式に基づいて、ホームディレクトリーが作成されます。
- システムデフォルトまたは **sssd.conf** ファイル内の設定に基づいて、シェルが作成されます。

- ユーザーが Active Director 内のいずれかのグループに所属する場合、SID を使って SSSD が Linux 上のこれらのグループにユーザーを追加します。

11.2.13.1.1.4. Active Directory Users and Range Retrieval Searches

Microsoft Active Directory には、***MaxValRange*** の属性があり、複数の値の属性に戻す値の制限数を設定するものです。これは、**範囲取得** 検索の拡張機能です。基本的にこれは複数のミニ検索を実行し、すべての一致が返されるまでそれぞれの検索は特定の範囲内での結果のサブセットを返します。

例えば、***member*** 属性を検索する際に、各エントリーに複数の値があり、その属性に複数のエントリーがあり場合があります。2000 件 (またはそれ以上) の一致結果があった場合、***MaxValRange*** は一度に表示される数を制限します。これが、値の範囲です。そして、特定の属性には新たなフラグのセットが付けられて、その結果がセット内のどの範囲にあるかを示します。

```
attribute:range=low-high:value
```

例えば、検索結果を 100 件から 500 件にするには、

```
member;range=99-499: cn=John Smith...
```

これは Microsoft ドキュメンテーションの http://msdn.microsoft.com/en-us/library/cc223242.aspx で説明されています。

SSSD は、Active Directory プロバイダーを使った範囲取得を新たな設定なしにユーザーおよびグループ管理の一部としてサポートします。

しかし、**ldap_user_search_base** のような検索設定に利用可能な LDAP プロバイダー属性の中には、範囲取得との効率がよくないものもあります。Active Directory プロバイダードメインで検索ベースを設定する際には注意して、検索がどの範囲取得を引き起こすかを考慮してください。

11.2.13.1.1.5. パフォーマンスと LDAP の参照

参照は、追跡を試みる時間のために、全体的なパフォーマンスにマイナスの影響を与える場合があります。参照追跡を Active Directory プロバイダー内で使用すると、特にパフォーマンスが悪化します。参照のチェックを無効にすると、パフォーマンスは大幅に改善できます。

LDAP 参照はデフォルトで有効となっているので、以下のようにLDAP ドメイン設定で明示的に無効にする必要があります。

```
ldap_referrals = false
```

11.2.13.1.2. Active Directory ID プロバイダーの設定

重要

設定ファイルの変更後には、必ず SSSD を再起動します。

```
[root@rhel-server ~]# service sssd restart
```

Active Directory は、ID や認証、アクセス制御規則、ドメインの 全***_provider*** パラメーターのプロバイダーとして機能できます。さらには、デフォルトの RFC 2307 を使うのではなく、ユーザーおよびグループエントリー用のネイティブの Active Directory スキーマをロードすることもできます。

例11.7 Active Directory 2008 R2 ドメイン

```
[root@rhel-server ~]# vim /etc/sssd/sssd.conf

[sssd]
config_file_version = 2
domains = ad.example.com
services = nss, pam

...

[domain/ad.example.com]
id_provider = ad
ipa_server = ipaserver.example.com
ipa_hostname = ipa1.example.com
auth_provider = ad
chpass_provider = ad
access_provider = ad

# defines user/group schema type
ldap_schema = ad

# using explicit POSIX attributes in the Windows entries
ldap_id_mapping = False

# caching credentials
cache_credentials = true
enumerate = false

# access controls
ldap_access_order = expire
ldap_account_expire_policy = ad
ldap_force_upper_case_realm = true

# performance
ldap_disable_referrals = true
```

ID マッピングに必須のパラメーターは 2 つあります。Active Directory スキーマはロードする必要があり (**ldap_schema**) ID マッピングは明示的に有効にする必要があります (**ldap_id_mapping**)。

例11.8 ID マッピングによる Active Directory 2008 R2 ドメイン

```
[root@rhel-server ~]# vim /etc/sssd/sssd.conf

[sssd]
config_file_version = 2
domains = ad.example.com
services = nss, pam

...

[domain/ad.example.com]
id_provider = ad
ipa_server = ipaserver.example.com
```

```
ipa_hostname = ipa1.example.com
auth_provider = ad
chpass_provider = ad
access_provider = ad

# defines user/group schema type
ldap_schema = ad

# for SID-UID mapping
ldap_id_mapping = True

# caching credentials
cache_credentials = true
enumerate = false

# access controls
ldap_access_order = expire
ldap_account_expire_policy = ad
ldap_force_upper_case_realm = true

# performance
ldap_disable_referrals = true
```

Active Directory ドメインに可能な設定属性はすべて、**sssd-ldap(5)** および **sssd-ad(5)** の man ページに一覧表示されます。

11.2.13.1.3. LDAP プロバイダーとしての Active Directory の設定

Active Directory はタイプ固有の ID プロバイダーとして設定できますが、Kerberos 認証プロバイダーとともに純然たる LDAP プロバイダーとしても設定可能です。

1. SSSD は、SASL を使って Active Directory サーバーに接続することが推奨されます。これにより、ローカルホストは Linux ホスト上の **Windows ドメイン** 用にサービス keytab を備える必要があります。

   この keytab は Samba を使って作成できます。

   1. **/etc/krb5.conf** ファイルを設定して、Acti Directory レルムを使用します。

```
[logging]
 default = FILE:/var/log/krb5libs.log

[libdefaults]
 default_realm = AD.EXAMPLE.COM
 dns_lookup_realm = true
 dns_lookup_kdc = true
 ticket_lifetime = 24h
 renew_lifetime = 7d
 rdns = false
 forwardable = yes

[realms]
# Define only if DNS lookups are not working
# AD.EXAMPLE.COM = {
```

```
#  kdc = server.ad.example.com
#  admin_server = server.ad.example.com
# }

[domain_realm]
# Define only if DNS lookups are not working
# .ad.example.com = AD.EXAMPLE.COM
# ad.example.com = AD.EXAMPLE.COM
```

2. Samba 設定ファイル、**/etc/samba/smb.conf**、を設定し、Windows Kerberos レルムに向けます。

```
[global]
   workgroup = EXAMPLE
   client signing = yes
   client use spnego = yes
   kerberos method = secrets and keytab
   log file = /var/log/samba/%m.log
   password server = AD.EXAMPLE.COM
   realm = EXAMPLE.COM
   security = ads
```

3. **net ads** コマンドを実行して、管理者プリンシパルとしてログインします。この管理者アカウントにはマシンを Windows ドメインに追加することができる権限が必要ですが、ドメイン管理者権限は必要ありません。

```
[root@server ~]# net ads join -U Administrator
```

4. **net ads** コマンドを再度実行して、ホストマシンをドメインに追加します。これは、ホストプリンシパル (**host/FQDN**) またはオプションで NFS サービス (**nfs/FQDN**) で実行できます。

```
[root@server ~]# net ads join createupn="host/rhel-
server.example.com@AD.EXAMPLE.COM" -U Administrator
```

2. Services for Unix パッケージが Windows サーバー上にインストールされていることを確認してください。

3. SSSD で使用する Windows ドメインを設定します。

   1. Windows マシンで **サーバーマネージャー** を開きます。

   2. Active Directory ドメインサービスの役割を作成します。

   3. **ad.example.com** といった新規ドメインを作成します。

   4. Identity Management for UNIX サービスを Active Directory ドメインサービスの役割に追加します。設定ではドメイン名に Unix NIS ドメインを使用してください。

4. Active Directory サーバー上で Linux ユーザーのグループを作成します。

   1. **管理ツール** を開いて、**Active Directory ユーザーとコンピューター** を選択します。

   2. Active Directory ドメインの **ad.example.com** を選択します。

3. **ユーザー** タブで右クリックして、**新しいグループの作成** を選択します。
4. 新しいグループに **unixusers** と名前を付けて保存します。
5. **unixusers** グループエントリーをダブルクリックして、**ユーザー** タブを開きます。
6. **Unix 属性** タブを開きます。
7. **ad.example.com** 用に設定された NIS ドメインに NIS ドメインをセットして、オプションでグループ ID (GID) 番号を設定します。

5. ユーザーを Unix グループの一部として設定します。
   1. **管理ツール** を開いて、**Active Directory ユーザーとコンピューター** を選択します。
   2. Active Directory ドメインの **ad.example.com** を選択します。
   3. **ユーザー** タブで右クリックして、**新しいユーザーの作成** を選択します。
   4. 新しいユーザーに **aduser** と名前を付けて、**ユーザーは次回のログオン時にパスワード変更が必要** と **Lock account** のチェックボックスにチェックマークが**ついていない**ことを確認します。

      ユーザーを保存します。
   5. **aduser** ユーザーエントリーをダブルクリックして、**Unix 属性** タブを開きます。Unix 設定が、Active Directory ドメインと **unixgroup** グループのものと合致していることを確認します。
      - NIS ドメイン、Active Directory ドメイン用に作成されたもの
      - UID
      - **/bin/bash** へのログインシェル
      - **/home/aduser** へのホームディレクトリー
      - **unixusers** へのプライマリーグループ名

注記

大きなディレクトリーの パスワードルックアップ は、要求 1 件あたり数秒かかります。最初のユーザールックアップは、LDAP サーバーへのコールです。インデックス化されていない検索は、よりリソース集約的なのでインデックス化されている検索よりも時間が長くかかります。これは、サーバーが一致を求めてディレクトリー内のすべてのエントリーをチェックするためです。ユーザールックアップを迅速化するには、SSSD が検索する属性をインデックス化します。

- uid
- uidNumber
- gidNumber
- gecos

6. Linux システム上で SSSD ドメインを設定します。

```
[root@rhel-server ~]# vim /etc/sssd/sssd.conf
```

LDAP プロバイダーパラメーターの完全な一覧は、**sssd-ldap(5)** の man ページを参照してください。

例11.9 Services for Unix による Active Directory 2008 R2 ドメイン

```
[sssd]
config_file_version = 2
domains = ad.example.com
services = nss, pam

...

[domain/ad.example.com]
cache_credentials = true
enumerate = false

# for performance
ldap_referrals = false

id_provider = ldap
auth_provider = krb5
chpass_provider = krb5
access_provider = ldap

ldap_schema = rfc2307bis

ldap_sasl_mech = GSSAPI
ldap_sasl_authid = host/rhel-server.example.com@AD.EXAMPLE.COM

#provide the schema for services for unix
ldap_schema = rfc2307bis

ldap_user_search_base = ou=user accounts,dc=ad,dc=example,dc=com
ldap_user_object_class = user
ldap_user_home_directory = unixHomeDirectory
ldap_user_principal = userPrincipalName

# optional - set schema mapping
# parameters are listed in sssd-ldap
ldap_user_object_class = user
ldap_user_name = sAMAccountName

ldap_group_search_base = ou=groups,dc=ad,dc=example,dc=com
ldap_group_object_class = group

ldap_access_order = expire
ldap_account_expire_policy = ad
ldap_force_upper_case_realm = true
ldap_disable_referrals = true
```

```
krb5_realm = AD-REALM.EXAMPLE.COM
# required
krb5_canonicalize = false
```

7. SSSD を再開始します。

```
[root@rhel-server ~]# service sssd restart
```

11.2.14. ドメインオプション: ユーザー名形式の設定

SSSD が実行する主要なアクションの一つは、リモート ID プロバイダーにおけるローカルシステムユーザーの ID へのマッピングです。SSSD はユーザー名とドメインバックエンド名の組み合わせを使ってログイン ID を作成します。

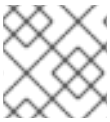
SSSD は、ユーザーが異なるドメインに属している限り、同一のユーザー名を持つ異なるユーザーを認識できます。例えば、SSSD は、**ldap.example.com**ドメインの **jsmith** と **ldap.otherexample.com**ドメインの **jsmith** の両方を問題なく認証できます。

完全なユーザー名の構築に使用する名前の形式は、(オプションで) 設定の **[sssd]** セクションでユニバーサルに定義され、その後に各ドメインセクションで個別に定義することができます。

LDAP や Samba、Active Directory、Identity Management、ローカルシステムなど、異なるサービスのユーザー名は、すべて異なる形式を使用しています。SSSD がユーザー名/ドメイン名のセットの識別に使用する表現は、異なる形式の名前の解釈ができなくてはなりません。この表現は、**re_expression** パラメーターで設定します。

グローバルのデフォルトでは、このフィルターが **name@domain** 形式の名前を構築します。

```
(?P<name>[^@]+)@?(?P<domain>[^@]*$)
```



注記

正規の表現形式は、Python 構文です。

ドメインの部分は、ID プロバイダーのドメイン名に基づいて自動的に提供される場合があります。つまり、ユーザーは **jsmith** としてログインして、このユーザーが例えば LOCAL ドメインに属しているとすると、SSSD はこのユーザーのユーザー名を **jsmith@LOCAL** と解釈します。

しかし、別の ID プロバイダーは別の形式を使用している可能性があります。例えば Samba は非常に厳格な形式を使用しているので、ユーザー名は **DOMAIN\username** という形式に一致する必要があります。このため Samba の正規表現は以下のようになります。

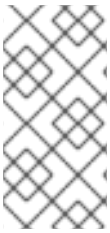
```
(?P<domain>[^\\]*?)\\?(?P<name>[^\\]+$)
```

Active Directory のようなプロバイダーは、複数の異なる名前形式をサポートします。例えば Active Directory と Identity Management は、デフォルトで以下の 3 つの異なる形式をサポートします。

- **username**
- **username@domain.name**
- **DOMAIN\username**

このため Active Directory および Identity Management プロバイダーのデフォルト値は、3 つすべての名前形式を可能にするより複雑なフィルターとなります。

```
(((?P<domain>[^\\]+)\\(?P<name>.+$))|((?P<name>[^@]+)@(?P<domain>.+$))|(^(?
P<name>[^@\\]+)$))
```



注記

jsmith@ldap.example.com といった完全修飾名で情報を要求すると、適切なユーザーアカウントが常に返されます。異なるドメイン内でに同一ユーザー名を持つ複数のユーザーがいると、ユーザー名の特定だけだと、ルックアップの順番で最初にくるドメインのユーザーが返されます。

re_expression はユーザー名形式の設定方法で最も重要なものですが、別のアプリケーションで便利な 2 つのオプションもあります。

デフォルトのドメイン名の値

一つは **re_expression** で、全ユーザーが使用するデフォルトのドメイン名を設定します。(これはグローバル設定で、**[sssd]** セクションのみで利用可能です。) 複数のドメインが設定されている場合があるかもしれませんが、ユーザーデータを保存するのは 1 つだけで、他はホストやサービス ID に使用されます。デフォルトのドメイン名を設定することで、ユーザーはドメイン名を特定せずにユーザー名だけでログインできます。ドメイン名は、プライマリードメイン外のユーザーの場合に必要となります。

```
[sssd]
...
default_domain_suffix = USERS.EXAMPLE.COM
```

出力向けの完全名形式

もうひとつのパラメーターは **re_expression** に関連していますが、ユーザー名の **解釈** 方法を定義するのではなく、特定された名前の **印刷** 方法を定義します。**full_name_format** パラメーターは、ユーザー名とドメイン名 (が決められた後に) の表示方法を設定します。

```
full_name_format = %1$s@%2$s
```

11.2.15. ドメインオプション: オフライン認証の有効化

ドメインサービスについての情報に加えて、ユーザー ID は常にキャッシュされます。しかし、ユーザー **認証情報** は、デフォルトではキャッシュされません。このため、SSSD は常に認証要求のためにバックエンド ID プロバイダーをチェックすることになります。ID プロバイダーがオフラインだったり利用できなかったりすると、これらの認証要求を処理することができず、ユーザー認証に失敗することになります。

SSSD キャッシュにユーザーアカウントの一部として (正常なログイン後に) 認証情報を保存する、**オフライン認証情報キャッシング** を有効にすることができます。つまり、もし ID プロバイダーが利用できなくても、ユーザーは保存された認証情報を使って認証ができることになります。オフライン認証情報キャッシングは本来、個別のドメインエントリーで設定されますが、PAM サービスセクションでセクション可能なオプション設定もいくつかあります。これは、認証情報キャッシングがリモートドメインに加えて、ローカルの PAM サービスとも対話するためです。

```
[domain/EXAMPLE]
cache_credentials = true
```

■

認証情報の有効期限を設定するオプションのパラメーターがあります。有効期限が便利なのは、古くなったアカウントや認証情報を持つユーザーがローカルサービスに無期限にアクセスすることを防ぐことができるからです。

認証情報の有効期限そのものは、PAM サービスで設定され、これがシステム向けに認証要求を処理します。

```
[sssd]
services = nss,pam
...

[pam]
offline_credentials_expiration = 3
...

[domain/EXAMPLE]
cache_credentials = true
...
```

offline_credentials_expiration は、正常なログイン後に、ユーザーの単一認証情報エントリーがキャッシュに保存される日数を設定します。これを 0 に設定すると、エントリーは無期限に保存されます。

認証情報キャッシュに特定して関連しているわけではありませんが、各ドメインには個別のユーザーとサービスキャッシュの有効期限に関する設定オプションがあります。

- **account_cache_expiration** は、正常なログイン後に、ユーザーアカウントエントリー全体が SSSD キャッシュから削除される日数を設定します。これは、個別の認証情報キャッシュ有効期限と同じか、それより長く設定する必要があります。
- **entry_cache_timeout** は、キャッシュに保存されている全エントリーについて、SSSD が ID プロバイダーから更新情報を要求するまでの有効期間 (秒単位) を設定します。また、グループやサービス、netgroup、sudo、autofs エントリーに関する個別のキャッシュタイムアウトのパラメーターもあり、**sssd.conf** man ページに一覧表示されます。デフォルト時間は、5400 秒 (90 分) です。

例:

```
[sssd]
services = nss,pam
...

[pam]
offline_credentials_expiration = 3
...

[domain/EXAMPLE]
cache_credentials = true
account_cache_expiration = 7
entry_cache_timeout = 14400
...
```

11.2.16. ドメインオプション: パスワード有効期限の設定

パスワードポリシーは一般的に、パスワードの有効期間と期限、変更時期を設定します。これらのパスワード有効期限ポリシーは ID プロバイダーを通してサーバー側で評価され、その後に PAM サービスを通して SSSD で警告が処理され、表示されます。

パスワードの警告には設定可能な領域が 2 つあります。

- 全ドメインについて、パスワードの有効期限切れの警告をどの程度前もって表示するかというグローバルデフォルト。これは、PAM サービス向けに設定されます。
- パスワードの有効期限切れの警告をどの程度前もって表示するかというドメインごとに設定。

  ドメインレベルのパスワード有効期限警告を使用する際には、認証プロバイダー (**auth_provider**) もドメイン向けに設定する必要があります。

例:

```
[sssd]
services = nss,pam
...

[pam]
pam_pwd_expiration_warning = 3
...

[domain/EXAMPLE]
id_provider = ipa
auth_provider = ipa
pwd_expiration_warning = 7
```

パスワード有効期限警告が表示されるには、サーバーから SSSD に警告が送信される必要があります。サーバーからパスワード警告が送信されないと、パスワード有効期限が SSSD で設定された期間内であっても、SSSD はメッセージを表示しません。

SSSD でパスワード有効期限警告が設定されていないか、0 に設定されていると、SSSD パスワード警告フィルターが適用されず、サーバー側のパスワード警告が自動的に表示されます。

注記

パスワード警告がサーバーから送信されていれば、PAM もしくはドメインパスワード有効期限は本来、バックエンド ID プロバイダーのパスワード警告設定に優先 (もしくは無視) します。

例えば、バックエンド ID プロバイダーでは警告が 28 日間に設定されていて、SSSD の PAM サービスでは 7 日間に設定されていたとします。プロバイダーは 28 日間で SSSD に警告を送信しますが、SSSD 設定のパスワード有効期限の通り、ローカルでは 7 日前になるまで警告は表示されません。

注記

同様のパラメーターは、Kerberos 認証情報をキャッシュする Kerberos 認証プロバイダー **krb5_store_password_if_offline** の使用時にも利用可能です。

11.2.17. ドメインオプション: DNS Service Discovery の使用

RFC 2782 で定義されている DNS service discovery は、アプリケーションが特定の種類の特定のサービスについてあるドメイン内の SRV 記録をチェックできるようにします。そして、この種類のサービスが発見されたサーバーを返します。

SSSD では、ID プロバイダーと認証プロバイダーは (IP アドレスかホスト名で) 明示的に定義されるか、service discovery を使って動的に発見することができます。**id_provider = ldap** が対応する **ldap_uri** パラメーターなしに設定されるなど、プロバイダーサーバーが一覧表示されない場合は、discovery が自動的に使われます。

DNS discovery クエリは、以下の形式になります。

```
_service._protocol.domain
```

例えば、**example.com** ドメインで TCP を使って LDAP サーバーをスキャンするには、以下のようになります。

```
_ldap._tcp.example.com
```

注記

service discovery を使用するサービスには、特別の DNS レコードを DNS サーバーに追加します:

```
_service._protocol._domain TTL priority weight port hostname
```

SSSD では、サービスの種類はデフォルトで LDAP で、ほとんどすべてのサービスが TCP を使用します (UDP で開始する Kerberos を除く)。service discovery を有効にするには、ドメイン名のみが必要です。デフォルトでは、マシンのホスト名のドメイン部分を使用しますが、別のドメインを指定することもできます(**dns_discovery_domain** パラメーターを使用)。

このため、1 つの例外を除いては、デフォルトでは service discovery に新たな設定を加える必要はありません。パスワード変更プロバイダーはデフォルトで server discovery が無効になっており、サービスの種類を設定して明示的に有効にする必要があります。

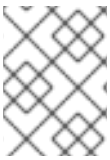
```
[domain/EXAMPLE]
...
chpass_provider = ldap
ldap_chpass_dns_service_name = ldap
```

設定は不要ですが、異なる DNS ドメイン (**dns_discovery_domain**) を使ったり、スキャンに異なるサービスの種類を設定したりすることで、server discovery はカスタマイズが可能です。例えば、以下のようになります。

```
[domain/EXAMPLE]
id _provider = ldap

dns_discovery_domain = corp.example.com
ldap_dns_service_name = ldap

chpass_provider = krb5
ldap_chpass_dns_service_name = kerberos
```

最後に、service discovery はバックアップサーバーと使われることは決してありません。これは、プロバイダーのためにプライマリーサーバーと使われるものです。つまり、ディスカバリはまずサーバーの位置を確認するために使われ、その後に SSSD がバックアップサーバーの使用に戻ります。ディスカバリをプライマリーサーバーに使用するには、**_srv_** をプライマリーサーバーの値として使用し、その後にバックアップサーバーを一覧表示します。以下に例を挙げます。

```
[domain/EXAMPLE]
id _provider = ldap
ldap_uri = _srv_
ldap_backup_uri = ldap://ldap2.example.com

auth_provider = krb5
krb5_server = _srv_
krb5_backup_server = kdc2.example.com

chpass_provider = krb5
ldap_chpass_dns_service_name = kerberos
ldap_chpass_uri = _srv_
ldap_chpass_backup_uri = kdc2.example.com
```



注記

サービスディスカバリは、バックアップサーバーでは使用できません。プライマリーサーバーのみとなります。

DNS ルックアップがホスト名の IPv4 アドレスを返すことに失敗した場合、SSSD は失敗を返す前に IPv6 アドレスのルックアップを試みます。これにより確実に非同期のリゾルバーが正しいアドレスを識別するようになります。

ホスト名解決動作は、**sssd.conf** 設定ファイルの ***lookup family order*** オプションで設定します。

11.2.18. ドメインオプション: 証明書のサブジェクト名での IP アドレスの使用 (LDAP のみ)

ldap_uri オプション内にサーバー名の代わりに IP アドレスを使用すると、TLS/SSL 接続の失敗につながる場合があります。TLS/SSL 証明書は IP アドレスではなく、サーバー名を含んでいます。しかし、証明書の **サブジェクトの別名** フィールドはサーバーの IP アドレスを含んで使用できるため、IP アドレスを使用して正しい安全な接続が許可されます。

1. 既存の証明書を要求される証明書に変更します。署名済みのキー (**-signkey**) とは、証明書を発行した本来の CA 発行者のキーです。これが外部の CA によって実行されていた場合、個別の PEM ファイルが必要になります。証明書が自己署名型の場合、それが証明書そのものです。例えば:

   ```
   openssl x509 -x509toreq -in old_cert.pem -out req.pem -signkey
   key.pem
   ```

   自己署名証明書の使用:

   ```
   openssl x509 -x509toreq -in old_cert.pem -out req.pem -signkey
   old_cert.pem
   ```

2. **/etc/pki/tls/openssl.cnf** 設定ファイルを編集して、***[v3_ca]*** セクションの下にサーバーの IP アドレスを含めてます。

```
subjectAltName = IP:10.0.0.10
```

3. 生成された証明書要求を使用して、指定した IP アドレスを持つ新規の自己署名証明書を生成します:

```
openssl x509 -req -in req.pem -out new_cert.pem -extfile
./openssl.cnf -extensions v3_ca -signkey old_cert.pem
```

-extensions オプションは証明書で使用する拡張子をセットします。これには、適切なセクションをロードする v3_ca となる必要があります。

4. **old_cert.pem** ファイルのプライベートキーブロックを **new_cert.pem** ファイルにコピーして、すべての関連情報を 1 つのファイルに保管します。

nss-utils パッケージで提供される **certutil** ユーティリティを介して証明書を作成する時、**certutil** は、証明書作成の目的のみで DNS サブジェクトの別名をサポートすることに注意してください。

11.2.19. ドメインの作成: Proxy

SSSD を持つプロキシは単に中継点であり、仲介の設定です。SSSD はそのプロキシサービスに接続して、それからそのプロキシが指定されたライブラリをロードします。これにより SSSD は他の状況では使用できない一部のリソースを使用できるようになります。例えば、SSSD は認証プロバイダーとしては LDAP と Kerberos のみをサポートしますが、プロキシを使用すると SSSD は指紋スキャナーやスマートカードなどの代替の認証メソッドを使用できるようになります。

表11.9 Proxy ドメイン設定のパラメータ

パラメーター	詳細
proxy_pam_target	PAM が認証プロバイダーとしてプロキシする必要のあるターゲットを指定します。PAM のターゲットとは、デフォルトの PAM ディレクトリ、**/etc/pam.d/** 内に PAM スタック情報を含んでいるファイルのことです。 これは認証プロバイダーの代理に使用されます。 **重要** プロキシ PAM スタックに **pam_sss.so** が再帰的に**格納されていない**ことを確認してください。
proxy_lib_name	識別要求の代理をする経由先の既存の NSS ライブラリを指定します。 以下が識別プロバイダーの代理に使用されます。

例11.10 プロキシ識別と Kerberos 認証

プロキシライブラリは、**proxy_lib_name** パラメータを使用してロードされます。このライブラリは、任意の認証サービスと互換性がある限りどんな役目もします。Kerberos 認証プロバイダーに対しては、NIS のように Kerberos 互換のライブラリでなければなりません。

```
[domain/PROXY_KRB5]
auth_provider = krb5
krb5_server = kdc.example.com
krb5_realm = EXAMPLE.COM

id_provider = proxy
proxy_lib_name = nis
enumerate = true
cache_credentials = true
```

例11.11 LDAP 識別とプロキシ認証

プロキシライブラリは、**proxy_pam_target** パラメータを使用してロードされます。このライブラリは該当する識別プロバイダーと互換性のある PAM モジュールでなければなりません。例えば、以下は LDAP と一緒に PAM 指紋モジュールを使用します。

```
[domain/LDAP_PROXY]
id_provider = ldap
ldap_uri = ldap://example.com
ldap_search_base = dc=example,dc=com

auth_provider = proxy
proxy_pam_target = sssdpamproxy
enumerate = true
cache_credentials = true
```

SSSD デーモンを設定した後には、指定した PAM ファイルが設定されていることを確認します。この例では、ターゲットは**sssdpamproxy** なので、**/etc/pam.d/sssdpamproxy** ファイルを作成して PAM/LDAP モジュールをロードします。

```
auth          required      pam_frprint.so
account       required      pam_frprint.so
password      required      pam_frprint.so
session       required      pam_frprint.so
```

例11.12 識別プロキシと認証プロキシ

SSSD は、識別プロキシと認証プロキシの両方を持つドメインを使用できます。そのため関与する唯一の設定はプロキシの設定です。認証 PAM モジュール用の **proxy_pam_target** と、NIS や LDAP の様なサービス用の **proxy_lib_name** の設定があります。

この例では可能な設定を説明していますが、現実的的な設定ではありません。ID と認証に LDAP が使われる場合は、ID プロバイダーと認証プロバイダーの両方が LDAP に設定されるべきで、プロキシではありません。

```
[domain/PROXY_PROXY]
auth_provider = proxy
```

```
id_provider = proxy
proxy_lib_name = ldap
proxy_pam_target = sssdproxyldap
enumerate = true
cache_credentials = true
```

SSSD ドメインが追加されると、その後にシステムのセッティングを更新してプロキシサービスを設定します:

1. **pam_ldap.so** モジュールを必要とする **/etc/pam.d/sssdproxyldap** ファイルを作成します:

```
auth          required      pam_ldap.so
account       required      pam_ldap.so
password      required      pam_ldap.so
session       required      pam_ldap.so
```

2. **nss-pam-ldap** パッケージがインストールされていることを確認して下さい。

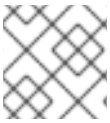
```
[root@server ~]# yum install nss-pam-ldap
```

3. LDAP ネームサービスデーモン用の設定ファイルである **/etc/nslcd.conf** の編集により、LDAP ディレクトリ用の情報を含むようにします:

```
uid nslcd
gid ldap
uri ldaps://ldap.example.com:636
base dc=example,dc=com
ssl on
tls_cacertdir /etc/openldap/cacerts
```

11.2.20. ドメインの作成: Kerberos 認証

LDAP とproxy 識別の両プロバイダーは別々の Kerberos ドメインを使用して認証を供給できます。Kerberos 認証プロバイダーの設定には、*key distribution center (キー配布センター)* (KDC) と Kerberos ドメインが必要です。すべてのプリンシパル名は指定された識別プロバイダー内で利用可能でなければなりません。そうでない場合は、SSSD が形式 **username@REALM** を使用してプリンシパルを構築します。



注記

Kerberos は、認証のみを提供できます。識別データベースは提供できません。

SSSD は、Kerberos KDC が Kerberos kadmin サーバーでもあることを想定しています。しかし、実稼働環境では、KDC の読み込み専用のレプリカが複数あり、kadmin サーバーは 1 つのみと言うのが一般的になっています。**krb5_kpasswd** オプションを使用することで、パスワード変更サービスが稼働する場所を、またはそれが非デフォルトのポートで稼働するかどうかを指定します。**krb5_kpasswd** オプションが指定されていない場合は、SSSD は、Kerberos KDC を使用してパスワード変更をします。

基本的な Kerberos 設定オプションは 表11.10「Kerberos 認証設定のパラメータ」 に一覧表示してあります。Kerberos 設定オプションについての情報には **sssd-krb5(5)** の man ページを参照してください。

例11.13 基本的な Kerberos 認証

```
# A domain with identities provided by LDAP and authentication by
Kerberos
[domain/KRBDOMAIN]
enumerate = false
id_provider = ldap
chpass_provider = krb5
ldap_uri = ldap://ldap.example.com
ldap_search_base = dc=example,dc=com
ldap-tls_reqcert = demand
ldap_tls_cacert = /etc/pki/tls/certs/ca-bundle.crt

auth_provider = krb5
krb5_server = kdc.example.com
krb5_backup_server = kerberos.example.com
krb5_realm = EXAMPLE.COM
krb5_kpasswd = kerberos.admin.example.com
krb5_auth_timeout = 15
```

例11.14 Kerberos チケット更新オプションの設定

Kerberos 認証プロバイダーのタスクの一つは、ユーザーとサービスに対してチケット保証チケット (TGT) を要求することです。このチケットは、チケットのプリンシパルであるユーザーがアクセスすることで、特定サービスに動的に他のチケットを生成するために使われるます。

ユーザー本人に最初に与えられた TGT は、そのチケットのライフタイムの間のみ有効です (デフォルトでは、設定済み KDC の設定)。その後は、チケットの更新や延長はできません。しかし、チケットを更新しないと、実行中のサービスにアクセスしようとする際やチケットの期限が切れた時に問題が起きるサービスもあります。

Kerberos チケットはデフォルトでは更新できませんが、**krb5_renewable_lifetime** や **krb5_renew_interval** パラメーターを使ってチケット更新を有効にできます。

チケットのライフタイムは **krb5_lifetime** パラメーターで SSSD 内で設定します。これはチケットの有効期間を指定し、KDC の他のどの値にも優先します。

チケット更新自体は **krb5_renewable_lifetime** パラメーターで有効化され、更新すべてを集計してチケットの最大ライフタイムを設定します。

例えば以下では、チケットのライフタイムは 1 時間に、更新可能なライフタイムは 24 時間に設定されます。

```
krb5_lifetime = 1h
krb5_renewable_lifetime = 1d
```

この設定では、チケットは 1 時間ごとに有効期限が切れ、最大 1 日間は続けて更新が可能になります。

ライフタイムと更新可能ライフタイムの値は、秒数 (s)、分数 (m)、時間 (h)、日数 (d) のいずれかにできます。

チケット更新で設定する必要のある別のオプションは、**krb5_renew_interval** パラメーターです。これは、チケットの更新が必要かどうかを SSSD がチェックする頻度を設定します。この設定に関係なく、チケットのライフタイムが半分になった時点で、チケットは自動的に更新されます (この値は常に秒単位)。

```
krb5_lifetime = 1h
krb5_renewable_lifetime = 1d
krb5_renew_interval = 60s
```

注記

krb5_renewable_lifetime の値が設定されていなかったり、**krb5_renew_interval** パラメーターが設定されていないか 0 に設定されていたりすると、チケット更新は無効になります。チケット更新の有効化には、**krb5_renewable_lifetime** と **krb5_renew_interval** の両方が必要です。

表11.10 Kerberos 認証設定のパラメータ

パラメーター	詳細
chpass_provider	パスワード変換操作に使用するサービスを指定します。これは認証プロバイダーと同じと想定されています。Kerberos を使用するには、これを **krb5** に指定します。
krb5_server	SSSD が接続先のプライマリー Kerberos サーバーを IP アドレスかホスト名で提供します。
krb5_backup_server	プライマリーサーバーが利用できない場合、SSSD の接続先となる Kerberos サーバーの IP アドレス、またはホスト名のカンマ区切りの一覧を提供します。この一覧は設定順に提供されるため、一覧の最初のサーバーが最初に試行されます。 SSSD は 1 時間後に ***krb5_server*** パラメーターで指定されているプライマリーサービスに再接続を試みます。 KDC サーバーまたは kpasswd サーバー用にサービスディスカバリを使用している場合は、SSSD は、UDP を接続プロトコルとして指定する DNS エントリを最初に検索して、それから TCP に切り替えます。
krb5_realm	KDC が実行する Kerberos レルムを識別します。
krb5_lifetime	指定したライフタイムを秒 (s)、分 (m)、時間 (h)、または日 (d) の表示で持つ Kerberos チケットを要求します。

パラメーター	詳細
krb5_renewable_lifetime	合計ライフタイムを指定した秒 (s)、分 (m)、時間 (h)、または日 (d) の表示で持つ更新可能な Kerberos チケットを要求します。
krb5_renew_interval	チケットが更新されるべきかどうかをチェックする SSSD の時間を秒で指定します。チケットはそのライフタイムの半分が経過すると自動的に更新されます。このオプションが欠如していたり、またはゼロにセットしてある場合は、自動チケット更新システムは無効になっています。
krb5_store_password_if_offline	Kerberos の認証プロバイダーがオフラインである場合に、ユーザーパスワードを保管しそのプロバイダーがオンラインに戻った時にチケットを要求するためにそのキャッシュを使用するかどうかを指定します。デフォルトは **`false`** になっており、パスワードは保管しません。
krb5_kpasswd	パスワード変更サービスが KDC 上で稼働していない場合、代替の Kerberos kadmin サービスを一覧表示します。
krb5_ccname_template	ユーザーの認証情報キャッシュを保管するために使用するディレクトリを提供します。これはテンプレート化が可能であり、以下のトークンがサポートされています: • **%u**、ユーザーのログイン名 • **%U**、ユーザーのログイン UID • **%p**、ユーザーのプリンシパル名 • **%r**、レルム名 • **%h**、ユーザーのホームディレクトリ • **%d**、**`krb5ccache_dir`** パラメータの値 • **%P**、SSSD クライアントのプロセス ID • **%%**、文字通りのパーセントマーク (%) • **XXXXXX**、SSSD に対して特有のファイル名を安全に作成するように指示するテンプレート末尾の文字列 例: `krb5_ccname_template = FILE:%d/krb5cc_%U_XXXXXX`

パラメーター	詳細
krb5_ccachedir	認証情報のキャッシュを保管するためのディレクトリを指定します。**%d** と **%P** を除いては、**krb5_ccname_template** と同じトークンを使用してテンプレート化できます。**%u**、**%U**、**%p**、または **%h** が使用される場合は、SSSD は各ユーザー用にプライベートディレクトリを作成します。それ以外の場合は、パブリックディレクトリを作成します。
krb5_auth_timeout	オンライン認証、またはパスワード変更要求が中断されるまでの時間を秒で提示します。可能な場合は、認証要求はオフラインで継続します。デフォルトは15秒です。

11.2.21. ドメインの作成: アクセス制御

SSSD はドメイン設定の為に初歩的なアクセス制御を提供して、単純なユーザーallow/deny 一覧か、または LDAP バックエンド自身の使用を可能にします。

11.2.21.1. Simple Access Provider の使用

Simple Access Provider (シンプルアクセスプロバイダー) はユーザー名またはグループの一覧を基にして許可または否定をします。

Simple Access Provider とは、特定のマシンへのアクセスを制限する手段です。例えば、ある企業がラップトップを使用している場合、Simple Access Provider は特定のユーザー、または特定のグループのみにアクセスを制限するのに使用できます。異なるユーザーが同じ認証プロバイダーに対して正しく認証される場合も制限は機能します。

最も一般的なオプションは **simple_allow_users** と**simple_allow_groups** であり、これらは明示的に特定のユーザー (該当するユーザーまたはグループメンバー) にアクセスを許可して、他の人にはアクセスを拒否します。また、deny リストを作成することもできます (明示した人々にアクセスを拒否して他の人には暗黙に許可する)。

Simple Access Provider は、以下にあげる 4 つのルールに従って、どのユーザーにアクセスを許可すべき/すべきでないかを決定します。

- allow (許可) と deny (拒否)の両方のリストが空の場合には、アクセスが許可されます。
- いかなるものでもリストが提供される場合は、allow ルールが最初に、その後に deny ルールが評価されます。実際には、deny ルールが allow ルールに優先することになります。
- allow リストが提供される場合は、そのリストに載っているユーザー以外はすべて、アクセスが拒否されます。
- deny リストのみが提供される場合は、そのリストに載っているユーザー以外はすべて、アクセスが許可されます。

この例では、二人のユーザーと IT グループに所属する全員にアクセスが許可されます。暗黙的に、他のすべてのユーザーは拒否されます。

```
[domain/example.com]
```

```
access_provider = simple
simple_allow_users = jsmith,bjensen
simple_allow_groups = itgroup
```

注記

SSSD 内の LOCAL ドメインは **simple** をアクセスプロバイダーとしてサポートしません。

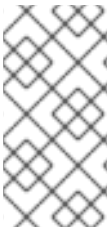
他のオプションは **sssd-simple** man ページに一覧表示してありますが、これらはたまにしか使用されません。

11.2.21.2. LDAP アクセスフィルターの使用

LDAP、Active Directory、Identity Management サーバーは、ドメイン用のアクセス制御ルールを提供できます。関連したフィルターオプション(**ldap_access_filter**) は、特定のホストへアクセスを許可されるユーザーを指定します。ユーザーフィルターを使用しないとすべてのユーザーがアクセスを拒否されます。

例:

```
[domain/example.com]
access_provider = ldap
ldap_access_filter = memberOf=cn=allowedusers,ou=Groups,dc=example,dc=com
```



注記

LDAP アクセスプロバイダーのオフラインキャッシングは、ユーザーの前回のオンラインログイン試行が正常であったかどうかの判定に限定されます。前回のログイン中にアクセスを許可されたユーザーはオフライン中でもアクセスの許可が継続することになります。

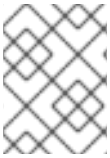
SSSD は、エントリーの ***authorizedService*** や ***host*** 属性で結果をチェックすることもできます。実際、ユーザーエントリーや設定によって、LDAP フィルター、 ***authorizedService***、***host*** のすべての評価が可能です。**ldap_access_order** パラメーターは、使用するすべてのアクセス制御方法を評価方法の順番で一覧表示します。

```
[domain/example.com]
access_provider = ldap
ldap_access_filter = memberOf=cn=allowedusers,ou=Groups,dc=example,dc=com
ldap_access_order = filter, host, authorized_service
```

認証済みサービスや許可されたホストの評価に使用するユーザーエントリー内の属性は、カスタマイズが可能です。追加のアクセス制御パラメーターは、**sssd-ldap(5)** man ページに一覧表示されます。

11.2.22. ドメインの作成: プライマリーサーバーとバックアップサーバー

ドメイン用の ID および認証プロバイダーは、自動フェイルオーバー向けの設定が可能です。SSSD は最初に、指定されたプロバイダーサーバーへの接続を試みます。このサーバーに到達できない場合は、リスト化されたバックアップサーバーに順番に接続を試みます。



注記

SSSD は、プライマリーサーバーへの接続が再確立されるまで、30 秒ごとに接続を試み、再確立ができるとバックアップからプライマリーにスイッチします。

メジャーなサービス領域にはすべて、プライマリーとバックアップサーバー向けのオプション設定があります[4]。

表11.11 プロバイダーおよびセカンダリーサーバーのパラメーター

サービスエリア	プライマリーサーバーの属性	バックアップサーバーの属性
LDAP ID プロバイダー	ldap_uri	ldap_backup_uri
Active Directory ID プロバイダー	ad_server	ad_backup_server
Identity Management (IdM または IPA) ID プロバイダー	ipa_server	ipa_backup_server
Kerberos 認証プロバイダー	krb5_server	krb5_backup_server
Kerberos 認証プロバイダー	krb5_server	krb5_backup_server
パスワード変更プロバイダー	ldap_chpass_uri	ldap_chpass_backup_uri

プライマリーサーバーとして設定可能なのは、1 つのサーバーのみです。(オプションでは、ホスト名ではなく **_srv_** を使ってプライマリーサーバーをサービスディスカバリ向けに設定することができます。) 複数のバックアップサーバーは、コンマ区切りのリストで設定できます。バックアップサーバーのリストは優先順位で並んでいるので、最初のサーバーが最初に試されます。

```
[domain/EXAMPLE]
id_provider = ad
ad_server = ad.example.com
ad_backup_server = ad1.example.com, ad-backup.example.com
```

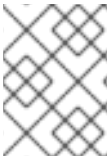
11.2.23. SSSD ユーティリティのインストール

SSSD キャッシュやユーザーエントリー、グループエントリーを扱う追加ツールは、**sssd-tools** パッケージに含まれています。このパッケージは必須ではありませんが、ユーザーアカウントの管理に役立ちます。

```
[root@server ~]# yum install sssd-tools
```

11.2.24. ローカルシステムユーザーの作成

ユーザーがログインするのを待って追加するよりも、ユーザーを SSSD データベースにシードする方が便利な場合があります。



注記

ユーザーアカウントを手動で追加するには、**sssd-tools** パッケージをインストールする必要があります。

新規システムユーザーを作成する場合、SSSD ローカル ID プロバイダードメイン内にユーザーを作成することができます。これは、新規システムユーザーの作成や SSSD 設定のトラブルシューティング、特定もしくはネスト化されたグループの作成に便利です。

新規ユーザーは、**sss_useradd** コマンドを使って追加できます。

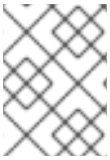
最も基本的なのは **sss_useradd** コマンドで、これは新規ユーザー名のみを必要とします。

```
[root@server ~]# sss_useradd jsmith
```

他のオプション (**sss_useradd(8)** man ページに一覧表示) は、ユーザーが所属する UID や GID、ホームディレクトリー、グループなどのアカウントの属性の設定に使用できます。

```
[root@server ~]# sss_useradd --UID 501 --home /home/jsmith --groups
admin,dev-group jsmith
```

11.2.25. キックスタート中にユーザーを SSSD キャッシュにシードする方法



注記

ユーザーアカウントを手動で追加するには、**sssd-tools** パッケージをインストールする必要があります。

SSSD では、リモートドメインのユーザーは、そのユーザーの ID が ID プロバイダーから取得されるまで、ローカルシステムの利用ができません。しかし、ユーザーがログインするまで利用できないネットワークインターフェースもいくつかあり、ユーザー ID がネットワーク上のどこかにある場合は、ユーザーはログインできません。この場合、適切なドメインに関連したそのユーザーの ID で SSSD キャッシュをシードすることが可能で、これによりユーザーはローカルでログインし、適切なインターフェースを有効にできます。

これは以下のように **sss_seed** ユーティリティを使用して行います。

```
sss_seed --domain EXAMPLE.COM --username testuser --password-file
/tmp/sssd-pwd.txt
```

このユーティリティは、少なくともユーザー名とドメイン名、パスワードを識別するオプションを必要とします。

- **--domain** は SSSD 設定からドメイン名を提供します。このドメイン名は、SSSD 設定に存在している必要があります。
- **--username** は、ユーザーアカウントの短縮名です。
- **--password-file** は、シードエントリーの一時的なパスワードを含むファイルのパスおよび名前用です。ユーザーアカウントが SSSD キャッシュに既にある場合は、このファイルの一時的なパスワードが SSSD キャッシュの保存済みパスワードを上書きします。

追加のアカウント設定オプションは、**sss_seed(8)** man ページに一覧表示されます。

これはほとんどの場合、キックスタートもしくは自動セットアップの一部として実行されるので、SSSD を有効にし、SSSD ドメインをセットアップし、パスワードファイルを作成する大規模なスクリプトの一部となります。以下に例を挙げます。

```
authconfig --enablesssd --enablesssdauth --update

function make_sssd {
cat <<- _EOF_
[sssd]
domains = LOCAL
services = nss,pam

[nss]

[pam]

[domain/LOCAL]
id_provider = local
auth_provider = local
access_provider = permit

_EOF_
}

make_sssd >> /etc/sssd/sssd.conf

function make_pwdfile {
cat <<1 _EOF_
password
_EOF_
}

make_pwdfile >> /tmp/sssd-pwd.txt

sss_seed --domain EXAMPLE.COM --username testuser --password-file
/tmp/sssd-pwd.txt
```

11.2.26. SSSD キャッシュの管理

SSSD は、同じタイプのドメインと異なるタイプのドメインを複数定義することができます。SSSD はそれぞれのドメイン用に別々のデータベースファイルを維持できます。これは各ドメインがそれ自身のキャッシュを持っていることを意味します。これらのキャッシュファイルは **/var/lib/sss/db/** ディレクトリ内に収納されています。

11.2.26.1. SSSD キャッシュの削除

LDAP アップデートはドメイン用の ID プロバイダーになされるため、新情報を迅速にリロードするためにキャッシュをクリアにする必要があります。

キャッシュ削除ユーティリティの **sss_cache** は、SSSD キャッシュにあるユーザーやドメイン、グループのレコードを無効にします。最新のレコードを無効にすることで、キャッシュは ID プロバイダーからアップデートされたレコードを強制的に取得するため、変更が迅速に実現できます。

注記

このユーティリティは、**sssd** パッケージの SSSD に含まれています。

最も一般的には、キャッシュのクリアとドメイン全体のレコードをアップデートするためにこのユーティリティが使われます。

例11.15 ドメインレコードの削除

```
[root@server ~]# sss_cache -d LDAP1
```

管理者が特定のレコード (ユーザー、グループ、ネットグループのいずれか) がアップデートされたことを認識している場合、**sss_cache** はその特定のアカウントのレコードを削除し、キャッシュの残りをそのまま残しておけます。

例11.16 ユーザーレコードの削除

```
[root@server ~]# sss_cache -u jsmith
```

表11.12 sss_cache オプション

短い引数	長い引数	詳細
-d **name**	--domain **name**	特定ドメイン内のユーザーやグループ、他のエントリーのキャッシュエントリーのみを無効にします。
-G	--groups	グループレコードすべてを無効にします。**-g** も使われた場合は、**-G** が優先され、 **-g** は無視されます。
-g **name**	--group **name**	特定グループのキャッシュエントリーを無効にします。
-N	--netgroups	ネットグループのキャッシュレコードすべてを無効にします。**-n** も使われた場合、**-N** が優先され **-n** は無視されます。
-n **name**	--netgroup **name**	特定ネットグループのキャッシュエントリーを無効にします。
-U	--users	ユーザーレコードすべてを無効にします。**-u** も使われた場合、**-U** が優先され **-u** は無視されます。

短い引数	長い引数	詳細
-u **name**	--user **name**	特定ユーザーのキャッシュエントリーを無効にします。

11.2.26.2. ドメインキャッシュファイルの削除

すべてのキャッシュファイルにはドメイン用の名前があります。例えば、**exampleldap** と言うドメインには、キャッシュファイル名は **cache_exampleldap.ldb** となります。

キャッシュファイルを削除する場合は、注意してください。 この操作は重大な影響を及ぼします。

- キャッシュファイルを削除すると、識別情報とキャッシュされている認証情報の両方でユーザー情報すべてが削除されます。従って、システムがオンラインでドメインのサーバーに対してユーザー名で認証できる状態でない場合は、キャッシュファイルを削除しないでください。認証情報のキャッシュがないと、オフライン認証は失敗します。
- 異なる識別プロバイダーを参照するように設定が変更された場合、SSSD は、変更前のプロバイダーからのキャッシュされたエントリがタイムアウトになるまで、両方のプロバイダーからのユーザーを認識します。

  キャッシュを排除することでこれを避けることは可能ですが、それより良いオプションとして新規のプロバイダーに異なるドメイン名を使用することができます。SSSD が再開始する際に、新しい名前で新規ファイルが作成されて、古いファイルは無視されます。

11.2.27. SSSD で NSCD の使用

SSSD は、NSCD デーモンと併用するようには設計されていません。SSSD は直接的に NSCD と競合することはありませんが、両方のサービスの併用は、特にエントリがキャッシュできる期間の長さに関して予期しない動作を起こす可能性があります。

問題が判明する最も一般的な証拠は NFS との競合です。ネットワーク接続を管理するために Network Manager を使用している際に、ネットワークインターフェイスが立ち上がるのに数分かかるかも知れません。この期間中に各種のサービスが開始を試みます。ネットワークの起動して DNS サーバーが利用可能になる前にこれらのサービスが開始した場合、これらのサービスは必要となる順引き、または逆引きの DNS の識別に失敗します。よってこれらのサービスは不正なまたは、多分空の **resolv.conf** ファイルを読み込むことになります。このファイルは標準的に読み込み専用であるため、このファイルへの変更は自動的に適用されません。サービスが手動で再開始されない限り、この状態が、NSCD サービスが実行しているマシン上で NFS は失敗する原因になります。

この問題を避けるためには、**/etc/nscd.conf** ファイル内でホストとサービス用にキャッシングを有効にすることで、**passwd**、**group**、及び **netgroup** のエントリの為の SSSD キャッシュに依存します。

/etc/nscd.conf ファイルを変更します:

```
enable-cache hosts yes
enable-cache passwd no
enable-cache group no
enable-cache netgroup no
```

NSCD がホストの要求に応答している時、これらのエントリは NSCD によってキャッシュされてから、ブートプロセス中に NSCD によって返されます。他のエントリはすべて SSSD によって処理されます。

11.2.28. SSSD のトラブルシューティング

- 「SSSD ドメイン用のデバッグログの設定」
- 「SSSD ログファイルのチェック」
- 「SSSD 設定に伴う問題」

11.2.28.1. SSSD ドメイン用のデバッグログの設定

ドメインはそれぞれのデバッグログレベルを設定します。ログレベルを上げると、SSSD やドメイン設定での問題についての情報をより多く提供できます。

ログレベルを変更するには、追加のログを作成する **sssd.conf** ファイルで各セクションの ***debug_level*** パラメーターを設定します。以下に例を挙げます。

```
[domain/LDAP]
enumerate = false
cache_credentials = true
debug_level = 9
```

表11.13 デバッグログレベル

レベル	詳細
0	致命的な失敗。SSSD の起動を妨げたり、SSSD の実行を停止させるもの。
1	重大な失敗。SSSD を強制終了するわけではないが、少なくとも 1 つのメジャーな機能が適切に動作していないことを示すエラー。
2	深刻な失敗。特定の要求や操作が失敗したことを告げるエラー。
3	マイナーな失敗。2 のオペレーション失敗を引き起こしかねないエラー。
4	設定セッティング
5	機能データ
6	操作関数のメッセージを追跡します。
7	内部制御関数のメッセージを追跡します。

レベル	詳細
8	関心のある可能性がある関数-内部変数のコンテンツ。
9	非常に低いレベルの追跡情報。

注記

SSSD 1.8 よりも古いバージョンでは、デバッグログレベルは **[sssd]** セクションでグローバル設定が可能です。各ドメインとサービスでは、それぞれのデバッグログレベルを設定する必要があります。

SSSD 設定ファイルのそれぞれの設定エリアにグローバル SSSD のデバッグログレベルをコピーするには、**sssd_update_debug_levels.py** スクリプトを使います。

```
python -m SSSDConfig.sssd_update_debug_levels.py
```

11.2.28.2. SSSD ログファイルのチェック

SSSD は、その運用に関する情報を報告するために多くのログファイルを使用し、それを **/var/log/sssd/** ディレクトリに配置します。SSSD は各ドメイン用のログファイルだけでなく、**sssd_pam.log** ファイルと **sssd_nss.log** ファイルも作製します。

更には、**/var/log/secure** ファイルは認証の失敗とその失敗の理由もログします。

11.2.28.3. SSSD 設定に伴う問題

問： SSSD の起動失敗

答： SSSD はデーモンが開始する前にすべての要求されたエントリを使用して設定ファイルが適切にセットアップされることを要求します。

SSSD には、サービスが開始する前に、少なくとも 1 つのドメインが適切に設定されていることが必要です。このようなドメインがない場合には、SSSD の起動を試みた際に、ドメインが設定されていないと言うエラーを返します:

```
# sssd -d4

[sssd] [ldb] (3): server_sort:Unable to register control with
rootdse!
[sssd] [confdb_get_domains] (0): No domains configured, fatal
error!
[sssd] [get_monitor_config] (0): No domains configured.
```

/etc/sssd/sssd.conf のファイルを編集して、少なくとも 1 つのドメインを作成します。

SSSD には、起動する前に、少なくとも 1 つの利用可能なサービスプロバイダーが必要です。問題がサービスプロバイダーの設定にある場合、サービスが設定されていないことを示すエラーメッセージが表示されます:

```
[sssd] [get_monitor_config] (0): No services configured!
```

/etc/sssd/sssd.conf ファイルを編集して、少なくとも 1 つのサービスプロバイダーを設定します。

重要

SSSD では、**/etc/sssd/sssd.conf** ファイル内の単一の ***services*** エントリに、カンマ区切りの一覧としてサービスプロバイダーを設定する必要があります。サービスが複数のエントリに記載されている場合は、最後のエントリのみが SSSD によって認識されます。

問： 'id' のあるグループや 'getent group' のあるグループメンバーが見当たりません。

答： これは、**sssd.conf** の **[domain/DOMAINNAME]** セクションにある **ldap_schema** の誤った設定が原因である可能性があります。

SSSD は、RFC 2307 と RFC 2307bis スキーマタイプをサポートします。デフォルトでは、SSSD はより一般的な RFC 2307 スキーマを使用します。

RFC 2307 と RFC 2307bis の違いは、グループメンバーシップが LDAPサーバーに保存される方法の違いです。RFC 2307 サーバーでは、グループメンバーは複数値の ***memberuid*** 属性として保存され、これにはメンバーであるユーザーの名前が含まれます。RFC2307bis サーバーでは、グループメンバーは複数値の ***member*** または ***uniqueMember*** 属性として保存され、これにはこのグループのメンバーであるユーザーもしくはグループの DN が含まれます。RFC2307bis は、ネスト化されたグループのメンテナンスも可能にします。

グループルックアップがいかなる情報も返さない場合は、以下を行います。

1. **ldap_schema** を **rfc2307bis** に設定。
2. **/var/lib/sss/db/cache_DOMAINNAME.ldb** を削除。
3. SSSD を再起動。

これが機能しない場合は、以下の行を **sssd.conf** に追加します。

```
ldap_group_name = uniqueMember
```

その後にキャッシュを削除して、再度 SSSD を再起動します。

問： LDAP に対する認証が失敗します。

答： 認証を実行するには、SSSD は通信チャンネルの暗号化を要求します。このため、**sssd.conf** が標準プロトコル (**ldap://**) で接続するように設定されている場合、Start TLS との通信チャンネルを暗号化しようとします。**sssd.conf** がセキュアなプロトコル (**ldaps://**)で接続するように設定されている場合は、SSSD は SSL を使用します。

つまり、LDAP サーバーは SSL または TLS で実行するように設定される必要があります。TLS は標準 LDAP ポート (389) で、SSL はセキュアな LDAPS ポート (636) で有効とする必要があります。SSL と TLS のいずれの場合も、LDAP サーバーは有効な証明書信頼で設定される必要があります。

無効な証明書信頼は、LDAP に対する認証における最も一般的な問題の一つです。クライアントに LDAP サーバー証明書の適切な信頼がないと、接続を確認できず、SSSD はパスワードの送信を拒否します。LDAP プロトコルでは、パスワードがプレーンテキストで LDAP サーバーに送信されることが要求されます。暗号化されていない接続でパスワードをプレーンテキストで送信することは、セキュリティー問題となります。

証明書が信頼されないと、**syslog** メッセージが書き込まれ、TLS 暗号化が開始できないことを意味します。証明書設定は、LDAP サーバーが SSSD 以外からアクセス可能かどうかをチェックすることでテストできます。以下の例では、TLS 接続で **test.example.com** への匿名バインドをテストします。

```
$ ldapsearch -x -ZZ -h test.example.com -b dc=example,dc=com
```

証明書信頼が適切に設定されていない場合、以下のエラーが出てテストは失敗します。

```
ldap_start_tls: Connect error (-11) additional info: TLS error -
8179:Unknown code ___f 13
```

証明書を信頼するには、以下を行います。

1. LDAP されていない証明書の署名に使われる証明書権限用の公開 CA 証明書のコピーを取得して、ローカルシステムに保存します。
2. ファイルシステム上の CA 証明書に向けた **sssd.conf** ファイルに以下の行を追加します。

   ```
   ldap_tls_cacert
   ```

3. LDAP サーバーが自己署名証明書を使用する場合は、**sssd.conf** ファイルから **ldap_tls_reqcert** の行を削除します。

   このパラメーターは CA 証明書が発行する証明書をいかなるものでも信頼するように SSSD に指示しますが、自己署名の CA 証明書ではセキュリティーリスクとなります。

問： 非標準ポートでの LDAP サーバーへの接続が失敗します。

答： enforcing モードで SELinux を実行する場合、クライアントの SELinux ポリシーは非標準ポートで LDAP サーバーに接続するように修正する必要があります。以下に例を挙げます。

```
# semanage port -a -t ldap_port_t -p tcp 1389
```

問： NSS がユーザー情報提供に失敗します。

答： これは通常、SSSD が NSS サービスに接続できないことを意味します。

NSS が稼働していることを確認します:

```
# service sssd status
```

NSS が稼働している場合、プロバイダーが **/etc/sssd/sssd.conf** ファイル内の **[nss]** セクションで正しく設定されていることを確認します。特に、***filter_users*** 属性と ***filter_groups*** 属性をチェックして下さい。

SSSD が使用するサービスのリスト内に NSS が含まれていることを確認します。

/etc/nsswitch.conf ファイル内の設定をチェックします。

問： NSS が、誤ったユーザー情報を返す。

答： ユーザー情報の検索で間違ったデータが返された場合には、別のドメインでユーザー名の競合がないことをチェックしてください。複数のドメインを使用している場合、**/etc/sssd/sssd.conf** ファイル内の ***use_fully_qualified_domains*** 属性を **true** に設定します。これが異なるドメインで同じ名前を持つ異なるユーザーを区別します。

問： ローカルの SSSD ユーザー用のパスワード設定で、パスワードの入力が 2 回要求される。

答： ローカルの SSSD ユーザーのパスワードの変更を試みると、パスワードを２回要求されることがあります。

```
[root@clientF11 tmp]# passwd user1000
Changing password for user user1000.
New password:
Retype new password:
New Password:
Reenter new Password:
passwd: all authentication tokens updated successfully.
```

これは、PAM 設定が正しくないために生じます。使用している **/etc/pam.d/system-auth** ファイル内で ***use_authtok*** オプションが正しく設定されていることを確認して下さい。

問： Identity Management (IPA) プロバイダーで sudo ルールを使おうとしていますが、sudo が適切に設定されているのに sudo ルールが見つかりません。

答： SSSD クライアントは正常に Identity Management サーバーを認証し、LDAP ディレクトリーで適切に sudo ルールを検索しています。しかし、ルールが存在しないことを示しています。例えばログでは以下のようになります。

```
(Thu Jun 21 10:37:47 2012) [sssd[be[ipa.test]]]
[sdap_sudo_load_sudoers_process] (0x0400): Receiving sudo rules with
base [ou=sudoers,dc=ipa,dc=test]
(Thu Jun 21 10:37:47 2012) [sssd[be[ipa.test]]]
[sdap_sudo_load_sudoers_done] (0x0400): Received 0 rules
(Thu Jun 21 10:37:47 2012) [sssd[be[ipa.test]]]
[sdap_sudo_purge_sudoers] (0x0400): Purging SUDOers cache of user's
[admin] rules
(Thu Jun 21 10:37:47 2012) [sssd[be[ipa.test]]]
[sysdb_sudo_purge_byfilter] (0x0400): No rules matched
(Thu Jun 21 10:37:47 2012) [sssd[be[ipa.test]]]
[sysdb_sudo_purge_bysudouser] (0x0400): No rules matched
(Thu Jun 21 10:37:47 2012) [sssd[be[ipa.test]]]
[sdap_sudo_load_sudoers_done] (0x0400): Sudoers is successfuly stored
in cache
(Thu Jun 21 10:37:47 2012) [sssd[be[ipa.test]]]
[be_sudo_handler_reply] (0x0200): SUDO Backend returned: (0, 0,
Success)
```

SSSD に Identity Management プロバイダーを使用する場合、SSSD は Kerberos/GSS-API を使って基礎的な LDAP ディレクトリーに接続しようとします。しかし、デフォルトではSSSD は匿名接続を使って LDAP に接続し、sudo ルールを取得します。つまり、デフォルト設定では、SSSD は Identity Management サーバーから sudo ルールを取得できないことになります。

Kerberos/GSS-API 接続での sudo ルール取得をサポートするには、**sssd.conf** の ID プロバイダー設定で GSS-API を認証メカニズムとして有効にします。以下に例を挙げます。

```
[domain/ipa.example.com]
id_provider = ipa
ipa_server = ipa.example.com
ldap_tls_cacert = /etc/ipa/ca.crt

sudo_provider = ldap
ldap_uri = ldap://ipa.example.com
ldap_sudo_search_base = ou=sudoers,dc=ipa,dc=example,dc=com
ldap_sasl_mech = GSSAPI
ldap_sasl_authid = host/hostname.ipa.example.com
ldap_sasl_realm = IPA.EXAMPLE.COM
krb5_server = ipa.example.com
```

問： 大きなディレクトリーでは パスワードルックアップ は、要求 1 件あたり数秒かかります。どうしたら改善できますか？

答： 最初のユーザールックアップは、LDAP サーバーへのコールです。インデックス化されていない検索は、よりリソース集約的なのでインデックス化されている検索よりも時間が長くかかります。これは、サーバーが一致を求めてディレクトリー内のすべてのエントリーをチェックするためです。ユーザールックアップを迅速化するには、SSSD が検索する属性をインデックス化します。

uid

uidNumber

gidNumber

gecos

問： Active Directory ID プロバイダーは sssd.conf ファイルで適切に設定されているのに、SSSD は接続に失敗し、GSS-API エラーがでます。

答： SSSD はホスト名を使ってのみ、Active Directory プロバイダーに接続できます。ホスト名が提供されないと、SSSD クライアントはホストへの IP アドレスが解決できず、認証に失敗します。

例えば以下の設定だと、

```
[domain/ADEXAMPLE]
debug_level = 0xFFF0
id_provider = ad
ad_server = 255.255.255.255
ad_domain = example.com
krb5_canonicalize = False
```

SSSD クライアントは以下の GSS-API エラーを返して、認証要求が失敗します。

```
(Fri Jul 27 18:27:44 2012) [sssd[be[ADTEST]]] [sasl_bind_send]
(0x0020): ldap_sasl_bind failed (-2)[Local error]
(Fri Jul 27 18:27:44 2012) [sssd[be[ADTEST]]] [sasl_bind_send]
(0x0080): Extended failure message: [SASL(-1): generic failure: GSSAPI
Error: Unspecified GSS failure.  Minor code may provide more
information (Cannot determine realm for numeric host address)]
```

このエラーを避けるには、***ad_server*** を Active Directory ホストの名前に設定します。

問： SSSD を中央認証に設定しましたが、Firefox や Adobe などいくつかのアプリケーションが起動しません。

答： 64 ビットシステム上でも 32 ビットのアプリケーションはパスワードや ID キャッシュへのアクセスに 32 ビットバージョンの SSSD を必要とします。32 ビットバージョンの SSSD が利用できない場合でも、システムは SSSD キャッシュを使うように設定されており、したがって 32 ビットのアプリケーションは起動に失敗します。

例えば、Firefox は権限拒否のエラーで失敗します。

```
Failed to contact configuration server. See
http://www.gnome.org/projects/gconf/
for information. (Details -  1: IOR file '/tmp/gconfd-
somebody/lock/ior'
not opened successfully, no gconfd located: Permission denied 2: IOR
file '/tmp/gconfd-somebody/lock/ior' not opened successfully, no
gconfd
located: Permission denied)
```

Adobe Reader の場合、以下のエラーは現行のシステムはユーザーが認識されていないことを示しています。

```
[jsmith@server ~]$ acroread
(acroread:12739): GLib-WARNING **: getpwuid_r(): failed due to unknown
user id (366)
```

他のアプリケーションでも、同様のユーザーもしくは権限エラーが表示される可能性があります。

問： SSSDは、削除した自動マウントの場所を示しています。

答： 自動マウントのロケーションの SSSD キャッシュは、ロケーションがその後で変更されたり削除されたりしても、消えずに残ります。SSSD の autofs 情報を更新するには、以下を行います。

1. 「SSSD キャッシュの削除」 の説明にあるように、autofs キャッシュを削除します。
2. 「SSSD の起動と停止」 にあるように、SSSD を再起動します。

---

[4] ほとんどのサービスでは、そのサービスで指定サーバーが設定されていなければ、ID プロバイダーサーバーをデフォルトとしています。

第12章 OPENSSH

SSH (Secure Shell: セキュアシェル) は、クライアント/サーバーアーキテクチャーを使用する 2 つのシステム間でのセキュアな通信を容易にし、ユーザーがリモートでサーバーホストシステムにログインできるようにするプロトコルです。**FTP** や **Telnet** などの他のリモート通信プロトコルとは異なり、SSH はログインセッションを暗号化するため、侵入者が暗号化されていないパスワードを入手するための接続が難しくなります。

ssh プログラムは、**telnet** や **rsh** などのリモートホストへログインするために使用される、旧式でセキュリティの低いターミナルアプリケーションと置き換わるように設計されています。また、**scp** と呼ばれる関連プログラムが、ホスト間でファイルをコピーするために設計された **rcp** のような旧式プログラムと置き換わります。こうした旧式のアプリケーションは、クライアントとサーバー間で送信されるパスワードを暗号化しないため、可能な限り使用を避けるようにして下さい。リモートシステムへのログインにセキュアな方法を使用することで、クライアントシステム及びリモートホストの両方に対するリスクが軽減されます。

Red Hat Enterprise Linux には、全般的な OpenSSH パッケージ (**openssh**) と共に、OpenSSH サーバー (**openssh-server**) 及びクライアント (**openssh-clients**) パッケージが含まれています。OpenSSH パッケージには、重要暗号化ライブラリをインストールして OpenSSH の暗号化通信を可能にする OpenSSL パッケージ (**openssl**) が必要な点に注意して下さい。

12.1. SSH プロトコル

12.1.1. SSH を使用する理由

潜在的な侵入者は、ネットワークトラフィックの中断/傍受/経路変更を可能にする様々なツールを自由に駆使して、システムに侵入します。一般的には、これらの脅威は以下のとおり分類できます。

2 システム間の通信の傍受

攻撃者は、ネットワーク上で通信を行う二者の間のどこかに潜み、両者間で渡される情報をコピーしている可能性があります。攻撃者は情報を傍受して保持する、または情報を改ざんして対象となる受信者に送信する場合があります。

このような攻撃は、通常 *パケットスニファ* を使用して行われます。パケットスニファは、ネットワークを通過するパケットをキャプチャしてその内容を分析するかなり一般的なネットワークユーティリティです。

特定のホストの偽装

攻撃者のシステムは、送信の対象となる受信者を装うように設定されます。この戦略が成功すると、ユーザーのシステムは不正なホストと通信していることに気がつかないままとなります。

この攻撃は、*DNS ポイズニング* として知られる手法か *IP スプーフィング* と呼ばれる手法を用いて実行されます。前者の場合、侵入者はクラックされた DNS サーバーを使用して、クライアントシステムを不当に複製されたホストへポイントします。後者の場合は、侵入者は信頼されたホストから送信されたように見せかけた偽装ネットワークパケットを送信します。

いずれの手法でも、潜在的な機密情報を傍受することが可能です。その傍受が悪意のある理由で行われる場合には、悲惨な結果をもたらしかねません。リモートシェルログインとファイルコピー用に SSH を使用すると、こうしたセキュリティ脅威は大幅に軽減される場合があります。これは、SSH クライアントとサーバーがデジタル署名を使用してそれぞれの ID を確認するためです。さらに、クライアントシステムとサーバーシステム間の全通信は暗号化されています。各パケットはローカルシステムとリモートシステムのみに知られている鍵を使用して暗号化されるため、通信のいずれか一方の ID をスプーフィングする試みは成功しません。

12.1.2. 主要な機能

SSH プロトコルは、以下のような保護手段を提供します。

対象のサーバーとして装うことができるものはありません

初回接続後に、クライアントは以前に接続したサーバーと同じサーバーに接続していることを確認できます。

認証情報をキャプチャできるものはありません

クライアントは、強力な 128 ビット暗号化を使用してサーバーへ認証情報を送信します。

通信を傍受できるものはありません

セッション中に送受信された全データは、128 ビット暗号化を使用して転送されるため、傍受された送信データの復号化と読み取りは非常に難しくなります。

さらに、SSH プロトコルは以下のようなオプションも提供します。

ネットワーク上でグラフィカルアプリケーションを使用するセキュアな手段を提供

クライアントは、*X11 転送* と呼ばれる手法を使用して、サーバーから *X11* (*X Window System*) アプリケーションを転送することが可能です。

セキュアでないプロトコルをセキュアにする手段を提供

SSH プロトコルは、送受信するすべてを暗号化します。SSH サーバーは、*ポート転送* と呼ばれる手法を使用して、POP のようなセキュアでないプロトコルをセキュアにするための経路となり、システムとデータ全体のセキュリティを強化することができます。

セキュアなチャンネルの作成時に使用可能

OpenSSH サーバーとクライアントは、サーバーとクライアントマシン間のトラフィックに対し仮想プライベートネットワークに似たトンネルを作成するよう設定できます。

Kerberos 認証に対応

OpenSSH のサーバーとクライアントは、Kerberos ネットワーク認証プロトコルの GSSAPI (Generic Security Services Application Program Interface: 汎用セキュリティサービス API) 実装を使用して認証を行うよう設定できます。

12.1.3. プロトコルのバージョン

現在、SSH には バージョン 1 と新しいバージョン 2 の 2 種類があります。Red Hat Enterprise Linux の OpenSSH スイートでは、SSH バージョン 2 を使用します。このバージョンは、バージョン 1 での既知のエクスプロイトに対して脆弱性のない拡張された鍵交換アルゴリズムを使用します。ただし、互換性上の理由から OpenSSH スイートはバージョン 1 の接続にも対応しています。

重要

接続のセキュリティを最大限に確保するために、可能な限り SSH バージョン 2 との互換性があるサーバーとクライアントのみを使用することが推奨されます。

12.1.4. SSH 接続のイベントシーケンス

以下にあげる一連のイベントは、2 つのホスト間で行われる SSH 通信の整合性を保護するために役立ちます。

1. 暗号化ハンドシェイクが行われ、クライアントは正しいサーバーと通信していることを確認できます。
2. クライアントとリモートホスト間の接続のトランスポート層は、対称暗号方式を使用して暗号化されます。
3. クライアントはサーバーに対して自己認証します。
4. リモートクライアントは、暗号化された接続でリモートホストと対話します。

12.1.4.1. トランスポート層

トランスポート層の主な役割は、認証時及びその後の通信中における 2 つのホスト間の安全でセキュアな通信を容易にすることです。トランスポート層は、データの暗号化と復号化を処理し、データパケットの送受信時にその整合性を保護することでその役割を果たします。また、トランスポート層は、情報を圧縮して転送を高速化します。

SSH クライアントがサーバーにコンタクトすると、基本情報が交換されるため両システムはトランスポート層を適正に構築することができます。以下は、こうした基本情報の交換中に発生するステップです。

- 鍵が交換されます
- 公開鍵暗号化アルゴリズムが決定されます
- 対称暗号化アルゴリズムが決定されます
- メッセージ認証アルゴリズムが決定されます
- ハッシュアルゴリズムが決定されます

鍵交換の間、サーバーは一意の *ホスト鍵* を用いて、クライアントに対して自己識別を行います。クライアントがこの特定のサーバーと過去に通信したことがなければ、サーバーのホスト鍵はクライアントには未知であり、接続は成立しません。OpenSSH はこの問題に対処するためにサーバーのホスト鍵を承認します。これは、ユーザーが通知を受けて新規のホスト鍵を受け取り検証した後に行われます。それ以降の接続では、サーバーのホスト鍵は、クライアント上に保存されているバージョンと照合され、クライアントが本当に目的のサーバーと通信していることを確信できます。今後このホスト鍵が一致しなくなった場合には、ユーザーは接続前にクライアントの保存してあるバージョンを削除する必要があります。

警告

ローカルシステムは、対象サーバーと攻撃者が設定した偽サーバーとの違いを認識しないため、攻撃者は初回コンタクト中に SSH サーバーをマスカレードすることが可能です。この問題を防ぐために、初回接続の前かホスト鍵の不一致が発生した場合には、サーバー管理者へ連絡して新しい SSH サーバーの整合性を確認して下さい。

SSH は、ほとんどすべての公開鍵アルゴリズムまたはエンコード形式に対応するように設計されています。初回の鍵交換で、交換に使用されるハッシュ値と共有秘密値が作成された後、2 つのシステムは新しい鍵とアルゴリズムの計算を直ちに開始して、認証と今後この接続で送信されるデータを保護します。

所定の鍵とアルゴリズムを使用して一定量のデータ (正確な量は SSH 実装により異なる) が送信された後に、もう 1 回鍵交換が行われてハッシュ値と新しい共有秘密値の別のセットが生成されます。攻撃者がハッシュ値と共有秘密値を判別できたとしても、その情報が役立つのは限られた時間だけです。

12.1.4.2. 認証

トランスポート層が 2 つのシステム間で情報を渡すためのセキュアなトンネルを構築すると 、サーバーは秘密鍵でエンコードされた署名の使用やパスワードの入力などサポートされている別の認証方法をクライアントに伝えます。次に、クライアントはサポートされているどちらかの方法を使ってサーバーに対して自己認証を試みます。

SSH サーバーとクライアントは、異なるタイプの認証を採用できるように設定可能なため、双方の制御が最適化されます。サーバーはそのセキュリティモデルに基づき、サポートする暗号化方法を決定することができ、クライアントは利用可能なオプションの中から試行する認証方法の順番を選択できます。

12.1.4.3. チャンネル

SSH トランスポート層での認証に成功した後は、*多重化*[5] と呼ばれる手法により複数のチャンネルが開かれます。これらの各チャンネルは、異なるターミナルセッションと転送された X11 セッションの通信を処理します。

クライアントとサーバーとも、新しいチャンネルを作成できます。その後、各チャンネルに別々の番号が接続の両端に割り当てられます。クライアントが新しいチャンネルを開こうとする時には、クライアントは要求と共にチャンネル番号を送信します。この情報はサーバーにより保存され、そのチャンネルに通信を移動するために使用されます。この目的は、異なるタイプのセッションが相互に影響しないように、あるセッションの終了時にそのチャンネルが SSH による一次接続を停止せずに閉じることができるようにするためです。

また、チャンネルは *フロー制御* もサポートしているため規則的な方法でデータを送受信することができます。この方法では、チャンネルが開いているというメッセージをクライアントが受信するまで、データはチャンネル上で送信されません。

クライアントが要求するサービスのタイプとユーザーがネットワークに接続される方法に応じて、クライアントとサーバーは、各チャンネルの特性を自動的にネゴシエートします。これにより、プロトコルの基本インフラストラクチャーを変更しなくても、異なるタイプのリモート接続を非常に柔軟に処理することができます。

12.2. OPENSSH の設定

12.2.1. 設定ファイル

設定ファイルには、クライアントプログラム用 (**ssh**、**scp** 及び **sftp**) とサーバー用 (**sshd** デーモン) の異なる 2 つのセットがあります。

システム全体の SSH 設定情報は、表12.1「システム全体の設定ファイル」 の説明のとおり **/etc/ssh/** ディレクトリ内に格納されています。ユーザー固有の SSH 設定情報は、ユーザーのホームディレクトリ内の **~/.ssh/** に格納されています。詳しくは、表12.2「ユーザー固有の設定ファイル」に記載しています。

表12.1 システム全体の設定ファイル

ファイル	詳細
/etc/ssh/moduli	セキュアなトランスポート層を構築するために非常に重要となる、Diffie-Hellman 鍵交換に使用される Diffie-Hellman グループが格納されています。SSH セッションの始めで鍵が交換される時、共有秘密値が作成されますが、どちらか一方の当事者だけでは決定できません。この値はホスト認証を行う場合に使用されます。
/etc/ssh/ssh_config	デフォルトの SSH クライアント設定ファイルです。**~/.ssh/config** が存在する場合には、これにより上書きされる点に注意して下さい。
/etc/ssh/sshd_config	**sshd** デーモン用の設定ファイルです。
/etc/ssh/ssh_host_dsa_key	**sshd** デーモンにより使用される DSA 秘密鍵です。
/etc/ssh/ssh_host_dsa_key.pub	**sshd** デーモンにより使用される DSA 公開鍵です。
/etc/ssh/ssh_host_key	**sshd** デーモンにより使用される SSH プロトコルのバージョン 1 用の RSA 秘密鍵です。
/etc/ssh/ssh_host_key.pub	**sshd** デーモンにより使用される SSH プロトコルのバージョン 1 用の RSA 公開鍵です。
/etc/ssh/ssh_host_rsa_key	**sshd** デーモンにより使用される SSH プロトコルのバージョン 2 用の RSA 秘密鍵です。
/etc/ssh/ssh_host_rsa_key.pub	**sshd** デーモンにより使用される SSH プロトコルのバージョン 2 用の RSA 公開鍵です。

表12.2 ユーザー固有の設定ファイル

ファイル	詳細
~/.ssh/authorized_keys	サーバー用の認証済み公開鍵の一覧が含まれています。クライアントがサーバーに接続する時、サーバーはこのファイル内に格納されている署名済み公開鍵を確認してクライアントを認証します。
~/.ssh/id_dsa	ユーザーの DSA 秘密鍵が含まれています。
~/.ssh/id_dsa.pub	ユーザーの DSA 公開鍵です。
~/.ssh/id_rsa	**ssh** により使用される SSH プロトコルのバージョン 2 用の RSA 秘密鍵です。

ファイル	詳細
~/.ssh/id_rsa.pub	**ssh** により使用される SSH プロトコルのバージョン 2 用の RSA 公開鍵です。
~/.ssh/identity	**ssh** により使用される SSH プロトコルのバージョン 1 用の RSA 秘密鍵です。
~/.ssh/identity.pub	**ssh** により使用される SSH プロトコルのバージョン 1 用の RSA 公開鍵です。
~/.ssh/known_hosts	ユーザーによりアクセスされる SSH サーバーの DSA ホスト鍵が格納されています。このファイルは SSH クライアントが正しい SSH サーバーに接続していることを確認するために非常に重要です。

SSH 設定ファイルに使用可能な各種ディレクティブについての情報は、**ssh_config**(5) および **sshd_config**(5) の man ページを参照してください。

12.2.2. OpenSSH サーバーの起動

OpenSSH サーバーを実行するには、openssh-server および openssh パッケージがインストールされている必要があります (Red Hat Enterprise Linux 6 に新規パッケージをインストールする方法については 「パッケージのインストール」 を参照してください)。

sshd デーモンを起動するには、シェルプロンプトで以下のコマンドを入力します:

```
~]# service sshd start
```

実行中の **sshd** デーモンを停止するには、以下のコマンドを使用します:

```
~]# service sshd stop
```

ブート時にデーモンが自動的に起動するよう設定するには、以下のコマンドを入力します:

```
~]# chkconfig sshd on
```

これにより、全ランレベルでサービスが有効になります。さらなる設定オプションについては、10 章 *サービスとデーモン* に記載されているサービスの管理方法に関する詳しい情報を参照して下さい。

システムを再インストールすると、新しい識別鍵のセットが作成される点に注意して下さい。そのため、再インストールの前にいずれかの OpenSSH ツールを使用してシステムに接続したことがあるクライアントには、以下のようなメッセージが表示されます:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle
attack)!
It is also possible that the RSA host key has just been changed.
```

これを防ぐには、**/etc/ssh/** ディレクトリから関連ファイルをバックアップしておくと (全一覧は 表 12.1「システム全体の設定ファイル」 を参照)、システムを再インストールする時にはいつでも復元できます。

12.2.3. リモート接続に必要な SSH

SSH を真に有効にするためには、セキュリティ保護されていない接続プロトコルは使用しないことをお勧めします。そうしないと、ユーザーのパスワードは SSH を使用した 1 回のセッションでは保護されても、その後に Telnet を使用してログインした時には結局傍受されてしまうためです。無効にするサービスは、**telnet**、**rsh**、**rlogin**、**vsftpd** などがあります。

これらのサービスを無効にするには、シェルプロンプトで次のコマンドを入力します:

```
~]# chkconfig telnet off
~]# chkconfig rsh off
~]# chkconfig rlogin off
~]# chkconfig vsftpd off
```

ランレベルとサービス設定全般に関する詳細は 10章 *サービスとデーモン* を参照して下さい。

12.2.4. 鍵ベース認証の使用

システムのセキュリティをさらに強化するには、通常のパスワード認証を無効にすることで鍵ベース認証を強制することができます。これを行うには、**/etc/ssh/sshd_config** の設定ファイルを **vi** や **nano** などのテキストエディターで開き、**PasswordAuthentication** オプションを以下のように変更します:

```
PasswordAuthentication no
```

ssh、**scp** または **sftp** を使用してクライアントマシンからサーバーに接続できるようにするには、以下の手順に従って認証鍵ペアを生成します。鍵はユーザーごとに別々に生成する必要がある点に注意して下さい。

Red Hat Enterprise Linux 6 は、デフォルトでは SSH プロトコル 2 と RSA 鍵を使用します (詳細は 「プロトコルのバージョン」 を参照)。

重要

以下の手順を root で完了すると、その鍵を使用できるのは root のみとなってしまうためです。

注記

システムを再インストールする場合に、以前に生成された鍵ペアを保持したいときは、**~/.ssh/** ディレクトリをバックアップして下さい。再インストール後に、このディレクトリをホームディレクトリにコピーします。この手順は、root を含むシステム上の全ユーザーを対象に実行できます。

12.2.4.1. 鍵ペアの生成

以下の手順に従い、SSH プロトコルのバージョン 2 用の RSA 鍵ペアを生成します。

1. RSA 鍵ペアを生成するには、シェルプロンプトで以下を入力します:

```
~]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/john/.ssh/id_rsa):
```

2. **Enter** キーを押して、新規作成された鍵用のデフォルトの場所 (**~/.ssh/id_rsa**) を確認します。

3. パスフレーズを入力します。プロンプトが表示されたら再入力して確認します。セキュリティ上の理由により、アカウントにログインする時と同じパスワードは使用しないで下さい。

   この後に、以下のようなメッセージが表示されます:

```
Your identification has been saved in /home/john/.ssh/id_rsa.
Your public key has been saved in /home/john/.ssh/id_rsa.pub.
The key fingerprint is:
e7:97:c7:e2:0e:f9:0e:fc:c4:d7:cb:e5:31:11:92:14
john@penguin.example.com
The key's randomart image is:
+--[ RSA 2048]----+
|              E.  |
|             . .  |
|              o . |
|               . .|
|        S .    . |
|         + o o ..|
|          * * +oo|
|           O +..=|
|           o*  o.|
+-----------------+
```

4. **~/.ssh/** ディレクトリのパーミッションを変更します:

```
~]$ chmod 700 ~/.ssh
```

5. **~/.ssh/id_rsa.pub** の内容を接続先のマシンの **~/.ssh/authorized_keys** にコピーします。ファイルが既に存在している場合は、そのファイルの最後に追加して下さい。

6. 次のコマンドを使用して **~/.ssh/authorized_keys** ファイルのパーミッションを変更します:

```
~]$ chmod 600 ~/.ssh/authorized_keys
```

以下の手順に従い、SSH プロトコルのバージョン 2 用の DSA 鍵ペアを生成します。

1. DSA 鍵ペアを生成するには、シェルプロンプトで以下を入力します :

```
~]$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/john/.ssh/id_dsa):
```

2. **Enter** キーを押して、新規作成された鍵用のデフォルトの場所 (**~/.ssh/id_dsa**) を確認します。

3. パスフレーズを入力します。プロンプトが表示されたら再入力して確認します。セキュリティ上の理由により、アカウントにログインする時と同じパスワードは使用しないで下さい。

   この後に、以下のようなメッセージが表示されます:

```
Your identification has been saved in /home/john/.ssh/id_dsa.
Your public key has been saved in /home/john/.ssh/id_dsa.pub.
The key fingerprint is:
81:a1:91:a8:9f:e8:c5:66:0d:54:f5:90:cc:bc:cc:27
john@penguin.example.com
The key's randomart image is:
+--[ DSA 1024]----+
|   .oo*o.        |
|  ...o Bo        |
| .. . + o.       |
|.  .   E o       |
| o..o   S        |
|. o= .           |
|. +              |
| .               |
|                 |
+-----------------+
```

4. **~/.ssh/** ディレクトリのパーミッションを変更します:

```
~]$ chmod 700 ~/.ssh
```

5. **~/.ssh/id_dsa.pub** の内容を接続先のマシンの **~/.ssh/authorized_keys** にコピーします。ファイルが既に存在している場合は、そのファイルの最後に追加して下さい。

6. 次のコマンドを使用して **~/.ssh/authorized_keys** ファイルのパーミッションを変更します:

```
~]$ chmod 600 ~/.ssh/authorized_keys
```

以下の手順に従い、SSH プロトコルのバージョン 1 用の RSA 鍵ペアを生成します。

1. RSA 鍵ペアを生成するには、シェルプロンプトで以下を入力します:

```
~]$ ssh-keygen -t rsa1
Generating public/private rsa1 key pair.
Enter file in which to save the key (/home/john/.ssh/identity):
```

2. **Enter** キーを押して、新規作成された鍵用のデフォルトの場所 (**~/.ssh/identity**) を確認します。

3. パスフレーズを入力します。プロンプトが表示されたら再入力して確認します。セキュリティ上の理由により、アカウントにログインする時と同じパスワードは使用しないで下さい。

   この後に、以下のようなメッセージが表示されます:

```
Your identification has been saved in /home/john/.ssh/identity.
Your public key has been saved in /home/john/.ssh/identity.pub.
The key fingerprint is:
```

```
cb:f6:d5:cb:6e:5f:2b:28:ac:17:0c:e4:62:e4:6f:59
john@penguin.example.com
The key's randomart image is:
+--[RSA1 2048]----+
|                 |
|     . .         |
|    o o          |
|     + o E       |
|    . o S        |
|       = +   .   |
|      . = . o . .|
|       . = o o..o|
|       .o o  o=o.|
+-----------------+
```

4. **~/.ssh/** ディレクトリのパーミッションを変更します:

```
~]$ chmod 700 ~/.ssh
```

5. **~/.ssh/identity.pub** の内容を接続先のマシンの **~/.ssh/authorized_keys** にコピーします。ファイルが既に存在している場合は、そのファイルの最後に追加して下さい。

6. 次のコマンドを使用して **~/.ssh/authorized_keys** ファイルのパーミッションを変更します:

```
~]$ chmod 600 ~/.ssh/authorized_keys
```

システムにパスフレーズを記憶させる設定方法については 「ssh-agent の設定」 を参照して下さい。

重要

秘密鍵は、個人使用向けのものであり、他人には決して教えないことが重要です。

12.2.4.2. ssh-agent の設定

ssh-agent 認証エージェントを使用すると、パスフレーズを保存することができるためリモートマシンとの接続を開始する度にパスフレーズを入力する必要がなくなります。GNOME を実行している場合は、ログイン時には常にパスフレーズを求めるプロンプトを表示して、セッション中を通してそのパスフレーズを記憶させておくように設定できます。それ以外の方法として、特定のシェルプロンプト用にパスフレーズを保存しておくことも可能です。

以下の手順に従い、GNOME セッション中にパスフレーズを保存します。

1. openssh-askpass パッケージがインストールされていることを確認します。インストールされていない場合は、「パッケージのインストール」 の Red Hat Enterprise Linux に新規パッケージをインストールする方法を参照して下さい。

2. パネルから **システム** → **設定** → **自動起動するアプリ** の順に選択します。**自動起動するアプリの設定** が起動し、利用可能な自動起動するプログラムの一覧が表示されたタブがデフォルトで表示されます。

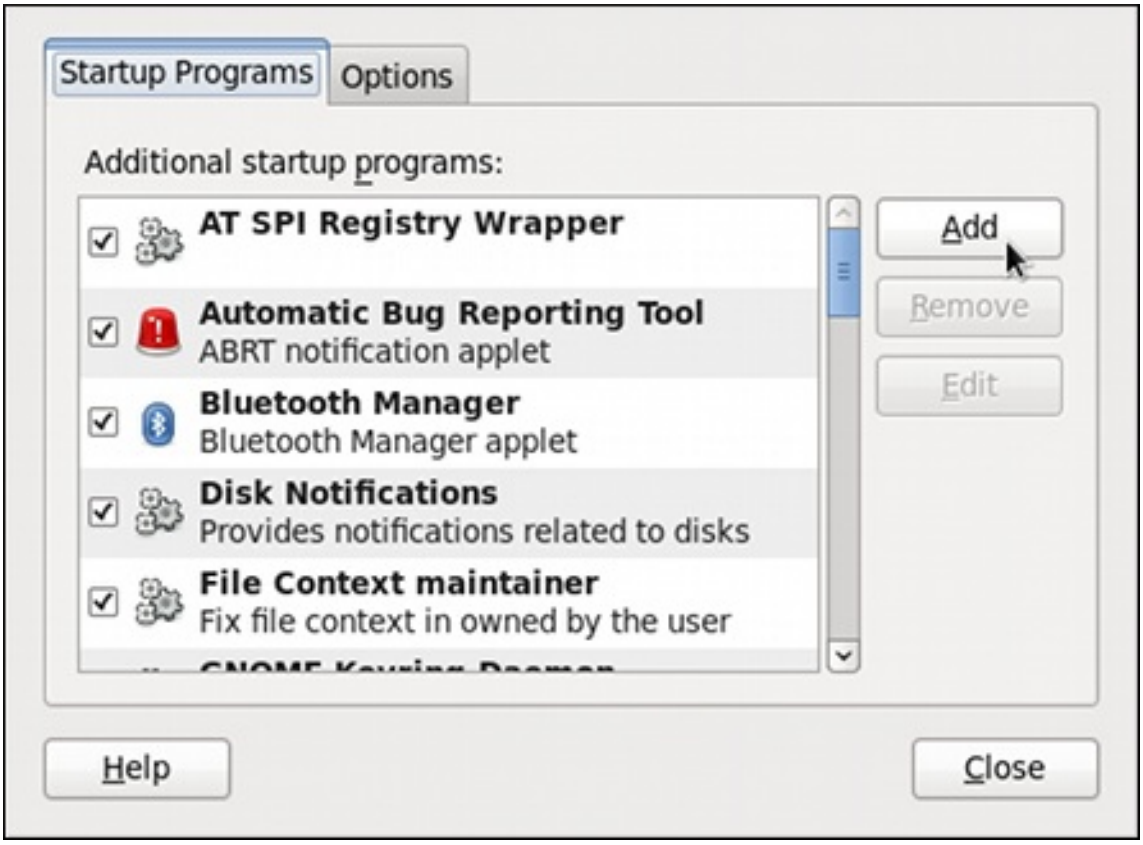


図12.1 自動起動するアプリの設定

3. 右側の **追加** ボタンをクリックして、**コマンド** フィールドに **/usr/bin/ssh-add** と入力します。

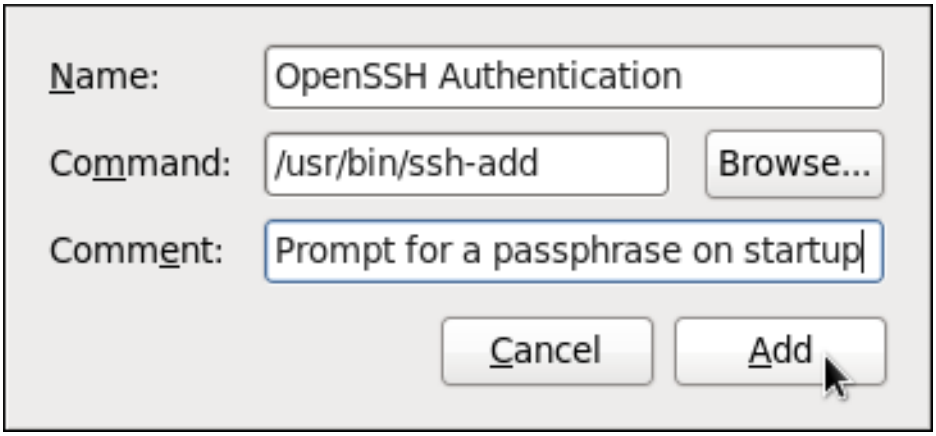


図12.2 新規アプリケーションの追加

4. **追加** をクリックした後に、新しく追加した項目の横のチェックボックスにチェックマークが付いていることを確認してください。

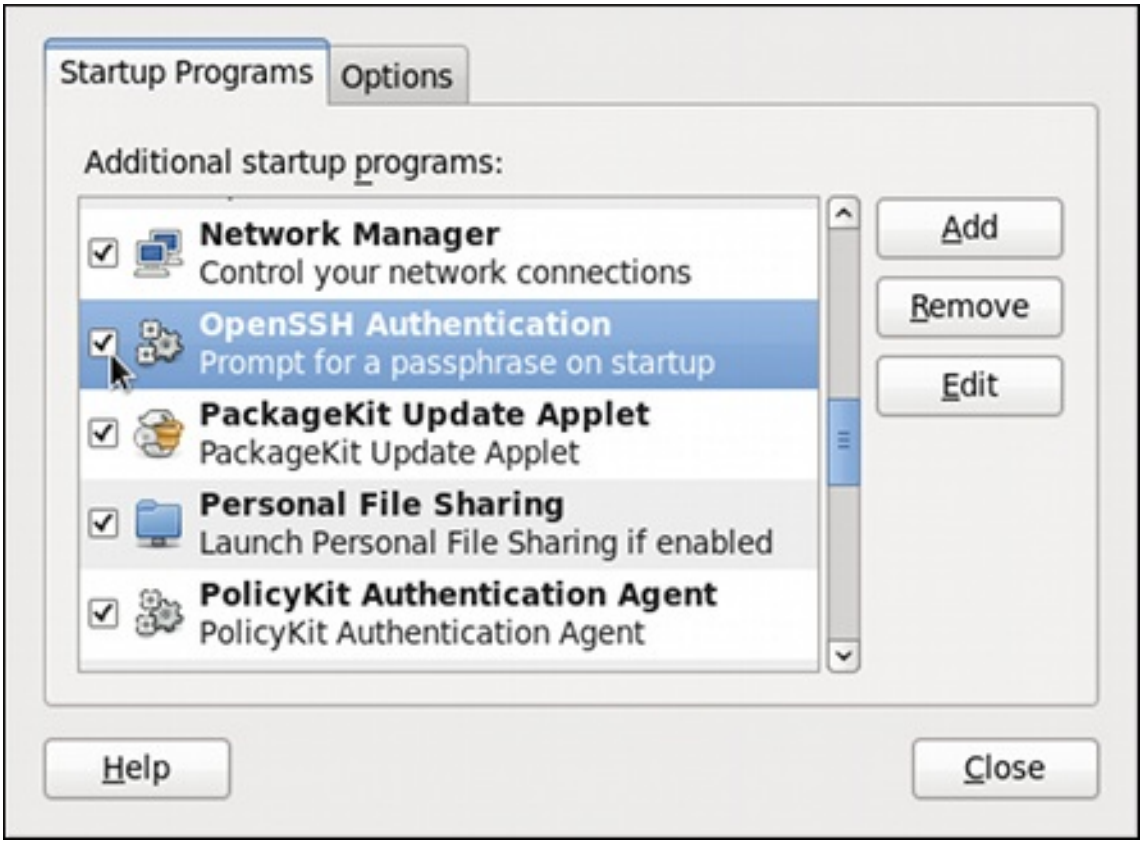


図12.3 アプリケーションの有効化

5. 一度ログアウトしてから再度ログインします。パスフレーズの入力を求めるダイアログボックスが表示されます。これ以降は、**ssh**、**scp** または **sftp** によるパスワードの入力を要求されることはありません。

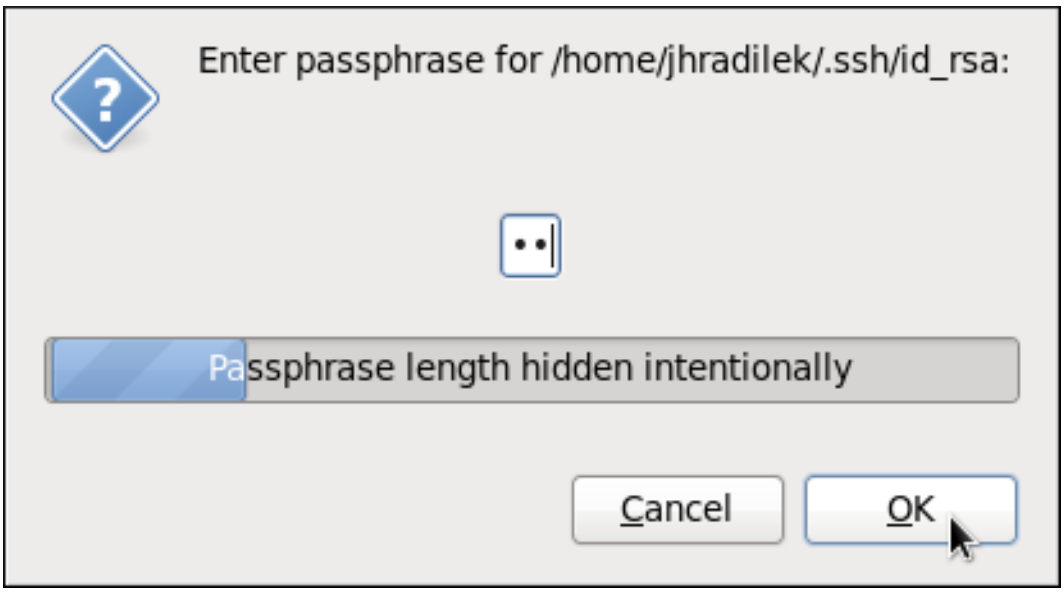


図12.4 パスフレーズの入力

特定のシェルプロンプト用のパスフレーズを保存するには、以下のコマンドを使用します:

```
~]$ ssh-add
Enter passphrase for /home/john/.ssh/id_rsa:
```

ログアウト時には、パスフレーズは記憶されない点に注意して下さい。仮想コンソールまたはターミナルウィンドウにログインする度にコマンドを実行する必要があります。

12.3. OPENSSH クライアント

クライアントマシンから OpenSSH サーバーに接続するには、openssh-clients および openssh パッケージがインストールされている必要があります (Red Hat Enterprise Linux に新規パッケージをインストールする方法については 「パッケージのインストール」 を参照して下さい)。

12.3.1. ssh ユーティリティの使用

ssh ユーティリティを使用すると、リモートマシンにログインしてそのマシン上でコマンドを実行することできます。これは、**rlogin**、**rsh** および **telnet** プログラムに代わるセキュアな手段です。

telnet と同様に、リモートマシンにログインするには、以下のコマンドを入力します。

```
ssh hostname
```

例えば、**penguin.example.com** という名前のリモートマシンにログインするには、シェルプロンプトで以下を入力します。

```
~]$ ssh penguin.example.com
```

これによって、ローカルマシンで使用しているのと同じユーザー名でログインします。別のユーザー名を指定したい場合には、以下の形式のコマンドを使用してください。

```
ssh username@hostname
```

例えば、**john** として **penguin.example.com** にログインするには、以下のように入力します。

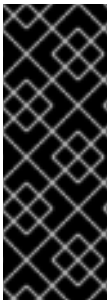
```
~]$ ssh john@penguin.example.com
```

初回接続時には、以下のようなメッセージが表示されます:

```
The authenticity of host 'penguin.example.com' can't be established.
RSA key fingerprint is 94:68:3a:3a:bc:f3:9a:9b:01:5d:b3:07:38:e2:11:0c.
Are you sure you want to continue connecting (yes/no)?
```

yes と入力して確定します。サーバーが既知ホストの一覧に追加されたことを知らせるメッセージとパスワードの入力を求めるプロンプトが以下のように表示されます:

```
Warning: Permanently added 'penguin.example.com' (RSA) to the list of
known hosts.
john@penguin.example.com's password:
```



重要

SSH サーバーのホスト鍵が変更された場合、クライアントはサーバーのホスト鍵が **~/.ssh/known_hosts** ファイルから削除されるまで接続を続行できないことをユーザーに知らせます。サーバーのホスト鍵を削除するには、そのファイルをテキストエディターで開いて、対象のリモートマシン名が冒頭に記載されている行を削除します。ただし、これを実行する前に、SSH サーバーのシステム管理者に連絡してサーバーが侵害を受けていないことを確認して下さい。

パスワードを入力すると、リモートマシン用のシェルプロンプトが表示されます。

別の方法として、シェルプロンプトにログインせずに、**ssh** プログラムを使用してリモートマシン上でコマンドを実行することができます。

```
ssh [username@]hostname command
```

例えば、**/etc/redhat-release** ファイルは Red Hat Enterprise Linux のバージョンに関する情報を提供します。**penguin.example.com** でこのファイルの内容を表示するには、以下を入力します。

```
~]$ ssh john@penguin.example.com cat /etc/redhat-release
john@penguin.example.com's password:
Red Hat Enterprise Linux Server release 6.2 (Santiago)
```

正しいパスワードを入力すると、ユーザー名が表示され、ローカルのシェルプロンプトへ戻ります。

12.3.2. scp ユーティリティの使用

scp を使用すると、暗号化されたセキュアな接続でマシン間のファイル転送を行うことができます。設計に関しては、**rcp** と非常に似ています。

ローカルファイルをリモートシステムへ転送するには、以下の形式でコマンドを使用します:

```
scp localfile username@hostname:remotefile
```

例えば、**taglist.vim** を **penguin.example.com** という名前のリモートマシンに転送したい場合は、シェルプロンプトで以下のように入力します:

```
~]$ scp taglist.vim john@penguin.example.com:.vim/plugin/taglist.vim
john@penguin.example.com's password:
taglist.vim                                   100%  144KB 144.5KB/s
00:00
```

一度に複数のファイルを指定することも可能です。**.vim/plugin/** の内容を **penguin.example.com** のリモートマシン上の同じディレクトリに転送するには、以下のコマンドを入力します:

```
~]$ scp .vim/plugin/* john@penguin.example.com:.vim/plugin/
john@penguin.example.com's password:
closetag.vim                                  100%   13KB  12.6KB/s
00:00
snippetsEmu.vim                               100%   33KB  33.1KB/s
00:00
taglist.vim                                   100%  144KB 144.5KB/s
00:00
```

リモートファイルをローカルシステムへ転送するには、以下の構文を使用します:

```
scp username@hostname:remotefile localfile
```

例えば **.vimrc** 設定ファイルをリモートマシンからダウンロードするには、以下のように入力します:

```
~]$ scp john@penguin.example.com:.vimrc .vimrc
john@penguin.example.com's password:
.vimrc                                        100% 2233     2.2KB/s
00:00
```

12.3.3. sftp ユーティリティの使用

sftp ユーティリティを使用すると、セキュアでインタラクティブな FTP セッションを開始することができます。設計に関しては、暗号化されたセキュアな接続を使用する以外は **ftp** と似ています。

リモートシステムに接続するには、以下の形式でコマンドを使用します:

```
sftp username@hostname
```

例えば **penguin.example.com** という名前のリモートマシンに **john** というユーザー名でログインするには、以下のように入力します:

```
~]$ sftp john@penguin.example.com
john@penguin.example.com's password:
Connected to penguin.example.com.
sftp>
```

正しいパスワードを入力すると、プロンプトが表示されます。**sftp** ユーティリティは、**ftp** で使用されるコマンドセットと似たものを使用します (表12.3「利用可能な sftp コマンドの抜粋」 を参照)。

表12.3 利用可能な sftp コマンドの抜粋

コマンド	詳細
ls [*directory*]	リモート *directory* の内容を一覧表示します。何も指定されていない場合は、デフォルトで現在の作業ディレクトリが使用されます。
cd *directory*	リモートの作業ディレクトリを *directory* に変更します。
mkdir *directory*	リモートの *directory* を作成します。
rmdir *path*	リモートの *directory* を削除します。
put *localfile* [*remotefile*]	*localfile* をリモートマシンに転送します。
get *remotefile* [*localfile*]	*remotefile* をリモートマシンから転送します。

利用可能なコマンドの全一覧は、**sftp**(1) の man ページを参照してください。

12.4. セキュアシェルの高機能

セキュアなコマンドラインインターフェースは、数多くある SSH の用途の中でも初歩的なものに過ぎません。十分な帯域幅があれば、X11 セッションは SSH チャンネル上で送信できます。あるいは、TCP/IP 転送を使用することで、以前はセキュリティ保護されていなかったシステム間のポート接続を特定の SSH チャンネルにマッピングすることができます。

12.4.1. X11 転送

SSH 接続上で X11 セッションを開始するには、以下の形式でコマンドを使用します:

```
ssh -Y username@hostname
```

例えば **penguin.example.com** という名前のリモートマシンに **john** というユーザー名でログインするには、以下のように入力します:

```
~]$ ssh -Y john@penguin.example.com
john@penguin.example.com's password:
```

セキュアなシェルプロンプトから X プログラムが実行されると、SSH クライアントとサーバーは新しいセキュアなチャンネルを作成し、X プログラムデータはそのチャンネル上で透過的にクライアントマシンに送信されます。

X11 転送は非常に有用です。例えば、X11 転送を使用すると、**プリンター設定** ユーティリティのセキュアかつインタラクティブなセッションを作成できます。これを行うには、**ssh** を使用してサーバーに接続し、以下のコマンドを入力します:

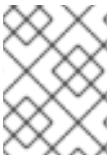
```
~]$ system-config-printer &
```

プリンター設定ツール が表示され、リモートユーザーはリモートシステム上で印刷の設定を安全に行うことができます。

12.4.2. ポート転送

SSH は、ポート転送によりセキュリティ保護されていない **TCP/IP** プロトコルをセキュアにすることができます。この手法を使用する場合、SSH サーバーは SSH クライアントをつなぐ暗号化された経路となります。

ポート転送は、クライアント上のローカルポートをサーバー上のリモートポートにマッピングすることで機能します。SSH ではサーバーの任意のポートをクライアント上の任意のポートにマッピングすることが可能です。この方法が機能するために、ポート番号が一致する必要はありません。



注記

1024 未満のポートで待機するようにポート転送を設定するには、root レベルのアクセスが必要です。

localhost 上で接続を待機する TCP/IP ポート転送チャンネルを作成するには、以下の形式でコマンドを使用します:

```
ssh -L local-port:remote-hostname:remote-port username@hostname
```

例えば、暗号化された接続を介し **POP3** を使用して、**mail.example.com** と呼ばれるサーバーで電子メールを確認するには、以下のコマンドを使用します:

```
~]$ ssh -L 1100:mail.example.com:110 mail.example.com
```

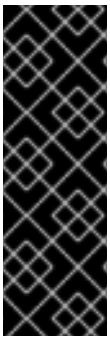
ポート転送チャンネルがクライアントマシンとメールサーバー間に配置されたら、POP3 メールクライアントに対し **localhost** 上のポート **1100** を使用して、新規の電子メールを確認するように指示します。クライアントシステム上のポート **1100** に送信される全要求は **mail.example.com** サーバーにセキュアに移動します。

SSH サーバーを実行しているのが **mail.example.com** ではなく、同一のネットワーク上にある別のマシンの場合でも、SSH を使用して接続の一部をセキュアにすることができます。ただし、若干異なるコマンドが必要です:

```
~]$ ssh -L 1100:mail.example.com:110 other.example.com
```

この例では、クライアントマシン上のポート **1100** からの POP3 要求がポート **22** の SSH 接続を介して SSH サーバー **other.example.com** に転送されます。次に、**other.example.com** は **mail.example.com** 上のポート **110** に接続して、新規の電子メールを確認します。この手法を使用する場合、クライアントシステムと **other.example.com** SSH サーバー間の接続のみがセキュアである点に注意して下さい。

ポート転送は、ネットワークのファイアウォールを介してセキュアに情報を取得する場合にも使用できます。ファイアウォールが SSH トラフィックをその標準ポート (ポート 22) を介して許可するよう設定されているが、他のポートへのアクセスはブロックする場合でも、確立された SSH 接続でそうした通信をリダイレクトすることにより、ブロックされたポートを使用した 2 つのホスト間の接続は可能になります。



重要

この方法でポート転送を使って接続を転送すると、クライアントシステム上のどのユーザーもそのサービスに接続できます。クライアントシステムが侵害された場合、攻撃者は転送されたサービスにアクセスすることもできます。

システム管理者がポート転送に懸念がある場合は、**/etc/ssh/sshd_config** にある **AllowTcpForwarding** の行に **No** パラメーターを指定して **sshd** サービスを再起動することにより、サーバー上でこの機能を無効にすることができます。

12.5. その他のリソース

OpenSSH および OpenSSL プロジェクトの開発は常に進められているため、最新情報はそれぞれの Web サイトを参照してください。OpenSSH と OpenSSL ツールの man ページでも詳しい情報を確認することができます。

12.5.1. インストールされているドキュメント

- **sshd**(8) — **sshd** デーモンの man ページです。
- **ssh**(1) — **ssh** クライアントの man ページです。
- **scp**(1) — **scp** ユーティリティの man ページです。

- **sftp**(1) — **sftp** ユーティリティの man ページです。
- **ssh-keygen**(1) — **ssh-keygen** ユーティリティの man ページです。
- **ssh_config**(5) — 使用可能な SSH クライアントの設定オプションに関する全説明を記載した man ページです。
- **sshd_config**(5) — 使用可能な SSH デーモンの設定オプションに関する全説明を記載した man ページです。

12.5.2. 役立つ Web サイト

http://www.openssh.com/

さらに詳しいドキュメントや FAQ、メーリングリストへのリンク、バグレポート、その他役立つ資料を掲載した OpenSSH のホームページです。

http://www.openssl.org/

さらに詳しいドキュメントや FAQ、メーリングリストへのリンク、その他役立つリソースを掲載した OpenSSL のホームページです。

http://www.freesshd.com/

SSH サーバーの別の実装です。

---

[5] 多重接続は、共有されている共通の媒体上で送信される信号で構成されます。SSH により、異なるチャンネルが共通のセキュアな接続で送信されます。

パート V. サーバー

このセクションでは、ウェブサーバーのセットアップ、あるいはファイル及びディレクトリをネットワーク上で共有する方法などの、サーバーに関連した各種トピックを取り扱います。

第13章 DHCP サーバー

DHCP (Dynamic Host Configuration Protocol: 動的ホスト構成プロトコル) は、クライアントマシンに TCP/IP 情報を自動的に割り当てるネットワークプロトコルです。各 DHCP クライアントは、中央に配置された DHCP サーバーに接続します。DHCP サーバーは、クライアントのネットワーク設定情報 (IP アドレス、ゲートウェイ、DNS サーバーなど) を返します。

13.1. DHCP を使用する理由

DHCP は、クライアントのネットワークインターフェースの自動設定に役立ちます。クライアントシステムの設定時に、IP アドレス、ネットマスク、ゲートウェイや DNS サーバーを指定する代わりに DHCP を選択することができます。これにより、クライアントはそうした情報を DHCP サーバーから受け取ります。また、多数のシステムの IP アドレスを変更したい場合にも DHCP は役立ちます。すべてのシステムを再設定する代わりに、サーバー上の 1 つの設定ファイルを編集するだけで、新規の IP アドレスセットを設定できます。組織の DNS サーバーが変更された場合は、DHCP クライアントではなく、DHCP サーバーで変更が行われます。ネットワークを再起動、あるいはクライアントを再起動すると、変更が反映されます。

組織が機能的な DHCP サーバーをネットワークに正しく接続している場合、ラップトップや他のモバイルコンピューターのユーザーはそうしたデバイスをオフィス間で移動して使用できます。

13.2. DHCP サーバーの設定

dhcp パッケージには ISC (Internet Systems Consortium) DHCP サーバーが含まれています。最初に、スーパーユーザーとしてパッケージをインストールします。

```
~]# yum install dhcp
```

dhcp パッケージをインストールすると、**/etc/dhcp/dhcpd.conf** ファイルが作成されます。これは単なる空の設定ファイルです。

```
~]# cat /etc/dhcp/dhcpd.conf
#
# DHCP Server Configuration file.
#   see /usr/share/doc/dhcp*/dhcpd.conf.sample
```

設定ファイルのサンプルは、**/usr/share/doc/dhcp-*<version>*/dhcpd.conf.sample** にあります。このファイルは、**/etc/dhcp/dhcpd.conf** を設定する場合に使用することをお勧めします。詳しくは以下で説明します。

また、DHCP は **/var/lib/dhcpd/dhcpd.leases** ファイルを使用してクライアントのリースデータベースを格納します。詳細は 「リースデータベース」 を参照して下さい。

13.2.1. 設定ファイル

DHCP サーバーを設定するには、最初にクライアントのネットワーク情報を格納する設定ファイルを作成します。このファイルを使用して、クライアントシステム用のオプション及びグローバルオプションを宣言します。

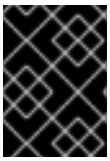
設定ファイルには追加のタブや空白行が含まれているため、簡単に書式を整えることができます。キーワードは大文字/小文字の区別がなく、行頭がハッシュ記号 (#) で始まる行はコメントとみなされます。

設定ファイルのステートメントには、次のような 2 つのタイプがあります。

- パラメーター — タスクの実行方法、タスクを実行するかどうか、クライアントに送信するネットワーク設定のオプションを規定します。
- 宣言 — ネットワークトポロジの記述、クライアントの記述、クライアントのアドレス指定、宣言グループへのパラメーターグループの適用を行います。

キーワードオプションから始まるパラメーターは、*オプション* と呼ばれます。オプションは、DHCP オプションを制御します。一方、パラメーターはオプションでない値の設定、DHCP サーバーの動作の制御を行います。

中括弧 ({ }) で囲まれたセクションの前に宣言されたパラメーター (オプションを含む) は、グローバルパラメーターとみなされます。グローバルパラメーターは、これ以降のすべてのセクションに適用します。



重要

設定ファイルが変更された場合、**service dhcpd restart** コマンドを使って DHCP デーモンを再起動するまでは変更内容は反映されません。

注記

毎回 DHCP 設定ファイルを変更してサービスを再起動させる代わりに、**omshell** コマンドを使用すると、DHCP サーバーへの接続、クエリ、設定の変更をインタラクティブに行うことができます。**omshell** を使用すると、DHCP サーバーの実行中でも変更を行うことができます。**omshell** の詳細については、**omshell** の man ページを参照して下さい。

例13.1「subnet 宣言」 では、**routers**、**subnet-mask**、**domain-search**、**domain-name-servers** 及び **time-offset** オプションは、subnet 宣言下で宣言されている **host** ステートメント用に使用されます。

機能するすべての **subnet**、DHCP サーバーの接続先となるすべての **subnet** に対して、**subnet** 宣言が 1 つ必要です。これは、アドレスがその **subnet** にあることを認識する方法を DHCP デーモンに伝えます。**subnet** に動的に割り当てるアドレスがない場合でも、各 **subnet** に対して **subnet** 宣言が 1 つ必要となります。

この例では、サブネット内のすべての DHCP クライアントに対するグローバルオプションがあり、**range** が宣言されています。クライアントに **range** 内の IP アドレスが割り当てられています。

例13.1 subnet 宣言

```
subnet 192.168.1.0 netmask 255.255.255.0 {
        option routers                  192.168.1.254;
        option subnet-mask              255.255.255.0;
        option domain-search              "example.com";
        option domain-name-servers       192.168.1.1;
        option time-offset              -18000;     # Eastern Standard
Time
 range 192.168.1.10 192.168.1.100;
}
```

サブネット内でシステムに動的 IP アドレスをリースする DHCP サーバーを設定するには、例

13.2「range パラメーター」 を使用する値に変更します。これにより、クライアント用のデフォルトのリース期間、最大リース期間、ネットワークの設定値を宣言します。この例では、192.168.1.10 から 192.168.1.100 の **range** の IP アドレスをクライアントシステムに割り当てます。

例13.2 range パラメーター

```
default-lease-time 600;
max-lease-time 7200;
option subnet-mask 255.255.255.0;
option broadcast-address 192.168.1.255;
option routers 192.168.1.254;
option domain-name-servers 192.168.1.1, 192.168.1.2;
option domain-search "example.com";
subnet 192.168.1.0 netmask 255.255.255.0 {
   range 192.168.1.10 192.168.1.100;
}
```

ネットワークインターフェースカードの MAC アドレスに基づいてクライアントに IP アドレスを割り当てるには、**host** 宣言内の **hardware ethernet** パラメーターを使用します。例13.3「DHCP を使用した静的 IP アドレス」 の例では、**host apex** 宣言は、MAC アドレス 00:A0:78:8E:9E:AA を持つネットワークインターフェースカードが常に IP アドレス 192.168.1.4 を受信するように指定します。

オプションのパラメーターである **host-name** を使用すると、クライアントにホスト名を割り当てることができる点にも注目して下さい。

例13.3 DHCP を使用した静的 IP アドレス

```
host apex {
   option host-name "apex.example.com";
   hardware ethernet 00:A0:78:8E:9E:AA;
   fixed-address 192.168.1.4;
}
```

同じ物理ネットワークを共有するすべてのサブネットは、例13.4「shared-network 宣言」 のとおり **shared-network** 宣言内で宣言する必要があります。**shared-network** 内だが囲まれた **subnet** 宣言の外にあるパラメーターは、グローバルパラメーターとみなされます。**shared-network** の名前は、そのネットワークを表す分かりやすいタイトルにします。例えば、テストラボ環境の全サブネットを表す「test-lab」というタイトルなどです。

例13.4 shared-network 宣言

```
shared-network name {
    option domain-search              "test.redhat.com";
    option domain-name-servers      ns1.redhat.com, ns2.redhat.com;
    option routers                  192.168.0.254;
    more parameters for EXAMPLE shared-network
    subnet 192.168.1.0 netmask 255.255.252.0 {
        parameters for subnet
        range 192.168.1.1 192.168.1.254;
    }
    subnet 192.168.2.0 netmask 255.255.252.0 {
        parameters for subnet
```

```
        range 192.168.2.1 192.168.2.254;
    }
}
```

例13.5「group 宣言」 のように **group** 宣言を使用すると、宣言グループにグローバルパラメーターを適用できます。例えば、共有ネットワーク、サブネット、ホストをグループ化できます。

例13.5 group 宣言

```
group {
   option routers                  192.168.1.254;
   option subnet-mask              255.255.255.0;
   option domain-search              "example.com";
   option domain-name-servers       192.168.1.1;
   option time-offset              -18000;     # Eastern Standard Time
   host apex {
      option host-name "apex.example.com";
      hardware ethernet 00:A0:78:8E:9E:AA;
      fixed-address 192.168.1.4;
   }
   host raleigh {
      option host-name "raleigh.example.com";
      hardware ethernet 00:A1:DD:74:C3:F2;
      fixed-address 192.168.1.6;
   }
}
```

注記

提供されている設定ファイルのサンプルをベースとして使用し、カスタムの設定オプションを追加することができます。このファイルを適切な場所にコピーするには、以下のコマンドを使用します。

```
cp /usr/share/doc/dhcp-<version_number>/dhcpd.conf.sample
/etc/dhcp/dhcpd.conf
```

<version_number> は DHCP のバージョン番号です。

オプションステートメントとその機能の全一覧については、**dhcp-options** の man ページを参照して下さい。

13.2.2. リースデータベース

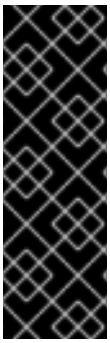
DHCP サーバーでは、**/var/lib/dhcpd/dhcpd.leases** ファイルが DHCP クライアントのリースデータベースを格納します。このファイルは変更しないで下さい。最近割り当てられた各 IP アドレス用の DHCP リース情報は、リースデータベースに自動的に保存されます。この情報には、リース期間、IP アドレスの割り当て先、リースの開始日と終了日、リースの取得に使用されるネットワークインターフェースカードの MAC アドレスが含まれます。

リースデータベースの時刻はすべて、現地時間でなく協定世界時 (UTC) を使用します。

リースデータベースは、サイズが大きくなり過ぎるのを避けるために適宜再作成されます。最初に、すべての既知のリースは一時的なリースデータベースに保存されます。**dhcpd.leases** ファイルの名前は **dhcpd.leases~** に変更され、一時的なリースデータベースが **dhcpd.leases** に書き込まれます。

リースデータベースがバックアップファイルに名前変更され新規ファイルが書き込まれる前に、DHCP デーモンが kill されるか、システムがクラッシュする可能性があります。この場合、**dhcpd.leases** ファイルは存在しませんが、サービスを起動する必要があります。その際に新規のリースファイルを作成しないで下さい。作成してしまうと、それまでのリースはすべて失われ、多くの問題が発生します。これを解決する適切な方法は、**dhcpd.leases~** バックアップファイルの名前を **dhcpd.leases** に変更して、デーモンを起動することです。

13.2.3. サーバーの起動と停止



重要

DHCP サーバーを初めて起動する場合、**dhcpd.leases** ファイルがなければ起動は失敗します。ファイルが存在しない場合は、**touch /var/lib/dhcpd/dhcpd.leases** コマンドを使用してファイルを作成して下さい。

同じ DHCP サーバーが DNS サーバーとして BIND も実行している場合は、この手順は必要はありません。**named** サービスを起動すると **dhcpd.leases** ファイルを自動的に確認するためです。

DHCP サービスを起動するには、**/sbin/service dhcpd start** コマンドを使用します。DHCP サーバーを停止するには、**/sbin/service dhcpd stop** コマンドを使用します。

デフォルトでは、DHCP サービスはブート時に起動しません。ブート時にデーモンが自動的に起動するように設定する方法については、10章 *サービスとデーモン* を参照してください。

システムに複数のネットワークインターフェースが接続されているが、そのうち 1 つのインターフェースのみで DHCP サーバーを起動する必要がある場合には、そのデバイスだけで起動するように DHCP サーバーを設定します。**/etc/sysconfig/dhcpd** 内にある **DHCPDARGS** の一覧にインターフェース名を追加します:

```
# Command line options here
DHCPDARGS=eth0
```

これは、ネットワークカードが 2 つあるファイアウォールマシンに役立ちます。1 つのネットワークカードを DHCP クライアントとして設定してインターネット用の IP アドレスを取得します。もう 1 つのネットワークカードは、ファイアウォールの背後にある内部ネットワーク用の DHCP サーバーとして使用できます。内部ネットワークに接続されたネットワークカードのみを指定することにより、ユーザーはインターネット経由でデーモンに接続できないため、システムをよりセキュアにすることができます。

以下は、**/etc/sysconfig/dhcpd** で指定できるその他のコマンドラインオプションです。

- **-p *<portnum>*** — **dhcpd** が待機する UDP ポート番号を指定します。デフォルト値はポート 67 です。DHCP サーバーは、指定された UDP ポートよりも番号が 1 つ大きいポートにある DHCP クライアントに応答を送信します。例えば、デフォルトのポート 67 を使用する場合、サーバーはポート 67 で要求を待ち受け、ポート 68 にあるクライアントに応答します。ここでポートが指定され、DHCP リレーエージェントが使用される場合、DHCP リレーエージェントが待機するポートと同じポートが指定されなければなりません。詳細は 「DHCP リレーエージェント」 を参照してください。

- **-f** — フォアグラウンドプロセスとしてデーモンを実行します。これは主にデバッグ用に使用されます。
- **-d** — 標準エラー記述子に DCHP サーバーデーモンを記録します。これは主にデバッグ用に使用されます。このオプションを指定しないと、ログは **/var/log/messages** に書き込まれます。
- **-cf *<filename>*** — 設定ファイルの場所を指定します。デフォルトの場所は **/etc/dhcp/dhcpd.conf** です。
- **-lf *<filename>*** — リースデータベースファイルの場所を指定します。リースデータベースファイルが既に存在する場合、DHCP サーバーを起動する度に、同じファイルが使用されるようにすることが非常に重要です。このオプションは、実稼働環境以外のマシンでデバッグする目的にのみ使用することを強くお勧めします。デフォルトの場所は **/var/lib/dhcpd/dhcpd.leases** です。
- **-q** — デーモンの起動時に、著作権に関するメッセージ全体を表示しません。

13.2.4. DHCP リレーエージェント

DHCP リレーエージェント (**dhcrelay**) を使用すると、DHCP 及び BOOTP の要求を、DHCP サーバーのないサブネットから他のサブネット上にある 1 つ以上の DHCP サーバーへ中継することができます。

DHCP クライアントが情報を要求すると、DHCP リレーエージェントは起動時に指定された DHCP サーバーの一覧に要求を転送します。DHCP サーバーが応答を返すと、その応答は元の要求を送信したネットワーク上でブロードキャストまたはユニキャストで返されます。

/etc/sysconfig/dhcrelay で **INTERFACES** 指示文によりインターフェースが指定されている場合を除き、DHCP リレーエージェントはすべてのインターフェース上で DHCP 要求を待機します。

DHCP リレーエージェントを開始するには、**service dhcrelay start** コマンドを使用します。

13.3. DHCP クライアントの設定

DHCP クライアントを手動で設定するには、**/etc/sysconfig/network** ファイルを修正して、ネットワーキングと **/etc/sysconfig/network-scripts** ディレクトリ内にある各ネットワークデバイスの設定ファイルを有効にします。このディレクトリ内には、デバイスごとに **ifcfg-eth0** と呼ばれる設定ファイルがあるはずです。**eth0** はネットワークデバイス名です。

/etc/sysconfig/network-scripts/ifcfg-eth0 ファイルに、以下の行が含まれていることを確認してください。

```
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
```

DHCP を使用するには、各デバイスごとに設定ファイルを設定します。

ネットワークスクリプトの他のオプションは以下のとおりです。

- **DHCP_HOSTNAME** — このオプションを使用するのは、DHCP サーバーが IP アドレスを受信する前にクライアントがホスト名を指定する必要がある場合のみです。

- **PEERDNS=<*answer*>** の **<*answer*>** は以下のいずれかになります。
  - **yes** — サーバーからの情報に応じて **/etc/resolv.conf** を変更します。DHCP を使用する場合は、**yes** がデフォルトです。
  - **no** — **/etc/resolv.conf** を変更しません。

グラフィカルインターフェースを使用したい場合、DHCP を使用するようにネットワークインターフェースを設定するための **NetworkManager** の使用手順については、8章*NetworkManager* を参照してください。

注記

プロトコルのタイミング、リースの要件と要求、動的 DNS サポート、エイリアス、さらにはクライアント側の設定に上書き、先頭や末尾に追加する様々な値など、クライアント DHCP オプションの高度な設定については、**dhclient** と **dhclient.conf** の man ページを参照して下さい。

13.4. マルチホーム DHCP サーバーの設定

マルチホーム DHCP サーバーは、複数のネットワーク、すなわち複数のサブネットとして機能します。以下のセクションにあげる例では、DHCP サーバーが複数のネットワークで機能するように設定する方法、待機するネットワークインターフェースを選択する方法、ネットワークを移動するシステム用にネットワーク設定を定義する方法について詳しく示しています。

変更を行う前に、既存の **/etc/sysconfig/dhcpd** 及び **/etc/dhcp/dhcpd.conf** ファイルをバックアップしておいて下さい。

DHCP デーモンは、指定されていない限りすべてのネットワークインターフェースで待機します。**/etc/sysconfig/dhcpd** ファイルを使用して、DHCP デーモンが待機するネットワークインターフェースを指定して下さい。以下の **/etc/sysconfig/dhcpd** の例では、DHCP デーモンが **eth0** 及び **eth1** のインターフェースで待機するよう指定しています:

```
DHCPDARGS="eth0 eth1";
```

システムに **eth0**、**eth1**、**eth2** の 3 つのネットワークインターフェースカードがあるが、DHCP デーモンを **eth0** カードでのみ待機させたい場合は、**/etc/sysconfig/dhcpd** で **eth0** のみを指定します。

```
DHCPDARGS="eth0";
```

以下は、10.0.0.0/24 ネットワークの **eth0** と、172.16.0.0/24 ネットワークの **eth1** という 2 つのネットワークインターフェースを持つサーバー用の基本的な **/etc/dhcp/dhcpd.conf** ファイルです。複数の **subnet** 宣言により、複数のネットワークに対して異なる設定を定義することができます。

```
default-lease-time 600;
max-lease-time 7200;
subnet 10.0.0.0 netmask 255.255.255.0 {
 option subnet-mask 255.255.255.0;
 option routers 10.0.0.1;
 range 10.0.0.5 10.0.0.15;
}
subnet 172.16.0.0 netmask 255.255.255.0 {
```

```
 option subnet-mask 255.255.255.0;
 option routers 172.16.0.1;
 range 172.16.0.5 172.16.0.15;
}
```

subnet *10.0.0.0* netmask *255.255.255.0*;

subnet 宣言は、ご使用の DHCP サーバーが機能するすべてのネットワークに必要です。複数のサブネットには、複数の **subnet** 宣言が必要です。**subnet** 宣言の範囲内で DHCP サーバーにネットワークインターフェースがない場合は、DHCP サーバーはそのネットワークでは機能しません。

subnet 宣言が 1 つのみで、かつそのサブネットの範囲内にネットワークインターフェースがない場合には、DHCP デーモンは起動に失敗して、以下のようなエラーが **/var/log/messages** に記録されます:

```
dhcpd: No subnet declaration for eth0 (0.0.0.0).
dhcpd: ** Ignoring requests on eth0.  If this is not what
dhcpd:    you want, please write a subnet declaration
dhcpd:    in your dhcpd.conf file for the network segment
dhcpd:    to which interface eth1 is attached. **
dhcpd:
dhcpd:
dhcpd: Not configured to listen on any interfaces!
```

option subnet-mask *255.255.255.0*;

option subnet-mask オプションは、サブネットマスクを定義し、**subnet** 宣言内の **netmask** 値を上書きします。簡単なケースでは、サブネットとネットマスクの値は同じです。

option routers *10.0.0.1*;

option routers オプションは、サブネット用のデフォルトゲートウェイを定義します。これは、システムが異なるサブネット上の内部ネットワーク、さらには外部ネットワークに届くために必要です。

range *10.0.0.5 10.0.0.15*;

range オプションは、利用可能な IP アドレスのプールを指定します。指定された IP アドレスの範囲の中からアドレスがシステムに割り当てられます。

詳細は **dhcpd.conf(5)** の man ページを参照して下さい。

警告

別名インターフェースは、DHCP ではサポートされていません。別名インターフェースが **/etc/dhcp/dhcpd.conf** で指定されている唯一のサブネット内で唯一のインターフェースである場合は、DHCP デーモンは起動に失敗します。

13.4.1. ホストの設定

変更を行う前に、既存の **/etc/sysconfig/dhcpd** 及び **/etc/dhcp/dhcpd.conf** ファイルをバックアップしておいて下さい。

複数ネットワークに対する単一システムの設定

以下の **/etc/dhcp/dhcpd.conf** の例では、2 つのサブネットを作成し、接続するネットワークに応じて IP アドレスを同じシステム用に設定しています:

```
default-lease-time 600;
max-lease-time 7200;
subnet 10.0.0.0 netmask 255.255.255.0 {
 option subnet-mask 255.255.255.0;
 option routers 10.0.0.1;
 range 10.0.0.5 10.0.0.15;
}
subnet 172.16.0.0 netmask 255.255.255.0 {
 option subnet-mask 255.255.255.0;
 option routers 172.16.0.1;
 range 172.16.0.5 172.16.0.15;
}
host example0 {
 hardware ethernet 00:1A:6B:6A:2E:0B;
 fixed-address 10.0.0.20;
}
host example1 {
 hardware ethernet 00:1A:6B:6A:2E:0B;
 fixed-address 172.16.0.20;
}
```

host *example0*

host 宣言は、単一のシステム用に IP アドレスなどの特定のパラメーターを定義します。複数のホスト用に特定のパラメーターを設定するには、複数の **host** 宣言を使用します。

大半の DHCP クライアントは **host** 宣言内の名前を無視するため、他の **host** 宣言に対して一意である限りはどのような名前でも構いません。同じシステムを複数のネットワークに対して設定するには、各 **host** 宣言に異なる名前を使用します。そうしなければ、DHCP デーモンは起動に失敗します。システムは **host** 宣言内の名前ではなく、**hardware ethernet** オプションによって識別されます。

hardware ethernet *00:1A:6B:6A:2E:0B*;

hardware ethernet オプションは、システムを識別します。アドレスを確認するには、**ip link** コマンドを実行します。

fixed-address *10.0.0.20*;

fixed-address オプションは、**hardware ethernet** オプションによって指定されたシステムに有効な IP アドレスを割り当てます。このアドレスは、**range** オプションで指定された IP アドレスプール外でなければなりません。

option ステートメントの最後にセミコロンが付いていないと、DHCP デーモンは起動に失敗し、以下のようなエラーが **/var/log/messages** に記録されます:

```
/etc/dhcp/dhcpd.conf line 20: semicolon expected.
dhcpd: }
```

```
dhcpd: ^
dhcpd: /etc/dhcp/dhcpd.conf line 38: unexpected end of file
dhcpd:
dhcpd: ^
dhcpd: Configuration file errors encountered -- exiting
```

複数のネットワークインターフェースを持つシステムの設定

以下の **host** 宣言は、複数のネットワークインターフェースを持つ単一のシステムを設定するため、各インターフェースは同一の IP アドレスを受け取ります。両方のネットワークインターフェースが同じネットワークに同時に接続されている場合には、この設定は機能しません。

```
host interface0 {
 hardware ethernet 00:1a:6b:6a:2e:0b;
 fixed-address 10.0.0.18;
}
host interface1 {
 hardware ethernet 00:1A:6B:6A:27:3A;
 fixed-address 10.0.0.18;
}
```

この例では、**interface0** が第 1 のネットワークインターフェースで、**interface1** が第 2 のインターフェースです。異なる **hardware ethernet** オプションが各インターフェースを識別します。

このようなシステムを別のネットワークに接続するには、**host** 宣言をさらに追加します。ただし、以下の点に注意して下さい。

- ホストが接続されるネットワーク用の有効な **fixed-address** を割り当てます。
- **host** 宣言の名前は一意にします。

host 宣言で指定した名前が一意でないと、DHCP デーモンは起動に失敗し、以下のようなエラーが **/var/log/messages** に記録されます:

```
dhcpd: /etc/dhcp/dhcpd.conf line 31: host interface0: already exists
dhcpd: }
dhcpd: ^
dhcpd: Configuration file errors encountered -- exiting
```

このエラーは、**/etc/dhcp/dhcpd.conf** に定義された **host interface0** 宣言が複数定義されているために生じました。

13.5. IPV6 の DHCP (DHCPV6)

DHCPv6 サーバー、クライアント、リレーエージェント機能を装備した ISC DHCP バージョン 4 のリリース以降、ISC DHCP は IPv6 (DHCPv6) をサポートしています。サーバー、クライアント及びリレーエージェントは、IPv4 と IPv6 の両方に対応しています。ただし、クライアントとサーバーは、1 回に 1 つのプロトコルのみ管理できます。デュアルサポートには、IPv4 と IPv6 用に別々に起動する必要があります。

DHCPv6 サーバーの設定ファイルは、**/etc/dhcp/dhcpd6.conf** にあります。

サーバー設定ファイルのサンプルは、**/usr/share/doc/dhcp-<version>/dhcpd6.conf.sample** にあります。

DHCPv6 サービスを起動するには、**/sbin/service dhcpd6 start** コマンドを使用します。

以下は、DHCPv6 サーバー設定ファイルの簡単な例です:

```
subnet6 2001:db8:0:1::/64 {
        range6 2001:db8:0:1::129 2001:db8:0:1::254;
        option dhcp6.name-servers fec0:0:0:1::1;
        option dhcp6.domain-search "domain.example";
}
```

13.6. その他のリソース

追加情報については、『The DHCP Handbook (Ralph Droms、Ted Lemon 共著、2003)』 や以下のリソースを参照して下さい。

13.6.1. インストールされているドキュメント

- **dhcpd** の man ページ — DHCP デーモンの動作の仕組みを説明しています。
- **dhcpd.conf** の man ページ — DHCP 設定ファイルの設定方法と数例が含まれています。
- **dhcpd.leases** の man ページ — リースの持続的データベースについて説明しています。
- **dhcp-options** の man ページ — **dhcpd.conf** の DHCP オプション宣言の構文に関する説明と数例が記載されています。
- **dhcrelay** の man ページ — DHCP リレーエージェントとその設定オプションについて説明しています。
- **/usr/share/doc/dhcp-*<version>*/** — DHCP サービスの現行バージョン用のサンプルファイル、README ファイル及びリリースノートが含まれています。

第14章 DNS サーバー

ネームサーバー としても知られる **DNS** (Domain Name System ドメインネームシステム) は、ホスト名とその IP アドレスを関連付けるネットワークシステムです。ユーザーにとっては、ネットワーク上のマシンを名前で参照できるという利点があります。数字で表されるネットワークアドレスを覚えるより通常は簡単でしょう。システム管理者にとっては、ネームサーバーを使用すると、名前ベースのクエリに影響を与えることなくホスト用の IP アドレスを変更でき、そうしたクエリをどのマシンが処理するか決定することができます。

14.1. DNS の概要

DNS は通常、特定のドメインに権威ある 1 つまたは複数の集中型サーバーを使用して実装されています。クライアントホストがネームサーバーから情報を要求すると、ネームサーバーは通常ポート 53 に接続します。その後、ネームサーバーは要求された名前の解決を試行します。ネームサーバーに権威ある回答がない場合、または以前のクエリよりキャッシュされた回答がない場合は、*ルートネームサーバー* と呼ばれる他のネームサーバーにクエリを行い、問題の名前に対して権威あるネームサーバーがどれか決定します。その後、要求した名前を取得するためにルートネームサーバーにクエリを行います。

14.1.1. ネームサーバーのゾーン

BIND (Berkeley Internet Name Domain) などの DNS サーバーでは、全情報は *リソースレコード* (RR) と呼ばれる基本的なデータ要素に保存されています。リソースレコードは通常、ホストの *完全修飾ドメイン名* (FQDN) であり、ツリーのような階層で構成された複数のセクションに分かれています。この階層はメイントランク、プライマリブランチ、セカンダリブランチというように構成されています。

例14.1 シンプルなリソースレコード

```
bob.sales.example.com
```

階層の各レベルはピリオド (`.`) で分かれています。例14.1「シンプルなリソースレコード」 では、**com** が *トップレベルドメイン*、**example** がそのサブドメイン、**example** のサブドメインである **sales** が定義されています。この場合は、**sales.example.com** ドメインの一部である **bob** がリソースレコードを識別します。左端の部分 (**bob**) を除いては、これらのセクションはそれぞれ *ゾーン* と呼ばれ、特定の *名前空間* を定義します。

ゾーンは *ゾーンファイル* を使用して権威ネームサーバー上で定義されます。ゾーンファイルには、各ゾーンのリソースレコードの定義が含まれています。ゾーンファイルは、ファイルへの変更が行われる *プライマリネームサーバー* (別名 *マスターネームサーバー*) 、プライマリネームサーバーからゾーン定義を受け取る *セカンダリネームサーバー* (別名 *スレーブネームサーバー*) に保存されています。プライマリネームサーバー、セカンダリネームサーバーともゾーンに対し権威があり、クライアントには同一に見えます。設定により、どのネームサーバーも複数ゾーンに対してプライマリサーバーまたはセカンダリサーバーとして同時に機能することができます。

14.1.2. ネームサーバーのタイプ

ネームサーバーの設定タイプには以下のように 2 つあります。

権威

権威ネームサーバーはゾーンの一部であるリソースレコードに対してのみ回答します。このカテゴリにはプライマリ (マスター) ネームサーバーとセカンダリ (スレーブ) ネームサーバーの両方が含まれます。

再帰

再帰ネームサーバーは解決サービスを行いますが、どのゾーンに対しても権威ではありません。すべての解決への回答は一定期間はメモリにキャッシュされ、取得したリソースレコードで指定されます。

ネームサーバーは権威的であると同時に再帰的でもありますが、設定タイプを組み合わせることは推奨されません。権威サーバーが機能できるようにするには、常にすべてのクライアントが利用できるものでなくてはなりません。一方で、再帰的ルックアップは権威ある応答よりはるかに時間がかかるため、再帰的なサーバーは限られた数のクライアントだけが利用できるものでなくてはなりません。そうでないと、DDoS 攻撃 (分散型サービス拒否攻撃) の可能性が高まります。

14.1.3. ネームサーバーとしての BIND

BIND は一連の DNS 関連プログラムで構成されています。それには **named** と呼ばれるネームサーバー、**rndc** という管理ユーティリティ、**dig** というデバッグツールが含まれています。Red Hat Enterprise Linux におけるサービスの実行方法については 10章 *サービスとデーモン* を参照してください。

14.2. BIND

この章では、Red Hat Enterprise Linux に収納されている DNS サーバーである **BIND** (Berkeley Internet Name Domain) について説明します。ここでは、その設定ファイルの構造に焦点を置いて、ローカルとリモートの両方での管理方法を記述しています。

14.2.1. named サービスの設定

named サービスが始まると、表14.1「named サービスの設定ファイル」 説明してあるように、ファイルから設定を読み込みます。

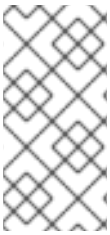
表14.1 named サービスの設定ファイル

パス	詳細
/etc/named.conf	主要設定ファイル
/etc/named/	主要設定ファイル内に含まれている設定ファイル用の補助ディレクトリ

設定ファイルは、弓括弧で囲まれた入り子オプションを持つステートメントの集合で成り立っています。このファイルを編集する時は、構文エラーを出さないように注意する必要があります。もしエラーがあると **named** サービスは開始しません。標準的な **/etc/named.conf** ファイルは以下のような構成となります:

```
statement-1 ["statement-1-name"] [statement-1-class] {
  option-1;
  option-2;
  option-N;
};
statement-2 ["statement-2-name"] [statement-2-class] {
  option-1;
  option-2;
```

```
  option-N;
};
statement-N ["statement-N-name"] [statement-N-class] {
  option-1;
  option-2;
  option-N;
};
```



注記

bind-chroot パッケージをインストールしているならば、BIND サービスは **/var/named/chroot** 環境内で実行出来ます。その場合、初期化スクリプトが、**mount --bind** コマンドを使用して上記の設定ファイルをマウントするため、この環境外でも設定の管理ができるようになります。

14.2.1.1. 一般的なステートメントのタイプ

以下のタイプのステートメントが一般的に **/etc/named.conf** で使用されます:

acl

acl (Access Control List) (アクセス制御リスト) ステートメントにより、ホストのグループを定義できるようになるため、それらのホストはネームサーバーへのアクセスを許可/拒否できるようになります。以下の形式を取ります:

```
acl acl-name {
  match-element;
  ...
};
```

acl-name ステートメント名はアクセス制御リストの名前です。そして *match-element* オプションは通常、個別の IP アドレス (**10.0.1.1** など) または、CIDR (クラスレス ドメイン間ルーティング) ネットワーク表記 (例えば、**10.0.1.0/24**) です。定義済みキーワードの一覧については 表14.2「事前定義のアクセス制御一覧」 を参照してください。

表14.2 事前定義のアクセス制御一覧

キーワード	詳細
any	すべての IP アドレスに一致
localhost	ローカルシステムで使用中のすべての IP アドレスに一致
localnets	ローカルシステムが接続しているすべてのネットワーク上の IP アドレスに一致
none	いずれの IP アドレスにも一致しない

acl ステートメントは **options** などの他のステートメントとの併用で特に便利です。例14.2「options との併用で acl の使用」 は **black-hats** と **red-hats** の 2 つのアクセス制御一覧を定義して、**red-hats** に通常アクセスを許可する一方で **black-hats** をブラックリストに追加します。

例14.2 options との併用で acl の使用

```
acl black-hats {
  10.0.2.0/24;
  192.168.0.0/24;
  1234:5678::9abc/24;
};
acl red-hats {
  10.0.1.0/24;
};
options {
  blackhole { black-hats; };
  allow-query { red-hats; };
  allow-query-cache { red-hats; };
};
```

include

include ステートメントにより、ファイルを **/etc/named.conf** 内に含むことが出来るため、機密性のあるデータを制限付き権限で別のファイルに配置することができます。以下の形式を取ります

```
include "file-name"
```

file-name ステートメント名はファイルへの絶対パスとなります。

例14.3 /etc/named.conf へファイルをインクルードする

```
include "/etc/named.rfc1912.zones";
```

options

options ステートメントにより、グローバルサーバー設定オプションを定義できると共に他のステートメントのデフォルトをセットすることも出来ます。これは **named** の作業ディレクトリの場所、許可されるクエリのタイプ、及びその他多くを指定するのに使用されます。以下の形式を取ります:

```
options {
  option;
  ...
};
```

頻繁に使用される *option* 指示文の一覧については、以下の 表14.3「一般的に使用されるオプション」 をご覧下さい。

表14.3 一般的に使用されるオプション

オプション	詳細

オプション	詳細
allow-query	権限のあるリソースレコード用のネームサーバーにクエリを許可されるホストを指定します。これはアクセス制御一覧、IP アドレスの集合、または CIDR 表記にあるネットワーク群などを受け付けます。デフォルトではすべてのホストが許可されています。
allow-query-cache	回帰的クエリなど権限の必要ないデータ用のネームサーバーにクエリを許可されるホストを指定します。デフォルトでは、**localhost** と **localnets** のみが許可されています。
blackhole	ネームサーバーにクエリを許可**されない**ホストを指定します。このオプションは、特定のホストやネットワークがサーバーに要求を集中させる時には使用すべきではありません。デフォルトのオプションは **none** です。
directory	**named** サービス用の作業ディレクトリを指定します。デフォルトのオプションは **/var/named/** です。
dnssec-enable	DNSSEC 関連のリソースレコードを返すかどうかを指定します。デフォルトのオプションは **yes** です。
dnssec-validation	DNSSEC を介してリソースが純正であることを証明するかどうかを指定します。デフォルトのオプションは **yes** です。
forwarders	解決の為に要求を転送すべき相手のネームサーバーの有効な IP アドレス一覧を指定します。
forward	**forwarders** 指示文の動作を指定します。以下のオプションを受け付けます: • **first** — サーバーは、それ自身で名前の解決を試行する前に **forwarders** 指示文に一覧表示されているネームサーバーにクエリします。 • **only** — **forwarders** 指示文に一覧表示されたネームサーバーにクエリすることが出来ない時は、サーバーはそれ自身での名前解決を試行しません。
listen-on	クエリのリッスン先として IPv4 ネットワークインターフェイスを指定します。ゲートウェイとして機能する DNS サーバー上では、このオプションを使用することで単独ネットワークから由来するクエリのみに応答することができます。すべての IPv4 インターフェイスがデフォルトで使用されています。
listen-on-v6	クエリのリッスン先として IPv6 ネットワークインターフェイスを指定します。ゲートウェイとして機能する DNS サーバー上では、このオプションを使用することで単独ネットワークから由来するクエリのみに応答することができます。すべての IPv6 インターフェイスがデフォルトで使用されています。
max-cache-size	サーバー用キャッシュとして使用されるメモリーの最大容量を指定します。最大値に到達すると、その限度を超過しないようにサーバーは記録が早期期限切れになるようにします。複数表示を持つサーバーでは、この制限は各表示のキャッシュ毎に別々に適用されます。デフォルトのオプションは **32M** です。

オプション	詳細
notify	あるゾーンが更新された時にセカンダリネームサーバーに通知するかどうかを指定します。以下のオプションを受け付けます: • **yes** — サーバーはすべての セカンダリネームサーバーに通知します。 • **no** — サーバーはいずれの セカンダリネームサーバーにも通知**しません。** • **master-only** — サーバーはゾーン用のプライマリサーバーにのみ通知します。 • **explicit** — サーバーは、ゾーンステートメント内の **also-notify** 一覧に指定してあるセカンダリサーバーにのみ通知します。
pid-file	**named** サービスで作成されたプロセス ID ファイルの場所を指定します。
recursion	再帰的なサーバーとして動作するかどうかを指定します。デフォルトのオプションは **yes** です。
statistics-file	統計ファイルの代替の場所を指定します。デフォルトでは、**/var/named/named.stats** ファイルがデフォルトで使用されています。

重要

配信されるサービス拒否 (DDoS) 攻撃を阻止するために、**allow-query-cache** オプションを使用することで、クライアントの特定サブセット用にのみ再帰的な DNS サービスを制限することをお薦めします。

利用可能なオプションの完全な一覧を見るには 「インストールされているドキュメント」 内で参照されている『BIND 9 Administrator Reference Manual』 及び **named.conf** man ページを参照して下さい。

例14.4 オプションステートメントの使用

```
options {
  allow-query       { localhost; };
  listen-on port    53 { 127.0.0.1; };
  listen-on-v6 port 53 { ::1; };
  max-cache-size    256M;
  directory         "/var/named";
  statistics-file   "/var/named/data/named_stats.txt";

  recursion         yes;
  dnssec-enable     yes;
  dnssec-validation yes;
};
```

zone

zone ステートメントにより、その設定ファイルの場所やゾーン特有のオプションなどのゾーンの特性を定義できるようになり、グローバル **options** ステートメントを上書きするのに使用することができます。以下の形式を取ります:

```
zone zone-name [zone-class] {
  option;
  ...
};
```

zone-name 属性はゾーンの名前であり、*zone-class* はゾーンのオプションクラスであり、そして *option* は 表14.4「一般的に使用されるオプション」 で説明してあるように **zone** ステートメントのオプションです。

zone-name 属性は特に重要です。それは、**/var/named/** ディレクトリに配置されている該当ゾーンファイル内で使用される **$ORIGIN** 指示文に割り当てられたデフォルトの値だからです。**named** デーモンはゾーンの名前を、ゾーンファイル内に一覧表示された非完全修飾型のドメイン名のいずれかに追記します。例えば、**zone** ステートメントが **example.com** 用にネームスペースを定義する場合、**example.com** を *zone-name* として使用すると、それを **example.com** ゾーンファイル内のホスト名の末尾に配置することができます。

ゾーンファイルの詳細情報については、「ゾーンファイルの編集」 を参照して下さい。

表14.4 一般的に使用されるオプション

オプション	詳細
allow-query	このゾーンに関する情報要求が出来るクライアントを指定します。このオプションは、グローバル **allow-query** オプションを上書きします。デフォルトではすべてのクエリ要求が許可されます。
allow-transfer	ゾーン情報の転送要求を許可されるセカンダリサーバーを指定します。デフォルトでは、すべての転送要求が許可されています。
allow-update	自身のゾーン内で動的な情報更新を許可されるホストを指定します。デフォルトオプションでは、すべての動的更新要求は拒否されます。 そのゾーンに関してホストの情報更新の許可には注意が必要です。サーバーが信頼できるネットワークにある場合意外は、このオプションで IP アドレスをセットしないで下さい。その代わりに、「Transaction SIGnatures トランザクション署名 (TSIG)」 で説明してあるように TSIG キーを使用します。
file	ゾーンの設定データを収納している **named** 作業ディレクトリ内のファイル名を指定します。
masters	信頼できるゾーン情報の要求先の IP アドレスを指定します。このオプションは、ゾーンが **type slave** として定義されている場合にのみ使用されます。

オプション	詳細
notify	あるゾーンが更新された時にセカンダリネームサーバーに通知するかどうかを指定します。以下のオプションを受け付けます: • **yes** — サーバーはすべての セカンダリネームサーバーに通知します。 • **no** — サーバーはいずれの セカンダリネームサーバーにも通知**しません。** • **master-only** — サーバーはゾーン用のプライマリサーバーにのみ通知します。 • **explicit** — サーバーは、ゾーンステートメント内の **also-notify** 一覧に指定してあるセカンダリサーバーにのみ通知します。
type	ゾーンのタイプを指定します。以下のオプションを受け付けます: • **delegation-only** — COM、NET、ORG などのインフラストラクチャゾーンの委任ステータスを強制します。明示的あるいは暗示的な委任の無い受信回答は **NXDOMAIN** として取り扱われます。このオプションは、再帰的あるいはキャッシング実装で使用される TLD もしくは root のゾーンのファイルにのみ適用されます。 • **forward** — このゾーンに関する情報へのすべての要求を他のネームサーバーに転送します。 • **hint** — ゾーンが他の方法で認知されていない時にクエリを解決する root ネームサーバーへポイントするために使用される特殊タイプのゾーンです。 • **master** — このゾーン用の権威としてネームサーバーを指名します。ゾーンの設定ファイルがシステム上に存在する場合は、ゾーンは **master** としてセットされる必要があります。 • **slave** — このゾーン用のスレーブとしてネームサーバーを指名します。マスターサーバーは **masters** 指示文内に指定されています。

プライマリまたはセカンダリのネームサーバーの **/etc/named.conf** ファイルに対するほとんどの変更には、**zone** ステートメントの追加、修正、または削除が含まれ、通常は **zone** ステートメントオプションの小さなサブセットのみが、ネームサーバーの良好な機能のために必要となります。

例14.5「プライマリネームサーバー用のゾーンステートメント」 では、ゾーンは **example.com** として識別されており、タイプは **master** にセットされて、**named** サービスは **/var/named/example.com.zone** ファイルを読み込むように指示されています。これはまた、ゾーンの転送にセカンダリネームサーバー (**192.168.0.2**) だけを許可します。

例14.5 プライマリネームサーバー用のゾーンステートメント

```
zone "example.com" IN {
  type master;
  file "example.com.zone";
  allow-transfer { 192.168.0.2; };
```

```
};
```

セカンダリサーバーの **zone** ステートメントは少し異なります。タイプは **slave** にセットされて、**masters** 指示文は **named** にマスターサーバーの IP アドレスを伝えています。

例14.6「セカンダリネームサーバー用のゾーンステートメント」 では、**named** サービスは、**192.168.0.1** IP アドレスにあるプライマリサーバーに対して **example.com** ゾーンの情報についてクエリをするように設定されています。受信した情報はその後、**/var/named/slaves/example.com.zone** ファイルに保存されます。すべてのスレーブゾーンを **/var/named/slaves** ディレクトリに配置することに注意して下さい。そうしないとサービスはゾーンの転送に失敗します。

例14.6 セカンダリネームサーバー用のゾーンステートメント

```
zone "example.com" {
  type slave;
  file "slaves/example.com.zone";
  masters { 192.168.0.1; };
};
```

14.2.1.2. その他のステートメントタイプ

以下のタイプのステートメントは、**/etc/named.conf** 内では通常多くは使用されません:

controls

controls ステートメントにより各種設定が可能になり、**named** サービスを管理するための **rndc** コマンドの使用に必要な様々なセキュリティ要件を設定できるようになります。

rndc ユーティリティとその使用法の詳細については、「rndc ユーティリティの使用」 を参照してください。

key

key ステートメントによって、特定のキーを名前で定義できるようになります。キーは、安全な更新や、あるいは、**rndc** コマンドの使用など各種動作を認証するために使用されます。２つのオプションが **key** と一緒に使用されます:

- **algorithm** *アルゴリズム名* — 使用されるアルゴリズムのタイプ (例えば、 **hmac-md5**)。
- **secret "***キー値***"** — 暗号化キーです。

rndc ユーティリティとその使用法の詳細については、「rndc ユーティリティの使用」 を参照してください。

logging

logging ステートメントにより、*チャンネル (channels)* と呼ばれる複数のログタイプを使用できるようになります。**channel** オプションをステートメント内で使用すると、それ自身のファイル名 (**file**)、サイズ制限 (**size**)、バージョン制御 (**version**)、及び重要度レベル (**severity**) などを持

つカスタマイズしたログのタイプを構築することが出来ます。カスタムチャンネルが定義されると、**category** オプションの使用でチャンネルが分類化されて、**named** サービスが再開始した時にロギングが始まります。

デフォルトでは、**named** は標準メッセージを **rsyslog** デーモンに送信して、受信したデーモンはメッセージを **/var/log/messages** に配置します。数種類の標準チャンネルが各種重要度レベルで BIND に組み込まれています。それらの重要度レベルとして、**default_syslog** (情報ロギングメッセージを処理) と**default_debug** (特にデバッギングメッセージを処理) などがあります。**default** と呼ばれるデフォルトカテゴリは組み込み型チャンネルを使用して特別な設定無しで普通のロギングを行います。

ロギングプロセスのカスタマイズは入り込んだプロセスとなるため、この章の範囲外になります。カスタム BIND ログ作成の詳細については、「インストールされているドキュメント」 で参照してある 『BIND 9 Administrator Reference Manual (BIND 9 管理者リファレンスマニュアル)』 をご覧下さい。

server

server ステートメントにより、**named** サービスがリモートのネームサーバーに対しての反応の仕方に影響する、特に通知とゾーン転送に関して影響するオプションを指定できるようになります。

transfer-format オプションは、各メッセージと共に送信されるリソースレコードの数量を制御します。これは、**one-answer** (1 つのリソースレコードのみ)、かまたは **many-answers** (複数のリソースレコード) のいずれかになります。**many-answers** オプションがより効率的ですが、BIND の旧バージョンではサポートされていないことに注意して下さい。

trusted-keys

trusted-keys ステートメントにより、安全な DNS (DNSSEC) 用に使用される各種パブリックキーを指定できるようになります。このトピックの詳細については、「DNSSEC (DNS Security Extensions)」 を参照して下さい。

view

view ステートメントにより、ネームサーバーをクエリしているホストが存在するネットワークに応じて特別な表示を作成できるようになります。これによって、一部のホストはゾーンに関して 1 つの応答を受信することが出来て、他のホストは全く異なる情報を受信することが出来るようになります。別の観点から、一定のゾーンは特定の信頼されるホストにだけ利用可能であり、信頼できないホストは他のゾーンのクエリしかできない可能性もあります。

複数表示はその名前が特有である限り使用できます。**match-clients** オプションにより、特定の表示に適用する IP アドレスを指定できるようになります。**options** ステートメントが 1つの表示内で使用された場合は、既に設定済みのグローバルオプションを上書きします。最後に、ほとんどの **view** ステートメントは、**match-clients** 一覧に適用される複数の **zone** ステートメントを含んでいます。

view ステートメント群が一覧表示されている順序が重要になることに注意して下さい。特定クライアントの IP アドレスにマッチする最初のステートメントが使用されます。このトピックの詳細については、「複数表示」 を参照して下さい。

14.2.1.3. コメントタグ

ステートメントに加えて、**/etc/named.conf** ファイルはコメントも含んでいます。コメントは **named** サービスには無視されますが、ユーザーに追加情報を提供する時に役に立ちます。以下に有効なコメントタグを示します:

`//`

`//` 文字の後のテキストはいずれもその行末までコメントと考慮されます。例えば:

```
notify yes;  // すべてのセカンダリネームサーバーに通知します
```

`#`

`#` 文字の後のテキストはいずれもその行末までコメントと考慮されます。例えば:

```
notify yes;  # すべてのセカンダリネームサーバーに通知します
```

`/*` と `*/`

`/*` と `*/` によって囲まれたテキストのブロックはコメントと考慮されます。例えば:

```
notify yes;  /* すべてのセカンダリネームサーバーに通知します */
```

14.2.2. ゾーンファイルの編集

「ネームサーバーのゾーン」 に要約してあるように、ゾーンファイルにはネームスペースの情報が含まれています。情報はデフォルトで **/var/named/** にある **named** 作業ディレクトリに保存されます。そして各ゾーンファイルは **zone** ステートメント内の **file** オプションに従って命名されます。通常は、**example.com.zone** などのように該当するドメインに関連しながら、ゾーンデータを含むファイルとして識別できる方法で命名されます。

表14.5 named サービスのゾーンファイル

パス	詳細
/var/named/	**named** サービスの作業ディレクトリです。ネームサーバーにはこのディレクトリに書き込む許可が **ありません。**
/var/named/slaves/	セカンダリゾーンのディレクトリです。このディレクトリは **named** サービスによる書き込みが可能です。
/var/named/dynamic/	動的 DNS (DDNS) ゾーン、または管理された DNSSEC キーなどの他のファイル用のディレクトリです。このディレクトリは **named** サービスによる書き込みが可能です。
/var/named/data/	様々な統計とデバッギングファイル用のディレクトリです。このディレクトリは **named** サービスによる書き込みが可能です。

ゾーンファイルは指示文とリソースの記録で構成されています。指示文はネームサーバーに対してタスクを実行するか、または特別なセッティングをゾーンに適用するように指示し、リソースレコードはゾーンのパラメータを定義して識別子を個々のホストに割り当てます。指示文はオプションですが、リソースレコードはゾーンにネームサービスを提供するために必須です。

指示文とリソースレコードはすべて、個々の行内に入れる必要があります。

14.2.2.1. 一般的な指示文

指示文はドルマーク文字で始まり、その後に指示文の名前が続きます。そして通常、ファイルの最上部に現れます。以下の指示文は一般的にゾーンファイルで使用されます:

$INCLUDE

$INCLUDE 指示文により、それが出現する場所にもう 1 つのファイルを含めることができるため、他のゾーンセッティングは別個のゾーンファイルに保存できるようになります。

例14.7 $INCLUDE 指示文の使用

```
$INCLUDE /var/named/penguin.example.com
```

$ORIGIN

$ORIGIN 指示文により、ホスト名だけの非完全修飾型の記録へドメイン名を追記できるようになります。ゾーン名はデフォルトで使用されるため **/etc/named.conf** 内でゾーンが指定されている場合は、この指示文の使用は必要ありません。

例14.8「$ORIGIN 指示文の使用」 では、リソースレコード内で使用される名前でトレーリングピリオドで終了していない名前はいずれも **example.com** が追記されています。

例14.8 $ORIGIN 指示文の使用

```
$ORIGIN example.com.
```

$TTL

$TTL 指示文により、ゾーン用のデフォルト *TTL* (Time to Live) 値をセットできるようになります。即ちゾーン記録が有効である時間の長さのセッティングです。各リソースレコードはそれ自身のTTL値を含むことができるため、それがこの指示文を上書きします。

この値を増加させることにより、リモートのネームサーバーはより長い期間でゾーン情報をキャッシュ化することが出きるため、ゾーンへのクエリ回数が減少でき、リソースレコード変更の伝達に必要な時間を延長させることができます。

例14.9 $TTL 指示文の使用

```
$TTL 1D
```

14.2.2.2. 一般的なリソースレコード

以下のリソースレコードは一般的にゾーンファイル内で使用されます:

A

Address レコードは名前に割り当てられる IP アドレスを指定します。以下の形式を取ります:

```
ホスト名 IN A IP-アドレス
```

hostname の値が欠如している場合、レコードは最後に指定された *hostname* を指します。

例14.10「A ソースレコードの使用」 では、**server1.example.com** 用の要求は、**10.0.1.3** または **10.0.1.5** を指しています。

例14.10 A ソースレコードの使用

```
server1  IN  A  10.0.1.3
         IN  A  10.0.1.5
```

CNAME

Canonical Name (別名) レコードは 1 つの名前を別の名前にマップします。このため、このタイプのレコードは時々、*エイリアスレコード*と呼ばれています。以下の形式を取ります:

```
エイリアス名 IN CNAME 本当の名前
```

CNAME レコードは Web サーバー用の **www** のように、共通の命名基準を使用するサービスを指すために最も一般的に使用されます。しかし、それらの使用については複数の制限が存在します:

- CNAME レコードは他の CNAME レコードを指してはいけません。これは主に無限のループの可能性を避けるためです。
- CNAME レコードは他のリソースレコードタイプ (例えば、A, NS, MX, など) を含んではいけません。唯一の例外は、ゾーンが署名されている時の DNSSEC 関連のレコード (即ち、RRSIG, NSEC など) です。
- ホストの完全修飾型ドメイン名 (FQDN) を指す他のリソースレコード (即ち、NS, MX, PTR) は CNAME レコードを指してはいけません。

例14.11「CNAME リソースレコードの使用」 では、**A** レコードがホスト名を IP アドレスにバインドして、**CNAME** レコードが一般的に使用される **www** ホスト名をそれに対して指しています。

例14.11 CNAME リソースレコードの使用

```
server1  IN  A      10.0.1.5
www      IN  CNAME  server1
```

MX

Mail Exchange レコードは、このゾーンによって制御されている特定のネームスペースに送信されるメールの行き先を指定します。以下の形式を取ります:

```
IN MX preference-value email-server-name
```

email-server-name は完全修飾型ドメイン名 (FQDN) です。*preference-value* によってネームスペースのメールサーバーの数値ランキングが可能になり、一部のメールシステムに他のシステムよりも優先度を与えます。最小の *preference-value* を持つ **MX** リソースレコードが他よりも優先されます。

しかし複数メールサーバーが同じ値を持つ可能性があり、その場合はメールトラフィックをサーバー間で均等に分配することになります。

例14.12「MX リソースレコードの使用」では、**example.com** ドメイン宛のメール受信時には最初の **mail.example.com** メールサーバーが **mail2.example.com** メールサーバーよりも優先されます。

例14.12 MX リソースレコードの使用

```
example.com.  IN  MX  10  mail.example.com.
              IN  MX  20  mail2.example.com.
```

NS

Nameserver レコードはある特定のゾーン用に正当なネームサーバーを表明します。以下の形式を取ります:

```
IN NS nameserver-name
```

nameserver-name は完全修飾型ドメイン名 (FQDN) である必要があります。ドメインに対して 2 つのネームサーバーが正当だとして一覧表示されている時には、これらのネームサーバーがセカンダリネームサーバーであるか、またはその 1 つがプライマリサーバーであるかどうかは重要でありません。両方とも正当と考慮されます。

例14.13 NS リソースレコードの使用

```
IN  NS  dns1.example.com.
IN  NS  dns2.example.com.
```

PTR

Pointer レコードはネームスペースの別の部分を指します。以下の形式を取ります:

```
last-IP-digit IN PTR FQDN-of-system
```

last-IP-digit 指示文は IP アドレスの末尾の番号です。そして*FQDN-of-system* は完全修飾型ドメイン名 (FQDN) です。

PTR レコードは主に逆引き名前解決に使用されます。これは逆に IP アドレスから特定の名前を指します。**PTR** レコードの使用例については 「逆引き名前解決ゾーンファイル」 を参照して下さい。

SOA

Start of Authority レコードはネームスペースについての信頼できる重要な情報をネームサーバーに表明します。指示文の後に配置されていて、ゾーンファイルでは最初のリソースレコードです。以下の形式を取ります

```
@  IN  SOA  primary-name-server hostmaster-email (
        serial-number
        time-to-refresh
```

```
        time-to-retry
        time-to-expire
        minimum-TTL  )
```

指示文は以下の通りです:

- @ シンボルは **$ORIGIN** 指示文 (または**$ORIGIN** 指示文がセットされていない場合は、ゾーン名) をこの**SOA** リソースレコードで定義されたネームスペースとして配置します。
- *primary-name-server* 指示文は、このドメイン用に正式となるプライマリネームサーバーのホスト名です。
- *hostmaster-email* 指示文は、ネームスペースに関して連絡する相手のメールです。
- *serial-number* 指示文は、**named** サービスがゾーンを再ロードする時間であることを示すためにゾーンファイルが変更される度に増加する数値です。
- *time-to-refresh* 指示文は、ゾーンに対して変更がなされたかどうかをプライマリネームサーバーに尋ねるまで待機する時間の長さを決定するためにセカンダリネームサーバーが使用する数値です。
- *time-to-retry* 指示文は、プライマリネームサーバーが応答しない事態にリフレッシュ要求を出すまで待機する時間の長さを決定するためにセカンダリネームサーバーによって使用される数値です。*time-to-expire* 指示文内で指定された時間が経過するまでに、プライマリネームサーバーがリフレッシュ要求に応答しない場合は、セカンダリサーバーはそのネームスペースに関する要求での権威としての応答を停止します。
- BIND 4 と 8 では、*minimum-TTL* 指示文は他のネームサーバーがゾーンの情報をキャッシュ化する時間の長さです。BIND 9 では、これは否定的な回答がキャッシュ化される時間の長さを定義します。否定的回答のキャッシュ化は最大で 3 時間にセットできます (即ち **3H**)。

BIND を設定する時、すべての時間は秒で指定されます。しかし、秒以外の時間単位を指定するのに短縮形を使用することができます。例えば、分 (**M**)、時間 (**H**)、日 (**D**)、及び週 (**W**) とします。表 14.6「秒表示とその他の時間単位」 では、時間数を秒で示し、更にはその同時間を他の形式で示しています。

表14.6 秒表示とその他の時間単位

秒	他の時間単位
60	**1M**
1800	**30M**
3600	**1H**
10800	**3H**
21600	**6H**
43200	**12H**
86400	**1D**

秒	他の時間単位
259200	**3D**
604800	**1W**
31536000	**365D**

例14.14 SOA リソースレコードの使用

```
@  IN  SOA  dns1.example.com.  hostmaster.example.com. (
        2001062501  ; serial
        21600       ; refresh after 6 hours
        3600        ; retry after 1 hour
        604800      ; expire after 1 week
        86400 )     ; minimum TTL of 1 day
```

14.2.2.3. コメントタグ

リソースレコードと指示文の他にも、ゾーンファイルもコメントを格納することができます。コメントは **named** サービスでは無視されますが、ユーザーに追加情報を提供する際に便利です。セミコロンの後の行末までのテキストはすべてコメントとみなされます。例えば:

```
    604800  ; expire after 1 week
```

14.2.2.4. 使用法の例

下記の例は、ゾーンファイルの基本的使用法を示したものです。

14.2.2.4.1. 単純なゾーンファイル

例14.15「単純なゾーンファイル」 では、標準の指示文と**SOA** 値の使用を提示しています。

例14.15 単純なゾーンファイル

```
$ORIGIN example.com.
$TTL 86400
@         IN  SOA  dns1.example.com.  hostmaster.example.com. (
              2001062501  ; serial
              21600       ; refresh after 6 hours
              3600        ; retry after 1 hour
              604800      ; expire after 1 week
              86400 )     ; minimum TTL of 1 day
;
;
          IN  NS     dns1.example.com.
          IN  NS     dns2.example.com.
dns1      IN  A      10.0.1.1
          IN  AAAA   aaaa:bbbb::1
```

```
dns2      IN  A      10.0.1.2
          IN  AAAA   aaaa:bbbb::2
;
;
@         IN  MX     10  mail.example.com.
          IN  MX     20  mail2.example.com.
mail      IN  A      10.0.1.5
          IN  AAAA   aaaa:bbbb::5
mail2     IN  A      10.0.1.6
          IN  AAAA   aaaa:bbbb::6
;
;
; This sample zone file illustrates sharing the same IP addresses
; for multiple services:
;
services  IN  A      10.0.1.10
          IN  AAAA   aaaa:bbbb::10
          IN  A      10.0.1.11
          IN  AAAA   aaaa:bbbb::11

ftp       IN  CNAME  services.example.com.
www       IN  CNAME  services.example.com.
;
;
```

この例では、正式なネームサーバーは **dns1.example.com** と **dns2.example.com** としてセットされており、**A** を使用して **10.0.1.1** と **10.0.1.2** の IP アドレスに結合されています。

MX レコードで設定されているメールサーバーは、**A** レコードを介して **mail** と **mail2** に向けられています。これらの名前はトレーリングピリオドで終了していないため、**$ORIGIN** ドメインがその後に配置されており、それらを **mail.example.com** および **mail2.example.com** に広げています。

www.example.com (WWW) などの標準の名前で利用可能なサービスは、**CNAME** レコードを使用して適切なサービスを指すようにしてあります。

このゾーンファイルは、**/etc/named.conf** 内で **zone** ステートメントを持つサービスに呼ばれることになります。以下に似ています:

```
zone "example.com" IN {
  type master;
  file "example.com.zone";
  allow-update { none; };
};
```

14.2.2.4.2. 逆引き名前解決ゾーンファイル

逆引き名前解決ゾーンファイルは、特定のネームスペース内の IP アドレスを完全修飾型ドメイン名 (FQDN) に解決するのに使用されます。例14.16「逆引き名前解決ゾーンファイル」 に示してあるように、これは標準のゾーンファイルに非常に似ていますが、**PTR** リソースレコードは IP アドレスを完全修飾型ドメイン名にリンクするために使用されます。

例14.16 逆引き名前解決ゾーンファイル

```
$ORIGIN 1.0.10.in-addr.arpa.
$TTL 86400
@  IN  SOA  dns1.example.com.  hostmaster.example.com. (
        2001062501  ; serial
        21600       ; refresh after 6 hours
        3600        ; retry after 1 hour
        604800      ; expire after 1 week
        86400 )     ; minimum TTL of 1 day
;
@  IN  NS   dns1.example.com.
;
1  IN  PTR  dns1.example.com.
2  IN  PTR  dns2.example.com.
;
5  IN  PTR  server1.example.com.
6  IN  PTR  server2.example.com.
;
3  IN  PTR  ftp.example.com.
4  IN  PTR  ftp.example.com.
```

この例では、**10.0.1.1** から **10.0.1.6** までの IP アドレスは該当する完全修飾型ドメイン名を指しています。

このゾーンファイルは **/etc/named.conf** ファイル内の **zone** ステートメントを持つサービスに呼ばれることになります。以下に似ています:

```
zone "1.0.10.in-addr.arpa" IN {
  type master;
  file "example.com.rr.zone";
  allow-update { none; };
};
```

この例と標準の **zone** ステートメントとの間には少ししか相違がありませんが、ゾーン名は異なります。逆引き名前解決ゾーンは、IP アドレスの最初の３つのブロックが逆になっていて、その後に **.in-addr.arpa** が続く必要があります。これにより、逆引き名前解決ゾーンファイルで使用される IP 番号のシングルブロックがそのゾーンと関連付けされるようになります。

14.2.3. rndc ユーティリティの使用

rndc ユーティリティは、ローカルとリモートのマシンから **named** サービスを管理できるようにするコマンドラインツールです。その使用法は以下のようになります:

```
rndc [option...] command [command-option]
```

14.2.3.1. ユーティリティの設定

サービスへの権限の無いアクセスを防止するには、**named** は選択したポート (デフォルトでは、 **953**) をリッスンするように設定してあり、このサービスと**rndc** ユーティリティの両方で同じキーが使用されなければなりません。

表14.7 関連したファイル

パス	詳細
/etc/named.conf	**named** サービス用のデフォルト設定ファイル
/etc/rndc.conf	**rndc** ユーティリティ用のデフォルト設定ファイル
/etc/rndc.key	デフォルトキーの場所

rndc 設定は **/etc/rndc.conf** に配置されています。ファイルが存在しない場合は、ユーティリティは、**rndc-confgen -a** コマンドを使用してインストールプロセス中に自動的に生成されていて **/etc/rndc.key** にあるキーを使用します。

named サービスは、「その他のステートメントタイプ」 で説明してあるように、**/etc/named.conf** 設定ファイル内の **controls** ステートメントを使用して設定されます。このステートメントが無いと、ループバックアドレス (**127.0.0.1**) からの接続のみが許可されることになり、**/etc/rndc.key** にあるキーが使用されます。

このトピックの詳細については、「その他のリソース」 に一覧表示してある『BIND 9 Administrator Reference Manual (管理者参照マニュアル)』 と man ページを参照して下さい。

重要

権限の無いユーザーがサービスに制御コマンドを送信することを阻止するために、root のみが **/etc/rndc.key** ファイルの読み取りを許可されるようにして下さい:

```
~]# chmod o-rwx /etc/rndc.key
```

14.2.3.2. サービスステータスの確認

named サービスの現在の状態をチェックするには、以下のコマンドを使用します:

```
~]# rndc status
version: 9.7.0-P2-RedHat-9.7.0-5.P2.el6
CPUs found: 1
worker threads: 1
number of zones: 16
debug level: 0
xfers running: 0
xfers deferred: 0
soa queries in progress: 0
query logging is OFF
recursive clients: 0/0/1000
tcp clients: 0/100
server is up and running
```

14.2.3.3. 設定とゾーンの再ロード

設定ファイルとゾーンの両方を再ロードするには、シェルプロンプトで以下を入力します:

```
~]# rndc reload
server reload successful
```

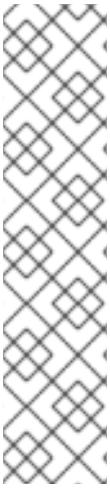
これがゾーンを再ロードすると同時に以前にキャッシュ化した応答を維持するため、すべての保存済みの名前解決を消失すること無くゾーンファイルを変更することができます。

単独ゾーンを再ロードするには、**reload** コマンドの後にその名前を指定します。例えば:

```
~]# rndc reload localhost
zone reload up-to-date
```

最後に、設定ファイルと新規に追加されたゾーンのみを再ロードするには、以下を入力します:

```
~]# rndc reconfig
```



注記

Dynamic (動的) DNS (DDNS) を使用するゾーンを手動で修正する計画がある場合は、必ず、 **freeze** コマンドを最初に実行するようにして下さい:

```
~]# rndc freeze localhost
```

それが終了すると、**thaw** コマンドを実行して DDNS を再度可能にしてゾーンを再ロードします:

```
~]# rndc thaw localhost
The zone reload and thaw was successful.
```

14.2.3.4. ゾーンキーの更新

DNSSEC キーを更新してゾーンに署名するには、**sign** コマンドを使用します。例えば:

```
~]# rndc sign localhost
```

上記のコマンドでゾーンに署名するには、ゾーンステートメント内で **maintain** に対して**auto-dnssec** オプションをセットしなければならないことに注意して下さい:

```
zone "localhost" IN {
  type master;
  file "named.localhost";
  allow-update { none; };
  auto-dnssec maintain;
};
```

14.2.3.5. DNSSEC 検証の有効化

DNSSEC 検証を有効にするには、シェルプロンプトで以下を入力します:

```
~]# rndc validation on
```

同様に、このオプションを無効にするには、以下を入力します:

```
~]# rndc validation off
```

/etc/named.conf 内でこのオプションを設定する方法についての情報には、「一般的なステートメントのタイプ」 で説明してある **options** ステートメントを参照して下さい。

14.2.3.6. クエリロギングの有効化

クエリロギングを有効にする (または、有効になっている時に無効にする) には、以下のコマンドを実行します:

```
~]# rndc querylog
```

現在のセッティングをチェックするには、「サービスステータスの確認」で説明してあるように**status** コマンドを使用します。

14.2.4. dig ユーティリティの使用

dig ユーティリティは、DNS ルックアップの実行とネームサーバー設定のデバッグができるようにするコマンドラインツールです。これは標準的に以下のように使用されます:

```
dig [@server] [option...] name type
```

一般的な *type* の一覧には 「一般的なリソースレコード」 を参照して下さい。

14.2.4.1. ネームサーバーのルックアップ

特定のドメイン用にネームサーバーをルックアップするには、以下の形式でコマンドを使用します:

```
digname NS
```

例14.17「ネームサーバールックアップのサンプル」 では、**dig** ユーティリティが **example.com** 用のネームサーバーを表示するために使用されています。

例14.17 ネームサーバールックアップのサンプル

```
~]$ dig example.com NS

; <<>> DiG 9.7.1-P2-RedHat-9.7.1-2.P2.fc13 <<>> example.com NS
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 57883
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;example.com.                   IN      NS

;; ANSWER SECTION:
example.com.            99374   IN      NS      a.iana-servers.net.
example.com.            99374   IN      NS      b.iana-servers.net.

;; Query time: 1 msec
```

```
;; SERVER: 10.34.255.7#53(10.34.255.7)
;; WHEN: Wed Aug 18 18:04:06 2010
;; MSG SIZE  rcvd: 77
```

14.2.4.2. IP アドレスのルックアップ

特定のドメインに割り当てられた IP アドレスをルックアップするには、以下の形式でコマンドを使用します:

```
dig name A
```

例14.18「IP アドレスルックアップのサンプル」 では、**example.com** の IP アドレスを表示する為に**dig** ユーティリティが使用されています。

例14.18 IP アドレスルックアップのサンプル

```
~]$ dig example.com A

; <<>> DiG 9.7.1-P2-RedHat-9.7.1-2.P2.fc13 <<>> example.com A
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4849
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:
;example.com.                   IN      A

;; ANSWER SECTION:
example.com.            155606  IN      A       192.0.32.10

;; AUTHORITY SECTION:
example.com.            99175   IN      NS      a.iana-servers.net.
example.com.            99175   IN      NS      b.iana-servers.net.

;; Query time: 1 msec
;; SERVER: 10.34.255.7#53(10.34.255.7)
;; WHEN: Wed Aug 18 18:07:25 2010
;; MSG SIZE  rcvd: 93
```

14.2.4.3. ホスト名のルックアップ

特定の IP アドレス用にホスト名をルックアップするには、以下の形式でコマンドを使用します:

```
dig
-x
address
```

例14.19「ホスト名ルックアップのサンプル」 では、**192.0.32.10** に割り当てられたホスト名を表示するために **dig** ユーティリティが使用されています。

例14.19 ホスト名ルックアップのサンプル

```
~]$ dig -x 192.0.32.10

; <<>> DiG 9.7.1-P2-RedHat-9.7.1-2.P2.fc13 <<>> -x 192.0.32.10
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 29683
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 5, ADDITIONAL: 6

;; QUESTION SECTION:
;10.32.0.192.in-addr.arpa.      IN      PTR

;; ANSWER SECTION:
10.32.0.192.in-addr.arpa. 21600 IN      PTR     www.example.com.

;; AUTHORITY SECTION:
32.0.192.in-addr.arpa.  21600   IN      NS      b.iana-servers.org.
32.0.192.in-addr.arpa.  21600   IN      NS      c.iana-servers.net.
32.0.192.in-addr.arpa.  21600   IN      NS      d.iana-servers.net.
32.0.192.in-addr.arpa.  21600   IN      NS      ns.icann.org.
32.0.192.in-addr.arpa.  21600   IN      NS      a.iana-servers.net.

;; ADDITIONAL SECTION:
a.iana-servers.net.     13688   IN      A       192.0.34.43
b.iana-servers.org.     5844    IN      A       193.0.0.236
b.iana-servers.org.     5844    IN      AAAA    2001:610:240:2::c100:ec
c.iana-servers.net.     12173   IN      A       139.91.1.10
c.iana-servers.net.     12173   IN      AAAA    2001:648:2c30::1:10
ns.icann.org.           12884   IN      A       192.0.34.126

;; Query time: 156 msec
;; SERVER: 10.34.255.7#53(10.34.255.7)
;; WHEN: Wed Aug 18 18:25:15 2010
;; MSG SIZE  rcvd: 310
```

14.2.5. BIND の高度な機能

ほとんどの BIND 実装は、名前解決サービスを提供するため、または特定のドメイン用の権威として機能するために **named** を使用するだけです。しかし、BIND バージョン 9 は数種の高度な機能を含んでおり、より安全で効率的な DNS サービスを可能にします。

重要

DNSSEC、TSIG、IXFR (増分ゾーン転送) などの高度な機能の使用を試みる前に、特に BIND の古いバージョンや BIND 以外のサーバーを使用している場合は、その特定の機能がネットワーク環境内のすべてのネームサーバーでサポートされていることを確認して下さい。

ここで言及されているすべての機能は 「インストールされているドキュメント」 内で参照されている『BIND 9 Administrator Reference Manual (管理者参照マニュアル)』 でより詳細に説明してあります。

14.2.5.1. 複数表示

オプションとして、要求発信元のネットワークに応じて異なる情報をクライアントに提供することができます。これは主に、ローカルネットワーク外のクライアントからの要注意 DNS エントリを拒否するために、そしてそれと同時にローカルネットワーク内のクライアントからのクエリを受け付けるために使用されます。

複数表示を設定するには、**view** ステートメントを **/etc/named.conf** 設定ファイルに追加します。**match-clients** オプションを使用して、IP アドレスか、またはネットワーク全体とマッチするようにしてから、それらに特別なオプションとゾーンデータを与えます。

14.2.5.2. IXFR (Incremental Zone Transfers 差分ゾーン転送)

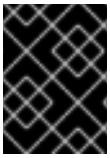
Incremental Zone Transfers 差分ゾーン転送 (*IXFR*) により、セカンダリネームサーバーはプライマリネームサーバー上で修正されたゾーンの更新部分だけをダウンロードすることができます。標準の転送プロセスに比較すると、これが通知と更新のプロセスを格段に効率的にします。

IXFR は、動的な更新を使用してマスターゾーンレコードに変更を加える時にのみ使用可能なことに注意して下さい。ゾーンファイルを手動の編集で変更する場合は、*Automatic Zone Transfer 自動ゾーン転送* (*AXFR*) が使用されます。

14.2.5.3. Transaction SIGnatures トランザクション署名 (TSIG)

Transaction SIGnatures トランザクション署名 (TSIG) は、転送を許可する前に共有の秘密鍵がプライマリとセカンダリの両方のサーバー上に存在することを確認します。これにより標準の IP アドレスベースの転送認証メソッドが強化されます。攻撃者は、ゾーンを転送するのに IP アドレスにアクセスする必要があるだけでなく、秘密鍵を知っている必要があります。

バージョン 9 からは、BIND は *TKEY* をサポートしますが、これはゾーン転送認証のもう 1 つの共有秘密鍵メソッドです。



重要

安全でないネットワーク上で通信している時には、IP アドレスベース認証のみに頼らないで下さい。

14.2.5.4. DNSSEC (DNS Security Extensions)

Domain Name System Security Extensions ドメイン名システムセキュリティ拡張 (*DNSSEC*) は、DNS データの発信元認証、認証による存在否定、及びデータ整合性を提供します。特定のドメインが安全としてマークされている時、検証に失敗した各リソースレコードに **SERFVAIL** 応答が返されます。

DNSSEC 署名のドメイン、または DNSSEC 認識のリゾルバーをデバッグするには、「dig ユーティリティの使用」 に説明してあるように、**dig** ユーティリティを使用することができます。役に立つオプションとして、**+dnssec** (DNSSEC OK キットを設定することで、DNSSEC 関連のリソースレコードを要求)、**+cd** (応答を検証しないように再帰的ネームサーバーに指示)、そして **+bufsize=512** (一部のファイヤーウォールを通過する為にパケットサイズを 512B に変更) があります。

14.2.5.5. インターネットプロトコルバージョン 6 (IPv6)

Internet Protocol version 6 インターネットプロトコルバージョン 6 (*IPv6*) は 表14.3「一般的に使用されるオプション」 に説明してあるように **AAAA** リソースレコードの使用、及び **listen-on-v6** 指示文を介してサポートされています。

14.2.6. 回避すべき一般的な間違い

ネームサーバー設定時にユーザーが一般的な間違いを回避する為のアドバイス一覧を以下に示します:

セミコロンと弓形括弧の正しい使用

/etc/named.conf ファイル内のセミコロンの欠如や、不一致な弓形括弧は **named** サービスの開始を阻止してしまいます。

ピリオドの正しい使用

ゾーンファイル内では、ドメイン名の末尾のピリオドは完全修飾型ドメイン名を示します。これが欠如していると、**named** サービスはゾーン名、または **$ORIGIN** の値を追記してそれを完結しようとします。

ゾーンファイルを編集する時のシリアル番号増加

シリアル番号が増加していない場合、プライマリネームサーバーは正しくて新しい情報を持ちますが、セカンダリネームサーバーは決して変更を通知されません。そのため、そのゾーンのデータをリフレッシュする試みをしません。

ファイヤーウォールの設定

ファイヤーウォールが、**named** サービスから他のネームサーバーへの接続をブロックしている場合は、ファイヤーウォールの設定変更が推奨されます。

警告

DNS セキュリティに関する最近の調査によると、DNS クエリのために固定の UDP ソースポートを使用することは、攻撃者がキャッシュポイゾニング攻撃をより簡単に実施できるようになるセキュリティ脆弱性を持つ可能性が出てきます。これを防止するためには、ランダムな UDP ソースポートからのクエリを許可するようにファイヤーウォールを設定します。

14.2.7. その他のリソース

以下の情報資料は、BIND に関するその他のリソースを提供します。

14.2.7.1. インストールされているドキュメント

BIND は、多種多様なトピックを網羅した広範囲に及ぶインストール済みのドキュメントを特徴としています。各ドキュメントはその議題のディレクトリ内に配置されています。以下の各項目には、*version* の部分をシステム上にインストールしてある bind パッケージのバージョンに入れ替えて下さい。

/usr/share/doc/bind-*version*/

最新のドキュメントを収納している主要ディレクトリ

/usr/share/doc/bind-*version*/arm/

このディレクトリは、HTML と SGML の形式で 『BIND 9 Administrator Reference Manual』 を収納しており、BIND のリソース要件、異種タイプのネームサーバーの設定方法、ロードバランシングの実行方法、及び他の高度なトピックを説明しています。ほとんどの BIND の新ユーザーにはこのディレクトリが最善の出発点でしょう。

/usr/share/doc/bind-*version*/draft/

このディレクトリは DNS サービスに関する問題点を再確認する各種技術ドキュメントを収納しており、問題に対応する方法を提案しています。

/usr/share/doc/bind-*version*/misc/

このディレクトリは特定の高度な問題に対応するために設計されています。BIND バージョン 8 のユーザーは、BIND 9 への移動時に実行すべき特定の変更に関する **migration** ドキュメントを確認する必要があります。**options** ファイルは、BIND 9 に実装されていて**/etc/named.conf** 内で使用されるすべてのオプションを一覧表示しています。

/usr/share/doc/bind-*version*/rfc/

BIND に関連したすべてのドキュメントを提供するディレクトリ

BIND に関わっている各種アプリケーションと設定ファイルのためにも多くの man ページが存在しています:

man rndc

rndc の man ページにはその使用法に関する総括的ドキュメントが含まれています。

man named

named の man ページには、BIND ネームサーバーデーモンを制御する為に使用できる各種引数についてのドキュメントが含まれています。

man lwresd

lwresd の man ページには、軽量のリゾルバーデーモンとその使用法についての総括的ドキュメントが含まれています。

man named.conf

この man ページには、**named** 設定ファイル内で利用可能なオプションの総合的一覧があります。

man rndc.conf

この man ページには、**rndc** 設定ファイル内で利用可能なオプションの総合的一覧があります。

14.2.7.2. 役立つ Web サイト

http://www.isc.org/software/bind

BIND プロジェクトのホームページには、『BIND 9 Administrator Reference Manual』 の PDF バージョンだけでなく、現在のリリースに関する情報が掲載されています。

14.2.7.3. 関連資料

『DNS and BIND』 (Paul Albitz、Cricket Liu 共著、O'Reilly & Associates 社出版)

BIND 設定オプションの一般的なものと難解なものの両方を説明し、DNS サーバーのセキュリティ戦略を提供する人気のある参照本です。

『The Concise Guide to DNS and BIND』 (Nicolai Langfeldt 著、Que 社出版)

タスク指向の技術トピックを強調しながら、複数ネットワークサービスと BIND の関連を考察しています。

第15章 WEB サーバー

HTTP (Hypertext Transfer Protocol ハイパーテキスト転送プロトコル) サーバー、または *Web サーバー* は、Web 上でクライアントにコンテンツを提供するネットワークサービスです。通常これは Web ページを意味しますが、他のドキュメントも同様に提供されます。

15.1. APACHE HTTP サーバー

このセクションでは、Apache Software Foundation によって開発された堅牢な全機能付きオープンソースウェブサーバーであり、Red Hat Enterprise Linux 6 に収納されている **Apache HTTP サーバー 2.2** に焦点を置いています。ここでは、**httpd** サービスの基本的な設定を説明すると共に、サーバーモジュールの追加、仮想ホストのセットアップ、及びセキュアなHTTP サーバーの設定を取り扱っています。

Apache HTTP サーバー 2.2 とバージョン 2.0 との間には重要な相違が存在します。そのため Red Hat Enterprise Linux の以前のバージョンからアップグレードする際には、**httpd** サービスの設定も同様に更新する必要があります。このセクションは、いくつかの新しい機能の追加を再確認して、重要な変更を概要し、旧来のファイルの更新についてガイドして行きます。

15.1.1. 新しい機能

Apache HTTP サーバーバージョン 2.2 は以下のような強化機能を導入しています:

- キャッシュモジュールの向上 **mod_cache** および **mod_disk_cache**。
- プロキシロードバランシングのサポート **mod_proxy_balancer** モジュール。
- 32-bit アーキテクチャ内での大容量ファイルのサポートにより、ウェブサーバーが 2GB 以上のファイルを処理できるようになります。
- 認証と権限付与サポートでの新構造が、以前のバージョンで提供されていた認証モジュールの入れ替えになります。

15.1.2. 注目すべき変更

バージョン 2.0 以降に、デフォルトの **httpd** サービス設定に変更が加えられています:

- 以下のモジュールは、デフォルトではもはやロードされません: **mod_cern_meta** および **mod_asis**。
- 以下のモジュールはデフォルトで新規にロードされます: **mod_ext_filter**。

15.1.3. 設定の更新

Apache HTTP サーバーバージョン 2.0 の設定ファイルを更新するには、以下の手順に従います:

1. モジュールは変更されている可能性があるため、すべてのモジュール名が正しいことを確認して下さい。改名されたモジュールそれぞれについて **LoadModule** 指示文を調節します。
2. サードパーティのモジュールはロードを試行する前にすべて再コンパイルをします。これは一般的に認証と権限付与のモジュールに該当します。
3. **mod_userdir** モジュールを使用する場合は、 ディレクトリ名 (通常 **public_html**) を示す **UserDir** ディレクティブが備わっていることを確認します。

4. Apache HTTP セキュアサーバーを使用する場合は、**/etc/httpd/conf.d/ssl.conf** を編集して Secure Sockets Layer (SSL) プロトコルを有効にします。

以下のコマンドを使用すると、設定のエラーの可能性をチェックすることができます:

```
~]# service httpd configtest
Syntax OK
```

Apache HTTP サーバーの設定をバージョン 2.0 から 2.2 に更新する方法の詳細については、http://httpd.apache.org/docs/2.2/upgrading.htmlを参照して下さい。

15.1.4. httpd サービスの実行

このセクションでは、Apache HTTP サーバーの開始、停止、再開始、及び現在のステータスのチェックの方法を説明しています。**httpd** サービスの使用を有効にするには、まず httpd がインストールされていることを確認します。これは以下のコマンドを実行することで達成できます:

```
~]# yum install httpd
```

ランレベルの概念と Red Hat Enterprise Linux 内のシステムサービスの管理法全般についての詳細には *10章 サービスとデーモン* を参照して下さい。

15.1.4.1. サービスの開始

httpd サービスを実行するには、シェルプロンプトで以下を入力します:

```
~]# service httpd start
Starting httpd:                                            [  OK  ]
```

ブート時にサービスを自動的に起動させたい場合には、以下のコマンドを入力します:

```
~]# chkconfig httpd on
```

これにより、ランレベル 2、3、4、5 でサービスが有効になります。別の方法では 「サービスの有効化と無効化」 の説明にある通り、**サービスの設定**ユーティリティを使用することもできます。

注記

Apache HTTP サーバーをセキュアサーバーとして実行している場合に、暗号化したプライベート SSL キーを使用するならば、マシンが起動した後にパスワードを要求される可能性があります。

15.1.4.2. サービスの停止

実行中の **httpd** サービスを停止するには、シェルプロンプトで以下を入力します:

```
~]# service httpd stop
Stopping httpd:                                            [  OK  ]
```

ブート時にサービスの自動開始を阻止したい場合には、以下を入力します:

```
~]# chkconfig httpd off
```

これにより、すべてのランレベルでサービスを無効にします。別の方法としては 「サービスの有効化と無効化」 の説明にあるように、**サービスの設定** ユーティリティを使用することもできます。

15.1.4.3. サービスの再開始

実行中の **httpd** サービスを再開始するには異なる 3 つの方法があります:

1. サービスを全面的に再開始するには、以下を入力します:

```
~]# service httpd restart
Stopping httpd:                                            [  OK
]
Starting httpd:                                            [  OK
]
```

このコマンドは先ず、実行中の **httpd** サービスを停止して、それからそれを再開始します。PHP などの動的にロードしたモジュールのインストール後や削除後にこのコマンドを使用します。

2. 設定を再ロードするだけの場合は、以下を入力します:

```
~]# service httpd reload
```

このコマンドは実行中の **httpd** サービスが設定ファイルを再ロードするようにします。注意すべき点は、現在プロセス中の要求はいずれも割り込みされることになり、クライアントのブラウザーがエラーメッセージを表示したり、ページの一部のみを描写する可能性があることです。

3. アクティブな要求に影響なく、設定ファイルを再ロードするには、以下を入力します:

```
~]# service httpd graceful
```

これにより、実行中の **httpd** サービスは設定ファイルを再ロードするようになります。この場合は、現在プロセス中の要求はいずれも古い設定を使用することになります。

別の方法では 「サービスの開始、再起動と停止」 の説明にあるように、**サービスの設定** ユーティリティを使用する方法もあります。

15.1.4.4. サービスステータスの確認

サービスが実行中であることを確認するには、シェルプロンプトで次のコマンドを入力します:

```
~]# service httpd status
httpd (pid 19014) is running...
```

他の方法として、「サービス設定ユーティリティの使用」で説明してあるように、**Service Configuration** ユーティリティを使用することもできます。

15.1.5. 設定ファイルの編集

httpd サービスは開始後にデフォルトで、表15.1「httpd サービスの設定ファイル」 に一覧表示してある場所から設定を読み込みます。

表15.1 httpd サービスの設定ファイル

パス	詳細
/etc/httpd/conf/httpd.conf	主要設定ファイル
/etc/httpd/conf.d/	主要設定ファイル内に含まれている設定ファイル用の補助ディレクトリ

デフォルトの設定がほとんどの状況に適合ますが、更に重要な設定オプションをいくつか知っておくと役に立ちます。どの変更でもその効果が反映されるには、まずサーバーを再開始する必要があることに注意して下さい。**httpd** サービスの再開始方法の詳細については 「サービスの再開始」 を参照して下さい。

起こり得るエラーの検知の為に設定をチェックするには、シェルプロンプトで以下を入力します:

```
~]# service httpd configtest
Syntax OK
```

失敗からの復旧をより簡単にするために、編集する前にオリジナルファイルのコピーを作成しておくことをお勧めします。

15.1.5.1. 一般的な httpd.conf の指示文

以下のような指示文が一般的に **/etc/httpd/conf/httpd.conf** 設定ファイルで使用されています:

<;Directory>

<;Directory> 指示文により、特定のディレクトリのみに特定の指示文を適用できるようになります。以下の形式を取ります:

```
<Directory ディレクトリ>
  指示文
  …
</Directory>
```

ディレクトリ はローカルファイルシステムの既存ディレクトリまでの完全パスか、またはワイルドカード表現になります。

この指示文は、**ScriptAlias** で指定されているディレクトリの外部にあるサーバー側の為の追加の **cgi-bin** ディレクトリを設定するのに使用されます。このケースでは、**ExecCGI** と **AddHandler** の指示文を供給する必要があり、そして目的地のディレクトリは正しくセットされていなければなりません (即ち、 **0755**)。

例15.1 <Directory> 指示文の使用

```
<Directory /var/www/html>
  Options Indexes FollowSymLinks
  AllowOverride None
  Order allow,deny
  Allow from all
</Directory>
```

<IfDefine>

IfDefine 指示文により、特定のパラメータがコマンドラインで示された時にのみ一定の指示文を使用できるようになります。以下の形式を取ります:

```
<IfDefine [!]パラメータ>
  指示文
  …
</IfDefine>
```

パラメータ は、**-D***parameter* コマンドラインオプションを使用して、シェルプロンプトで表示できます (例えば、 **httpd -DEnableHome**)。オプションの感嘆符 (即ち、 **!**) が存在するならば、囲まれた指示文は、パラメータが指定**されていない**時にのみ使用されます。

例15.2 <IfDefine> 指示文の使用

```
<IfDefine EnableHome>
  UserDir public_html
</IfDefine>
```

<IfModule>

<IfModule> 指示文により、特定のモジュールがロードされた時にのみ一定の指示文を使えるようになります。以下の形式を取ります:

```
<IfModule [!]モジュール>
  指示文
  …
</IfModule>
```

モジュール は、その名前、あるいはファイル名のどちらかで特定できます。オプションの感嘆符 (即ち、 **!**) が存在する場合は、囲まれた指示文はモジュールがロード**されていない** 時にのみ使用されます。

例15.3 <IfModule> 指示文の使用

```
<IfModule mod_disk_cache.c>
  CacheEnable disk /
  CacheRoot /var/cache/mod_proxy
</IfModule>
```

<Location>

<Location> 指示文により、特定の URL にのみ一定の指示文を適用できるようになります。以下の形式を取ります:

```
<Location url>
  指示文
  …
</Location>
```

url は、**DocumentRoot** 指示文で指定されているディレクトリへの相対パス (例えば、**/server-info**) か、あるいは **http://example.com/server-info** などの外部の URL のどちらかです。

例15.4 <Location> 指示文の使用

```
<Location /server-info>
  SetHandler server-info
  Order deny,allow
  Deny from all
  Allow from .example.com
</Location>
```

<Proxy>

<Proxy> 指示文により、プロキシサーバーにのみ一定の指示文を適用できるようになります。以下の形式を取ります

```
<Proxy パターン>
  指示文
  …
</Proxy>
```

パターン は外部の URL か、あるいは、ワイルドカード表現 (例えば、**http://example.com/***) になります

例15.5 <Proxy> 指示文の使用

```
<Proxy *>
  Order deny,allow
  Deny from all
  Allow from .example.com
</Proxy>
```

<VirtualHost>

<VirtualHost> 指示文により、特定の仮想ホストのみに一定の指示文を適用できるようになります。以下の形式を取ります:

```
<VirtualHost アドレス[:ポート]…>
  指示文
  …
</VirtualHost>
```

アドレス は IP アドレス、完全修飾型ドメイン名、あるいは 表15.2「利用可能な <VirtualHost> オプション」 に説明してある特殊な形式のいずれかになります。

表15.2 利用可能な <VirtualHost> オプション

オプション	詳細
*****	すべての IP アドレスを示します。
default	合致していない IP アドレスを示します。

例15.6 <VirtualHost> 指示文の使用

```
<VirtualHost *:80>
  ServerAdmin webmaster@penguin.example.com
  DocumentRoot /www/docs/penguin.example.com
  ServerName penguin.example.com
  ErrorLog logs/penguin.example.com-error_log
  CustomLog logs/penguin.example.com-access_log common
</VirtualHost>
```

AccessFileName

AccessFileName 指示文により、各ディレクトリ用のアクセス制御情報をカスタマイズするのに使用するファイルを指定できるようになります。以下の形式を取ります:

```
AccessFileName ファイル名…
```

ファイル名 は要求されたディレクトリ内で探すファイルの名前です。デフォルトでは、サーバーは **.htaccess** を探します。

セキュリティ目的で、指示文は標準的には **Files** タグを後に付けており、**.ht** で始まるファイルがウェブクライアントからアクセスされるのを阻止します。これには、**.htaccess** と **.htpasswd** のファイルが含まれます。

例15.7 AccessFileName 指示文の使用

```
AccessFileName .htaccess

<Files ~ "^\.ht">
  Order allow,deny
  Deny from all
  Satisfy All
</Files>
```

Action

Action 指示文により、一定のメディアタイプが要求された時に CGI スクリプトを実行するように指定できるようになります。以下の形式を取ります:

```
Action コンテンツタイプ パス
```

コンテンツタイプ は、**text/html**、**image/png**、または **application/pdf** のような有効な MIME タイプでなければなりません。*パス* は既存の CGI スクリプトを指していて、そし

て**DocumentRoot** 指示文で指定されたディレクトリに相対的でなければなりません (例えば、**/cgi-bin/process-image.cgi**)。

例15.8 Action 指示文の使用

```
Action image/png /cgi-bin/process-image.cgi
```

AddDescription

AddDescription 指示文により、該当するファイル用のサーバー生成されたディレクトリー覧内に短い説明表示を指定できるようになります。以下の形式を取ります:

```
AddDescription "説明" ファイル名…
```

説明 は二重引用符 (即ち **"**) に囲まれた短いテキストでなければなりません。*ファイル名* はファイルの全長名、ファイル拡張子、またはワイルドカード表現のいずれかにします。

例15.9 AddDescription 指示文の使用

```
AddDescription "GZIP compressed tar archive" .tgz
```

AddEncoding

AddEncoding 指示文の使用により、特定のファイル拡張子用のエンコーディングタイプを指定できるようになります。以下の形式を取ります:

```
AddEncoding エンコーディング 拡張子…
```

エンコーディング は、**x-compress** や **x-gzip** のような有効な MIME エンコーディングでなければなりません。*拡張子* は大文字/小文字を区別したファイル拡張子で、慣習的にドットが先に付きます (例えば、 **.gz**)。

この指示文は標準的にウェブブラウザーに対して、一定のファイルがダウンロードされる際にそれを展開するように指示します。

例15.10 AddEncoding 指示文の使用

```
AddEncoding x-gzip .gz .tgz
```

AddHandler

AddHandler 指示文の使用により、一定のファイル拡張子を選択したハンドラーにマップできるようになります。以下の形式を取ります:

```
AddHandler ハンドラー 拡張子…
```

ハンドラー は、以前に定義してあるハンドラーの名前にする必要があります。*拡張子* は大文字/小文字の区別があるファイル拡張子で、慣習的に先頭にドットが付きます (例えば、 **.cgi**)。

この指示文は標準的に、ファイルのあるディレクトリに関係なく、**.cgi** 拡張子の付いたファイルを CGI スクリプトとして扱うのに使用されます。更には、この指示文は一般的にサーバー解析した HTML ファイルとイメージマップファイルのプロセスにも使用されます。

例15.11 AddHandler オプションの使用

```
AddHandler cgi-script .cgi
```

AddIcon

AddIcon 指示文の使用により、サーバー生成のディレクトリー覧で特定ファイル用にアイコンを表示する指定ができるようになります。以下の形式を取ります:

```
AddIcon パス パターン…
```

パス は既存のアイコンファイルを指しており、そして**DocumentRoot** 指示文 (例えば、**/icons/folder.png**) で指定してあるティレクトリに相対的でなければなりません。*パターン* は、ファイル名、ファイル拡張子、ワイルドカード表現、または以下の表に説明してある特殊な形式のいずれかになります:

表15.3 利用可能な AddIcon オプション

オプション	詳細
^^DIRECTORY^^	ディレクトリを示します。
^^BLANKICON^^	空白の行を示します。

例15.12 AddIcon 指示文の使用

```
AddIcon /icons/text.png .txt README
```

AddIconByEncoding

AddIconByEncoding 指示文の使用により、サーバー生成のディレクトリー覧で特定エンコーディングタイプ用にアイコンの表示を指定できるようになります。以下の形式を取ります

```
AddIconByEncoding パス エンコーディング…
```

パス は既存のアイコンファイルを指しており、**DocumentRoot** 指示文 (例えば、**/icons/compressed.png**) で指定してあるディレクトリに相対的でなければなりません。*エンコーディング* は、**x-compress** や、**x-gzip** などの有効な MIME でなれればなりません。

例15.13 AddIconByEncoding 指示文の使用

```
AddIconByEncoding /icons/compressed.png x-compress x-gzip
```

AddIconByType

AddIconByType 指示文の使用により、サーバー生成のディレクトリー一覧で特定のメディア用にアイコンを表示する指定ができるようになります。以下の形式を取ります:

```
AddIconByType パス コンテンツタイプ…
```

パス は既存のアイコンファイルを指しており、**DocumentRoot** 指示文 (例えば、**/icons/text.png**) で指定してあるディレクトリに相対的でなければなりません。*コンテンツタイプ* は有効な MIME タイプ (例えば、**text/html** や **image/png**) であるか、または **text/*** や **image/*** のようなワイルドカード表現のどちらかになります。

例15.14 AddIconByType 指示文の使用

```
AddIconByType /icons/video.png video/*
```

AddLanguage

AddLanguage 指示文の使用で、ファイル拡張子を特定の言語に関連付けすることができます。以下の形式を取ります:

```
AddLanguage 言語 拡張子…
```

言語 は **cs** や、**en** や、**fr** などの有効な MIME 言語でなければなりません。*拡張子* は大文字/小文字の区別のあるファイル拡張子であり、慣習的に先頭にドットが付きます (例えば、**.cs**)。

この指示文は、クライアントの言語設定に基づいて複数の言語でコンテンツを提供するようなウェブサーバーに特に役に立ちます。

例15.15 AddLanguage 指示文の使用

```
AddLanguage cs .cs .cz
```

AddType

AddType 指示文の使用により、特定のファイル拡張子用にメディアタイプを定義、または上書きできるようになります。以下の形式を取ります:

```
AddType コンテンツタイプ 拡張子…
```

コンテンツタイプ は **text/html** や、**image/png** などのような有効な MIME タイプでなければなりません。*拡張子* は大文字/小文字の区別のあるファイル拡張子であり、慣習的に先頭にドットが付きます (例えば、**.cs**)。

例15.16 AddType 指示文の使用

```
AddType application/x-gzip .gz .tgz
```

Alias

Alias 指示文の使用により、**DocumentRoot** 指示文で指定されているデフォルトのディレクトリ外のファイルやディレクトリを参照できるようになります。以下の形式を取ります:

```
Alias url-パス real-パス
```

url-パス は、**DocumentRoot** 指示文で指定されているディレクトリに相対的でなければなりません (例えば、**/images/**)。*real-パス* は、ローカルファイルシステム内のファイルかディレクトリまでの全長のパスです。

この指示文は標準的に末尾に **Directory** タグを持ち、ターゲットディレクトリにアクセスするための追加の権限を付帯しています。デフォルトでは、**/icons/** エイリアスが作成されるため、**/var/www/icons/** のアイコンはサーバー生成のディレクトリー覧内に表示されます。

例15.17 Alias 指示文の使用

```
Alias /icons/ /var/www/icons/

<Directory "/var/www/icons">
  Options Indexes MultiViews FollowSymLinks
  AllowOverride None
  Order allow,deny
  Allow from all
<Directory>
```

Allow

Allow 指示文を使用すると、任意のディレクトリにアクセス権限を持つクライアントを指定できるようになります。以下の形式を取ります:

```
Allow from クライアント…
```

クライアント は、ドメイン名、IP アドレス (全長と部文の両方)、及び *ネットワーク*/*ネットマスク* ペアか、またはすべてのクライアント用の **all** となります。

例15.18 Allow 指示文の使用

```
Allow from 192.168.1.0/255.255.255.0
```

AllowOverride

AllowOverride 指示文の使用により、デフォルト設定を上書きできる **.htaccess** ファイル内の指示文を指定できるようになります。以下の形式をとります:

```
AllowOverride タイプ…
```

タイプ は、表15.4「利用可能な AllowOverride オプション」 で説明してあるように利用可能なグループオプションの 1 つでなければなりません。

表15.4 利用可能な AllowOverride オプション

オプション	詳細
All	**.htaccess** 内の指示文はすべて以前の設定構成を上書きする権限を持ちます。
None	**.htaccess** 内の指示文はどれも以前の設定構成を上書きする権限を持ちません。
AuthConfig	**AuthName**、**AuthType**、**Require** など権限付与の指示文の使用を可能にします。
FileInfo	**DefaultType**、**RequestHeader**、**RewriteEngine** などや、更には **Action** 指示文など、ファイルタイプとメタデータ、及び**mod_rewrite** 指示文の使用が可能になります。
Indexes	**AddDescription**、**AddIcon**。あるいは**FancyIndexing** など、ディレクトリ指標指示文の使用が可能になります。
Limit	**Allow**、**Deny**、及び **Order** などホストアクセス指示文の使用が可能になります。
Options [=*オプション*,...]	**Options** 指示文の使用を可能にします。更には、この指示文を使用してセットするオプションをカスタマイズするためにカンマで隔離したオプションの一覧を作成することもできます。

例15.19 AllowOverride 指示文の使用

```
AllowOverride FileInfo AuthConfig Limit
```

BrowserMatch

BrowserMatch 指示文の使用により、クライアントのウェブブラウザーを基にしてサーバーの動作を修正することができるようになります。以下の形式を取ります。

```
BrowserMatch パターン 変数…
```

パターン は、User-Agent HTTP ヘッダーフィールドにマッチするための正規表現です。*変数* はヘッダーフィールドがパターンにマッチした時にセットされる環境変数です。

デフォルトでは、この指示文は既知の問題を持つ特定のブラウザーへの接続を拒否して、keepalive と HTTP header flush で問題を持つと思われるブラウザーでそれらのアクションを無効にするために使用されます。

例15.20 BrowserMatch 指示文の使用

```
BrowserMatch "Mozilla/2" nokeepalive
```

CacheDefaultExpire

CacheDefaultExpireオプションの使用で、失効日がないか、または最後の修正日の指定がないドキュメントをキャッシュ化する期間をセットできるようになります。以下の形式を取ります:

```
CacheDefaultExpire 時間
```

時間 は、秒で指定します。デフォルトのオプションは、**3600** (即ち 1 時間) です。

例15.21 CacheDefaultExpire 指示文の使用

```
CacheDefaultExpire 3600
```

CacheDisable

CacheDisable 指示文の使用により、一定の URL のキャッシュ化を無効にできるようになります。以下の形式を取ります:

```
CacheDisable パス
```

パス は **DocumentRoot** 指示文で指定してあるディレクトリに相対的でなければなりません (例えば、**/files/**)。

例15.22 CacheDisable 指示文の使用

```
CacheDisable /temporary
```

CacheEnable

CacheEnable 指示文の使用により、一定の URL に使用するキャッシュタイプを指定できるようになります。以下の形式を取ります:

```
CacheEnable タイプ url
```

タイプ は 表15.5「利用可能なキャッシュタイプ」 で説明してあるように、有効なキャッシュタイプでなければなりません。*url* は **DocumentRoot** 指示文で指定されたディレクトリに相対的なパス (例えば、**/images/**) か、プロトコル (例えば、**ftp://**) か、**http://example.com/** のような外部の URL になります。

表15.5 利用可能なキャッシュタイプ

タイプ	詳細
mem	メモリーベースのストレージマネージャです。
disk	ディスクベースのストレージマネージャです。
fd	ファイルディスクリプタキャッシュ

例15.23 CacheEnable 指示文の使用

```
CacheEnable disk /
```

CacheLastModifiedFactor

CacheLastModifiedFactor 指示文の使用により、失効日が指定されていないものの、最後の修正日に関する情報を提供するドキュメントのキャッシュ化期間をカスタマイズできるようになります。以下の形式を取ります

```
CacheLastModifiedFactor 数値
```

数値 はドキュメントの最後の修正日から経過した時間を倍増する為の指数です。デフォルトオプションでは **0.1** (即ち、10分の 1) です。

例15.24 CacheLastModifiedFactor 指示文の使用

```
CacheLastModifiedFactor 0.1
```

CacheMaxExpire

CacheMaxExpire 指示文の使用で、ドキュメントのキャッシュ化の期間の最大値を指定できるようになります。以下の形式を使用します:

```
CacheMaxExpire 時間
```

時間 は秒で指定します。デフォルトのオプションは **86400** (即ち 1 日) です。

例15.25 CacheMaxExpire 指示文の使用

```
CacheMaxExpire 86400
```

CacheNegotiatedDocs

CacheNegotiatedDocs 指示文の使用により、コンテンツに基づいたネゴシエートをしたドキュメントのキャッシュ化を有効にできるようになります。以下の形式を取ります:

```
CacheNegotiatedDocs オプション
```

オプション は、表15.6「利用可能な CacheNegotiatedDocs オプション」 で説明してあるように、有効なキーワードでなければなりません。コンテンツに関してネゴシエートしたドキュメントは時間経過で、または要求者からの入力で変化するため、デフォルトのオプションは **Off** です。

表15.6 利用可能な CacheNegotiatedDocs オプション

オプション	詳細
On	コンテンツネゴシエート済みドキュメントのキャッシュ化を有効にします。

オプション	詳細
Off	コンテンツネゴシエート済みドキュメントのキャッシュ化を無効にします。

例15.26 CacheNegotiatedDocs 指示文の使用

```
CacheNegotiatedDocs On
```

CacheRoot

CacheRoot 指示文の使用により、キャッシュファイルを格納するディレクトリを指定できるようになります。以下の形式を取ります:

```
CacheRoot ディレクトリ
```

ディレクトリ は、ローカルシステムにある既存ディレクトリの全長パスでなければなりません。デフォルトのオプションは **/var/cache/mod_proxy/** です。

例15.27 CacheRoot 指示文の使用

```
CacheRoot /var/cache/mod_proxy
```

CustomLog

CustomLog 指示文の使用により、ログファイル名とログファイル形式を指定できるようになります。以下の形式を取ります:

```
CustomLog パス 形式
```

パス はログファイルを指しており、**ServerRoot** 指示文で指定されたディレクトリ (即ち、デフォルトでは **/etc/httpd/**) に相対的でなければなりません。*形式* は、明示的な形式の文字列か、または **LogFormat** 指示文の使用で以前に定義されている形式名になります。

例15.28 CustomLog 指示文の使用

```
CustomLog logs/access_log combined
```

DefaultIcon

DefaultIcon 指示文の使用により、他に関連したアイコンがない場合にサーバー生成のディレクトリ一覧で、ファイルにアイコンが表示されるように指定することができるようになります。以下の形式を取ります:

```
DefaultIcon パス
```

パス は既存のアイコンファイルを指しており、**DocumentRoot** 指示文で指定されたディレクトリ (例えば、**/icons/unknown.png**) に相対的でなければなりません。

例15.29 DefaultIcon 指示文の使用

```
DefaultIcon /icons/unknown.png
```

DefaultType

DefaultType 指示文の使用で、サーバーが適格な MIME タイプを決定できない場合に使用するメディアタイプを指定できるようになります。以下の形式を取ります:

```
DefaultType コンテンツタイプ
```

コンテンツタイプ は、**text/html**、**image/png**、**application/pdf** などの有効な MIME タイプでなければなりません。

例15.30 DefaultType 指示文の使用

```
DefaultType text/plain
```

Deny

Deny 指示文の使用により、任意のディレクトリにアクセスを拒否されるクライアントを指定できるようになります。以下の形式を取ります:

```
Deny from クライアント…
```

クライアント は、ドメイン名、IP アドレス (全長と部文の両方)、及び *ネットワーク*/*ネットマスク* ペアか、またはすべてのクライアント用の **all** となります。

例15.31 Deny 指示文の使用

```
Deny from 192.168.1.1
```

DirectoryIndex

DirectoryIndex 指示文の使用で、ディレクトリが要求された時 (即ち、URL が / 文字で終了する時) にクライアントに対して提供するドキュメントを指定できるようになります。以下の形式を取ります:

```
DirectoryIndex ファイル名…
```

ファイル名 は、要求されたディレクトリ内で見つけるファイルの名前です。デフォルトでは、サーバーは **index.html** と **index.html.var** を探します。

例15.32 DirectoryIndex 指示文の使用

```
DirectoryIndex index.html index.html.var
```

DocumentRoot

DocumentRoot 指示文の使用により、コンテンツの提供元となる主要ディレクトリを指定できるようになります。以下の形式を取ります:

```
DocumentRoot ディレクトリ
```

ディレクトリ は、ローカルファイルシステム内にある既存ディレクトリへの全長パスでなければなりません。デフォルトのオプションは **/var/www/html/** です。

例15.33 DocumentRoot 指示文の使用

```
DocumentRoot /var/www/html
```

ErrorDocument

ErrorDocument 指示文の使用により、特定のエラーへの反応として表示されるドキュメントまたはメッセージを指定できるようになります。以下の形式を取ります:

```
ErrorDocument エラーコード アクション
```

エラーコード は **403** (禁止)、**404** (発見不可)、**500** (内部サーバーエラー) などの有効なコードでなければなりません。*アクション* は URL (ローカルと外部の両方) か、あるいは二重引用符 (即ち、**"**) で囲まれたメッセージ文字列のどちらかです。

例15.34 ErrorDocument 指示文の使用

```
ErrorDocument 403 "Access Denied"
ErrorDocument 404 /404-not_found.html
```

ErrorLog

ErrorLog 指示文の使用により、サーバーエラーのログ先となるファイルを指定できるようになります。以下の形式を取ります:

```
ErrorLog パス
```

パス は、ログファイルを指しており、**ServerRoot** 指示文で指定されたディレクトリ (即ちデフォルトでは、**/etc/httpd/**) に対して絶対的、または相対的となるパスです。デフォルトのオプションは **logs/error_log** です。

例15.35 ErrorLog 指示文の使用

```
ErrorLog logs/error_log
```

ExtendedStatus

ExtendedStatus 指示文の使用で、サーバーステータスの詳細情報を有効にできるようになります。以下の形式を取ります:

```
ExtendedStatus オプション
```

オプション は 表15.7「利用可能な ExtendedStatus オプション」 に説明してあるように、有効なキーワードでなければなりません。デフォルトのオプションは **Off** です。

表15.7 利用可能な ExtendedStatus オプション

オプション	詳細
On	サーバーステータス詳細の生成を有効にします。
Off	サーバーステータス詳細の生成を無効にします。

例15.36 ExtendedStatus 指示文の使用

```
ExtendedStatus On
```

Group

Group 指示文の使用により、**httpd** サービスの実行元となるグループを指定できるようになります。以下の形式を取ります:

```
Group グループ
```

グループ は、既存の UNIX グループでなければなりません。デフォルトのオプションは **apache** です。

Group はもう **<VirtualHost>** 内ではサポートがないこと、及びそれが **SuexecUserGroup** 指示文に入れ替わっていることに注意して下さい。

例15.37 Group 指示文の使用

```
Group apache
```

HeaderName

HeaderName 指示文の使用により、サーバー生成のディレクトリー覧の先頭に前置きするファイルを指定できるようになります。以下の形式を取ります:

```
HeaderName ファイル名
```

ファイル名 は、要求されたディレクトリ内で探すファイルの名前です。デフォルトでは、サーバーは **HEADER.html** を探します。

例15.38 HeaderName 指示文の使用

```
HeaderName HEADER.html
```

HostnameLookups

HostnameLookups 指示文の使用により、IP アドレスの自動解決を有効にできるようになります。以下の形式を取ります:

```
HostnameLookups オプション
```

オプション は、表15.8「利用可能な HostnameLookups オプション」 で説明してあるように、有効なキーワードでなければなりません。サーバー上にリソースを保存するために、デフォルトのオプションは **Off** になっています。

表15.8 利用可能な HostnameLookups オプション

オプション	詳細
On	ホスト名がログできるように、各接続の IP アドレス解決を有効にします。しかし、これはプロセスの負担をかなり増加します。
Double	二重の逆引き DNS ルックアップの実施を可能にします。上記のオプションと比較すると、これはプロセスの負担を更に増大します。
Off	各接続の IP アドレス解決を無効にします。

サーバーのログファイル内にホスト名が必要になる際には、多くの場合、DNS ルックアップをより効率的に実行する様々なログアナライザーの 1 つを使用することができます。

例15.39 HostnameLookups 指示文の使用

```
HostnameLookups Off
```

Include

Include 指示文の使用により、他の設定ファイルを含むことができるようになります。以下の形式を取ります:

```
Include ファイル名
```

ファイル名 は、絶対パスでも、**ServerRoot** 指示文で指定されたディレクトリへの相対的パスでも、またはワイルドカード表現でも可能です。**/etc/httpd/conf.d/** ディレクトリの設定ファイルはすべて、デフォルトでロードされます。

例15.40 Include 指示文の使用

```
Include conf.d/*.conf
```

IndexIgnore

IndexIgnore 指示文の使用で、サーバー生成のディレクトリー覧から排除するファイル名の一覧を指定できるようになります。以下の形式を取ります:

-

```
IndexIgnore ファイル名…
```

*ファイル名*オプションは全長のファイル名か、またはワイルドカード表現になります。

例15.41 IndexIgnore 指示文の使用

```
IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t
```

IndexOptions

IndexOptions 指示文の使用により、サーバー生成のディレクトリー一覧の動作をカスタマイズできるようになります。以下の形式を取ります:

```
IndexOptions オプション…
```

オプション は、表15.9「利用可能なディレクトリー一覧のオプション」 で説明してあるような有効なキーワードでなければなりません。デフォルトのオプションには、**Charset=UTF-8**、**FancyIndexing**、**HTMLTable**、**NameWidth=***、及び **VersionSort** があります。

表15.9 利用可能なディレクトリー一覧のオプション

オプション	詳細
Charset=*エンコーディング*	生成されたウェブページの文字セットを指定します。*エンコーディング* は、**UTF-8** や **ISO-8859-2** などの有効な文字セットでなければなりません。
Type=*コンテンツタイプ*	生成されたウェブページのメディアタイプを指定します。*コンテンツタイプ* は **text/html** や **text/plain** などの有効な MIME タイプでなければなりません。
DescriptionWidth=*値*	説明カラムの幅を指定します。*値* は、文字数であるか、あるいは自動的に幅を調節するためのアスタリスク (即ち、*****) となります。
FancyIndexing	一定のファイル用の異なるアイコンや、あるいはカラムヘッダーのクリックでディレクトリー一覧の配列変更の可能性など高度な機能を有効にします。
FolderFirst	ディレクトリー一覧の先頭配置を有効にして、常にそれをファイルの上部に配置します。
HTMLTable	ディレクトリー一覧に HTML 表の使用を有効にします。
IconsAreLinks	リンクとしてのアイコン使用を有効にします。
IconHeight=*値*	アイコンの高さを指定します。*値* はピクセルの数になります。
IconWidth=*値*	アイコンの幅を指定します。*値* はピクセルの数です。

オプション	詳細
IgnoreCase	大文字/小文字の区別を持ったファイルとディレクトリの配列を有効にします。
IgnoreClient	クライアントからのクエリ変数の受理を無効にします。
NameWidth=*値*	ファイル名カラムの幅を指定します。*値* は文字数か、または幅を自動的に調節するためのアスタリスク (即ち、*****) となります。
ScanHTMLTitles	**AddDescription** 指示文で提供されていない場合に、説明 (即ち、**title** 要素) 用にファイルの構文解析を有効にします。
ShowForbidden	他の手段ではアクセス禁止のファイル一覧を有効にします。
SuppressColumnSorting	カラムヘッダーのクリックによるティレクトリー一覧の再配列を無効にします。
SuppressDescription	ファイル説明用のスペース予約を無効にします。
SuppressHTMLPreamble	**HeaderName** 指示文で指定されているファイルが存在する時には、標準の HTML 前置きの使用を無効にします。
SuppressIcon	ディレクトリー一覧内のアイコンの使用を無効にします。
SuppressLastModified	ディレクトリー一覧の最後の修正欄の日付表示を無効にします。
SuppressRules	ディレクトリー一覧の水平線の使用を無効にします。
SuppressSize	ディレクトリー一覧のファイルサイズフィールドの表示を無効にします。
TrackModified	HTTP ヘッダー内で **Last-Modified** と **ETag** の値の戻りを有効にします。
VersionSort	バージョン番号が期待された手法で含まれているファイルの配列を有効にします。
XHTML	デフォルトの HTML 3.2 ではなく、XHTML 1.0 の使用を有効にします。

例15.42 IndexOptions 指示文の使用

```
IndexOptions FancyIndexing VersionSort NameWidth=* HTMLTable
Charset=UTF-8
```

KeepAlive

KeepAlive 指示文の使用で、永続的な接続を有効にできるようになります。以下の形式を取ります:

```
KeepAlive オプション
```

オプション は 表15.10「利用可能な KeepAlive オプション」 で説明してあるような有効なキーワードでなければなりません。デフォルトのオプションは **Off** です。

表15.10 利用可能な KeepAlive オプション

オプション	詳細
On	永続的接続を有効にします。このケースでは、サーバーは接続毎に 1 つ以上の要求を受理します。
Off	keep-alive 接続を無効にします。

永続的接続が有効になった際に、多忙なサーバー上では子プロセスの数量が急激に増加して最終的には最大限度に到達し、サーバーを大幅に鈍化させます。このリスクを低減するために **KeepAliveTimeout** を少なめにセットして **/var/log/httpd/logs/error_log** のログファイルを注意して監視することをお勧めします。

例15.43 KeepAlive 指示文の使用

```
KeepAlive Off
```

KeepAliveTimeout

KeepAliveTimeout 指示文の使用により、接続を閉じるまで別の要求を待つ時間の長さを指定できるようになります。以下の形式を取ります:

```
KeepAliveTimeout 時間
```

時間 は秒で指定します。デフォルトのオプションは **15** です。

例15.44 KeepAliveTimeout 指示文の使用

```
KeepAliveTimeout 15
```

LanguagePriority

LanguagePriority 指示文の使用により、言語の優先度をカスタマイズできるようになります。以下の形式を取ります:

```
LanguagePriority 言語…
```

言語 は、**cs**、**en**、**fr** などの有効な MIME 言語でなければなりません。

この指示文は、クライアントの言語設定に基づいて複数の言語でコンテンツを提供するようなウェブサーバーに特に役に立ちます。

例15.45 LanguagePriority 指示文の使用

```
LanguagePriority sk cs en
```

`Listen`

Listen 指示文の使用で、リッスンする IP アドレス、またはポートを指定できるようになります。以下の形式を取ります:

```
Listen [ip-アドレス:]ポート [プロトコル]
```

ip-アドレス はオプションであり、供給されていないとサーバーはすべての IP アドレスからの任意の *ポート* 上で到来要求を受理します。*プロトコル* はポート番号から自動的に決定されるため、通常これは無視されます。デフォルトのオプションはポート **`80`** でリッスンすることになっています。

サーバーが 1024 以下のポートでリッスンするように設定されている場合は、スーパーユーザーのみが **`httpd`** サービスを開始できることに注意して下さい。

例15.46 Listen 指示文の使用

```
Listen 80
```

`LoadModule`

`LoadModule` 指示文の使用により、*Dynamic Shared Object* (DSO：動的共有オブジェクト) モジュールをロードできるようになります。以下の形式を取ります:

```
LoadModule 名前 パス
```

名前 は要求されたモジュールの有効な識別子でなければなりません。*パス* は、既存のモジュールファイルを指しており、ライブラリが配置してあるディレクトリへ相対的でなければなりません (即ち、デフォルトで 32-bit では、**`/usr/lib/httpd/`**、そして 64-bit では **`/usr/lib64/httpd/`**)。

Apache HTTP サーバーの DSO サポートの詳細については、「モジュールでの作業」 を参照して下さい。

例15.47 LoadModule 指示文の使用

```
LoadModule php5_module modules/libphp5.so
```

`LogFormat`

`LogFormat` 指示文の使用により、ログファイルの形式を指定できるようになります。以下の形式を取ります:

```
LogFormat 形式 名前
```

形式 は、表15.11「一般的な LogFormat オプション」 で説明してあるようにオプションで構成された文字列です。*名前*は **CustomLog** 指示文内の形式文字列の代わりに使用することができます。

表15.11 一般的な LogFormat オプション

オプション	詳細
%b	応答のサイズをバイトで示します。
%h	リモートクライアントの IP アドレス、またはホスト名を示します。
%l	供給されている場合は、リモートのログ名を示します。供給されていない場合は、代わりにハイフン (**-**) が使用れます。
%r	ブラウザー、またはクライアントから来た状態の要求文字列の最初の行を示します。
%s	ステータスコードを示します。
%t	要求の日付と時刻を示します。
%u	認証が必要な場合には、これはリモートユーザーを示します。認証無用なら、代わりにハイフン (**-**) が使用されます。
%{*フィールド*}	HTTP ヘッダー *フィールド* のコンテンツを示します。一般的なオプションには **%{Referer}** (クライアントをサーバーに照会したウェブページの URL) と **%{User-Agent}** (要求を出しているウェブブラウザーのタイプ) が含まれています。

例15.48 LogFormat 指示文の使用

```
LogFormat "%h %l %u %t \"%r\" %>s %b" common
```

LogLevel

LogLevel 指示文を使用すると、エラーログの詳細レベルをカスタマイズできるようになります。以下の形式を取ります:

```
LogLevel オプション
```

オプション は 表15.12「利用可能な LogLevel オプション」 で説明してあるような有効なキーワードでなければなりません。デフォルトのオプションは **warn** (警告) です。

表15.12 利用可能な LogLevel オプション

オプション	詳細
emerg	サーバーがその機能を実行できない緊急事態にのみログされます。

オプション	詳細
alert	すぐにアクションを要求される状況すべてがログされます。
crit	すべての重大な状況がログされます。
error	すべてのエラーメッセージがログされます。
warn	すべての警告メッセージがログされます。
notice	正常であっても、深刻な状態はログされます。
info	様々な情報的なメッセージがログされます。
debug	様々なデバッグメッセージがログされます。

例15.49 LogLevel 指示文の使用

```
LogLevel warn
```

MaxKeepAliveRequests

MaxKeepAliveRequests 指示文の使用により、永続的接続に対して最大要求数を指定できるようになります。以下の形式を取ります:

```
MaxKeepAliveRequests 数字
```

高い *数字* は、サーバーのパフォーマンスを向上できます。**0** を使用すると、無制限数の要求を認可することに注意して下さい。デフォルトのオプションは **100** です。

例15.50 MaxKeepAliveRequests オプションの使用

```
MaxKeepAliveRequests 100
```

NameVirtualHost

NameVirtualHost 指示文の使用により、名前ベースの仮想ホスト用に IP アドレスとポート番号を指定できるようになります。以下の形式を取ります:

```
NameVirtualHost ip-アドレス[:ポート]
```

ip-アドレス は全長の IP アドレスか、またはすべてのインターフェイスを表示するアスタリスク (*****) のいずれかとなります。IPv6 のアドレスは角カッコ (**[** と **]**) で囲む必要があります。*ポート* はオプションです。

名前ベースの仮想ホスト設定により、複数の IP アドレスを使用せずに単体の Apache HTTP サーバーが異なるドメイン群に対応できるようになります。

重要

名前ベースの仮想ホストは、セキュアな HTTP 接続以外で**のみ**機能します。セキュアなサーバーで仮想ホストを使用する場合は、その代わりとして IP アドレスベースの仮想ホストを使用して下さい。

例15.51 NameVirtualHost 指示文の使用

```
NameVirtualHost *:80
```

Options

Options 指示文の使用により、特定のディレクトリ内で利用可能になるサーバー機能を指定できるようになります。以下の形式を取ります:

```
Options オプション…
```

オプション は 表15.13「利用可能なサーバー機能」 に説明してあるような有効なキーワードでなければなりません。

表15.13 利用可能なサーバー機能

オプション	詳細
ExecCGI	CGI スクリプトの実行を有効にします。
FollowSymLinks	ディレクトリ内の次のシンボリックリンクを有効にします。
Includes	サーバー側の include を有効にします。
IncludesNOEXEC	サーバー側の include を有効にしますが、コマンドの実行を許可しません。
Indexes	サーバー生成のディレクトリー覧を有効にします。
MultiViews	コンテンツ交渉済みの 「MultiViews」 を有効にします。
SymLinksIfOwnerMatch	リンクと目的ファイルの両方が同一所有者を持つ場合にディレクトリ内の次のシンボリックリンクを有効にします。
All	**MultiViews** 以外の上記の機能すべてを有効にします。
None	上記の機能すべてを無効にします。

例15.52 Options 指示文の使用

```
Options Indexes FollowSymLinks
```

Order

Order 指示文の使用により、**Allow** と **Deny** の指示文が評価される順序を指定できるようになります。以下の形式を取ります:

```
Order オプション
```

オプション は、表15.14「利用可能な Order オプション」 で説明してあるように有効なキーワードでなければなりません。デフォルトのオプションは **allow,deny** です。

表15.14 利用可能な Order オプション

オプション	詳細
allow,deny	**Allow** 指示文が最初に評価されます。
deny,allow	**Deny** 指示文が最初に評価されます。

例15.53 Order 指示文の使用

```
Order allow,deny
```

PidFile

PidFile 指示文の使用により、サーバーの *プロセス ID* (PID) の保存先となるファイルを指定できるようになります。以下の形式を取ります:

```
PidFile パス
```

パス は、pid のファイルを指しており、絶対的パスか、または **ServerRoot** 指示文で指定されているディレクトリ (即ち、デフォルトで **/etc/httpd/**) へ相対的なパスのいずれかです。デフォルトのオプションは **run/httpd.pid** です。

例15.54 PidFile 指示文の使用

```
PidFile run/httpd.pid
```

ProxyRequests

ProxyRequests 指示文の使用により、フォワードプロキシの要求を有効にできるようになります。以下の形式を取ります:

```
ProxyRequests オプション
```

オプション は、表15.15「利用可能な ProxyRequests オプション」 で説明してあるような有効なキーワードでなければなりません。デフォルトのオプションは、**Off** です。

表15.15 利用可能な ProxyRequests オプション

オプション	詳細
On	フォワードプロキシの要求を有効にします。
Off	フォワードプロキシの要求を無効にします。

例15.55 ProxyRequests 指示文の使用

```
ProxyRequests On
```

ReadmeName

ReadmeName 指示文の使用により、サーバー生成のディレクトリー覧の末尾に追加するファイルを指定できるようになります。以下の形式を取ります:

```
ReadmeName ファイル名
```

ファイル名 は、要求されたディレクトリ内で探すファイルの名前です。デフォルトでは、サーバーは **README.html** を探します。

例15.56 ReadmeName 指示文の使用

```
ReadmeName README.html
```

Redirect

Redirect 指示文の使用により、別の URL へクライアントを転送できるようになります。以下の形式を取ります:

```
Redirect [ステータス] パス url
```

ステータス は、オプションですが提供される場合は、表15.16「利用可能な status オプション」 で説明してあるような有効なキーワードでなければなりません。*パス* は古い場所を指しており、**DocumentRoot** 指示文で指定してあるディレクトリ (f例えば、**/docs**) に相対的でなければなりません。*url* は、コンテンツの現在の場所を指します (例えば、**http://docs.example.com**)。

表15.16 利用可能な status オプション

Status	詳細
permanent	要求されたリソースが永久に移動されたことを示します。**301** (永久に移動済み) のステータスコードがクライアントに返されます。
temp	要求されたリソースが一時的にのみ移動されたことを示します。**302** (発見済み) ステータスコードがクライアントに返されます。

Status	詳細
seeother	要求されたリソースが入れ替えられていることを示します。**303** (他を確認) ステータスコードがクライアントに返されます。
gone	要求されたリソースが永久に排除されたことを示します。**410** (消失) ステータスコードがクライアントに返されます。

より高度な転送技術を確認するには、Apache HTTP サーバーインストールの一部である **mod_rewrite** モジュールを使用して下さい。

例15.57 Redirect 指示文の使用

```
Redirect permanent /docs http://docs.example.com
```

ScriptAlias

ScriptAlias 指示文を使用すると、CGI スクリプトの場所を指定できるようになります。以下の形式を取ります

```
ScriptAlias url-パス 実際のパス
```

url-パス は **DocumentRoot** 指示文で指定されたディレクトリ (例えば、**/cgi-bin/**) へ相対的でなければなりません。*実際のパス* はローカルファイルシステム内のファイル、またはディレクトリへの全長パスです。

この指示文には、標準的に目的ディレクトリへのアクセス権を持つ **Directory** タグが続きます。デフォルトでは、**/cgi-bin/** エイリアスが作成されるため、**/var/www/cgi-bin/** 内にあるスクリプトはアクセス可能になります。

ScriptAlias 指示文は、CGI スクリプトが平文のドキュメントとして閲覧されることを阻止するためにセキュリティ目的で使用されます。

例15.58 ScriptAlias 指示文の使用

```
ScriptAlias /cgi-bin/ /var/www/cgi-bin/

<Directory "/var/www/cgi-bin">
  AllowOverride None
  Options None
  Order allow,deny
  Allow from all
</Directory>
```

ServerAdmin

ServerAdmin 指示文を使用すると、サーバー管理者の電子メールアドレスがサーバー生成のウェブページ内に表示されるように指定できるようになります。以下の形式を取ります:

```
ServerAdmin 電子メールアドレス
```

デフォルトのオプションは **root@localhost** です。

この指示文は一般的に **webmaster@*hostname*** にセットされるものです。ここで *hostname* はサーバーのアドレスを示します。セットされると、**webmaster** を **/etc/aliases** 内のウェブサーバーの責任者にエイリアス化して、スーパーユーザーとなって **newaliases** コマンドを実行します。

例15.59 ServerAdmin 指示文の使用

```
ServerAdmin webmaster@penguin.example.com
```

ServerName

ServerName 指示文の使用により、ウェブサーバーのホスト名とポート番号を指定できるようになります。以下の形式を取ります:

```
ServerName ホスト名[:ポート]
```

ホスト名 は、サーバーの完全修飾型ドメイン名 (FQDN) でなければなりません。*ポート* はオプションですが、供給されている場合は、**Listen** 指示文に指定された番号と一致する必要があります。

この指示文を使用する際には、IP アドレスとサーバー名のペアが **/etc/hosts** ファイル内に含まれていることを確認して下さい。

例15.60 ServerName 指示文の使用

```
ServerName penguin.example.com:80
```

ServerRoot

ServerRoot 指示文の使用により、サーバーが運用できるディレクトリを指定できるようになります。以下の形式を取ります:

```
ServerRoot ディレクトリ
```

ディレクトリ は、ローカルファイルシステム内の既存のディレクトリへの全長パスでなければなりません。デフォルトのオプションは、**/etc/httpd/** です。

例15.61 ServerRoot 指示文の使用

```
ServerRoot /etc/httpd
```

ServerSignature

ServerSignature 指示文の使用により、サーバー生成ドキュメント上でサーバーに関する情報の表示を有効にできるようになります。以下の形式を取ります:

```
ServerSignature オプション
```

- *オプション* は、表15.17「利用可能な ServerSignature オプション」 で説明してあるような有効なキーワードでなければなりません。デフォルトのオプションは、**On** です。

表15.17 利用可能な ServerSignature オプション

オプション	詳細
On	サーバー生成ページへのサーバー名とバージョンの追記を有効にします。
Off	サーバー生成ページへのサーバー名とバージョンの追記を無効にします。
EMail	サーバー生成ページにへのサーバー名、バージョン、及び **ServerAdmin** 指示文で指定されているシステム管理者の電子メールアドレスの追記を有効にします。

例15.62 ServerSignature 指示文の使用

```
ServerSignature On
```

ServerTokens

ServerTokens 指示文の使用により、サーバーの応答ヘッダー内に含める情報をカスタマイズできるようになります。以下の形式を取ります:

```
ServerTokens オプション
```

オプション は、表15.18「利用可能な ServerTokens オプション」 で説明してあるような有効なキーワードでなければなりません。デフォルトのオプションは **OS** です。

表15.18 利用可能な ServerTokens オプション

オプション	詳細
Prod	製品名 (即ち、**Apache**) のみを含みます。
Major	製品名とサーバーのメジャーバージョン (例えば、**2**) を含みます。
Minor	製品名とサーバーのマイナーバージョン (例えば、**2.2**) を含みます。
Min	製品名とサーバーの最小バージョン (例えば、**2.2.15**) を含みます。
OS	製品名、サーバーの最小バージョン、及びシステムが実行しているオペレーティングシステムのタイプ (例えば、**Red Hat**) を含みます。
Full	上述の情報すべてと更にロード済みモジュールの一覧を含みます。

セキュリティの目的で、サーバーについては極力最低限の情報を表明することが推奨されます。

例15.63 ServerTokens 指示文の使用

```
ServerTokens Prod
```

SuexecUserGroup

SuexecUserGroup 指示文の使用により、CGI スクリプトの実行元となるユーザーとグループを指定できるようになります。以下の形式を取ります:

```
SuexecUserGroup ユーザー グループ
```

ユーザー は既存のユーザーであり、*グループ* は有効な UNIX グループでなければなりません。

セキュリティの目的で、CGI スクリプトは root 権限で実行すべきではありません。**<VirtualHost>** 内では、**SuexecUserGroup** が **User** と **Group** の指示文の入れ替わりになることに注意して下さい。

例15.64 SuexecUserGroup 指示文の使用

```
SuexecUserGroup apache apache
```

Timeout

Timeout 指示文を使用すると、接続を閉じるまでにイベントを待つ時間を指定できるようになります。以下の形式を取ります:

```
Timeout 時間
```

時間 は秒で指定します。デフォルトのオプションは **60** です。

例15.65 Timeout 指示文の使用

```
Timeout 60
```

TypesConfig

TypesConfig の使用で、MIME タイプ設定の場所を指定できるようになります。以下の形式を取ります:

```
TypesConfig パス
```

パス は、既存の MIME タイプの設定ファイルを指しており、絶対的パスであるか、または **ServerRoot** 指示文で指定されているディレクトリ (即ち、デフォルトでは、**/etc/httpd/**) に対して相対的パスのいずれかです。デフォルトのオプションは **/etc/mime.types** です。

/etc/mime.types の編集の代わりに、MIME タイプマッピングを Apache HTTP サーバーに追加する最善の推奨方法は、**AddType** 指示文の使用であることに注意して下さい。

例15.66 TypesConfig 指示文の使用

```
TypesConfig /etc/mime.types
```

UseCanonicalName

UseCanonicalName の使用により、サーバーがそれ自身を呼称する方法を指定できるようになります。以下の形式を取ります:

```
UseCanonicalName オプション
```

オプション は、表15.19「利用可能な UseCanonicalName オプション」 で説明してあるような有効なキーワードでなければなりません。デフォルトのオプションは **Off** です。

表15.19 利用可能な UseCanonicalName オプション

オプション	詳細
On	**ServerName** 指示文で指定された名前の使用を有効にします。
Off	**ServerName** 指示文で指定された名前の使用を無効にします。代わりに、要求元のクライアントで提供されたホスト名とポート番号が使用されます。
DNS	**ServerName** 指示文で指定された名前の使用を無効にします。代わりに、逆引き DNS ルックアップで決定したホスト名が使用されます。

例15.67 UseCanonicalName 指示文の使用

```
UseCanonicalName Off
```

User

User 指示文の使用により、**httpd** サービスの実行元となるユーザーを指定できるようになります。以下の形式を取ります:

```
User ユーザー
```

ユーザー は、既存の UNIX ユーザーでなければなりません。デフォルトのオプションは **apache** です。

セキュリティ目的で、**httpd** サービスは root 権限で実行すべきではありません。**User** はもう **<VirtualHost>** 内ではサポートされておらず、**SuexecUserGroup** 指示文に入れ替わっていることに注意して下さい。

例15.68 User 指示文の使用

```
User apache
```

UserDir

UserDir 指示文の使用により、ユーザーのホームディレクトリからのコンテンツ作業を有効にできるようになります。以下の形式を取ります:

```
UserDir オプション
```

オプション は、ユーザーのホームディレクトリ内で見つけるディレクトリの名前 (標準的に **public_html**) であるか、または 表15.20「利用可能な UserDir オプション」 で説明してあるような有効なキーワードのいずれかです。デフォルトのオプションは **disabled** です。

表15.20 利用可能な UserDir オプション

オプション	詳細
enabled *ユーザー...*	該当する *ユーザー* のホームディレクトリからのコンテンツ作業を有効にします。
disabled [*ユーザー...*]	すべてのユーザーに対して、または空白で隔離した *ユーザー* の一覧が供給されている場合は該当するユーザーのみに対して、ホームディレクトリからのコンテンツ作業を無効にします。

注記

ウェブサーバーがコンテンツにアクセスするためには、関連したディレクトリとファイルでの権限を正しく設定する必要があります。すべてのユーザーがホームディレクトリにアクセスできること、そして彼らが**UserDir** 指示文で指定されたディレクトリのコンテンツにアクセスして読み込みができることを確実にします。例えば:

```
~]# chmod a+x /home/username/
~]# chmod a+rx /home/username/public_html/
```

このディレクトリ内のファイルはすべてそのようにセットする必要があります。

例15.69 UserDir 指示文の使用

```
UserDir public_html
```

15.1.5.2. 一般的な ssl.conf 指示文

Secure Sockets Layer (セキュアソケットレイヤー) (SSL) 指示文の使用により、Apache HTTP セキュアサーバーの動作をカスタマイズできるようになります。多くの場合、サーバーはインストール中に適切に設定されます。しかし、間違えた設定はセキュリティ脆弱性の原因となりますので、設定の変更時には注意が必要です。

以下の指示文は通常、**/etc/httpd/conf.d/ssl.conf** 内で使用されます:

SetEnvIf

SetEnvIf 指示文を使用すると、受信接続のヘッダーに基づいて環境変数の設定ができるようになります。以下の形式を取ります:

> SetEnvIf *オプション パターン* [!]*変数*[=*値*]…

オプション は、HTTP ヘッダーフィールドか、以前に定義してある環境変数の名前か、あるいは 表15.21「利用可能な SetEnvIf オプション」 内に説明してあるような有効なキーワードのいずれかになります。*パターン* は、正式表現です。*変数* は、オプションがパターンにマッチした時にセットされる環境変数です。オプションの感嘆符 (**!**) が存在していると、変数は設定ではなく、削除されます。

表15.21 利用可能な SetEnvIf オプション

オプション	詳細
Remote_Host	クライアントのホスト名を指しています。
Remote_Addr	クライアントの IP アドレスを指しています。
Server_Addr	サーバーの IP アドレスを指しています。
Request_Method	要求の方法を指しています (例えば、**GET**)。
Request_Protocol	プロトコル名とバージョンを指しています (例えば、**HTTP/1.1**)。
Request_URI	要求されたリソースを指しています。

SetEnvIf 指示文は、HTTP keepalives を無効にするためと、クライアントブラウザーからの通知を封じることなく、SSL が接続を切断できるようするために使用します。SSL 接続の切断に信頼できない一部のウェブブラウザーには、これが必要です。

例15.70 SetEnvIf 指示文の使用

```
SetEnvIf User-Agent ".*MSIE.*" \
         nokeepalive ssl-unclean-shutdown \
         downgrade-1.0 force-response-1.0
```

/etc/httpd/conf.d/ssl.conf ファイルが存在するためには、mod_ssl をインストールしておく必要があることに注意して下さい。SSL サーバーのインストールと設定の方法に関する詳細については、「SSL サーバーのセットアップ」 を参照して下さい。

15.1.5.3. 一般的な Multi-Processing Module 指示文

Multi-Processing Module (マルチプロセッシングモジュール) (MPM) 指示文の使用により、特定の MPM の特定のサーバープールの動作をカスタマイズできるようになります。その性質は使用する MPM によって変化するため、指示文は **IfModule** に組み込まれています。デフォルトでは、サーバープールは **prefork** と **worker** の両方の MPM 用に定義してあります。

以下の MPM 指示文は一般的に **/etc/httpd/conf/httpd.conf** 内で使用されます:

MaxClients

MaxClients 指示文の使用により、一度にプロセスできる同時接続のクライアントの最大数を指定できるようになります。以下の形式を取ります:

```
MaxClients 数字
```

高い *数字* はサーバーのパフォーマンスを向上しますが、**prefork** MPM を使用している場合には、**256** を超過することは推奨できません。

例15.71 MaxClients 指示文の使用

```
MaxClients 256
```

MaxRequestsPerChild

MaxRequestsPerChild 指示文の使用により、子プロセスが終了するまでに対応できる最大要求数を指定できるようになります。以下の形式を取ります:

```
MaxRequestsPerChild 数字
```

数字 を **0** にセットすると、無制限の要求を受け付けることになります。

MaxRequestsPerChild 指示文は長期生存しているプロセスのメモリーリークを防止するために使用されます。

例15.72 MaxRequestsPerChild 指示文の使用

```
MaxRequestsPerChild 4000
```

MaxSpareServers

MaxSpareServers 指示文の使用で、予備子プロセスの最大数を指定できるようになります。以下の形式を取ります:

```
MaxSpareServers 数字
```

この指示文は **prefork** MPM によってのみ使用されます。

例15.73 MaxSpareServers 指示文の使用

```
MaxSpareServers 20
```

MaxSpareThreads

MaxSpareThreads 指示文の使用で、予備のサーバースレッドの最大数を指定できるようになります。以下の形式を取ります:

```
MaxSpareThreads 数字
```

数字 は、**MinSpareThreads** と **ThreadsPerChild** の合計に等しいか、それ以上でなければなり

ません。この指示文は、**worker** MPM によってのみ使用されます。

例15.74 MaxSpareThreads 指示文の使用

```
MaxSpareThreads 75
```

MinSpareServers

MinSpareServers 指示文の使用で、予備子プロセスの最小数を指定できるようになります。以下の形式を取ります:

```
MinSpareServers 数字
```

高い *数字* は、サーバーに対して高負荷のプロセス処理を与えることに注意して下さい。この指示文は **prefork** MPM によってのみ使用されます。

例15.75 MinSpareServers 指示文の使用

```
MinSpareServers 5
```

MinSpareThreads

MinSpareThreads 指示文の使用により、予備のサーバースレッドの最低数を指定できるようになります。以下の形式を取ります:

```
MinSpareThreads 数字
```

この指示文は、**worker** MPM によってのみ使用されます。

例15.76 MinSpareThreads 指示文の使用

```
MinSpareThreads 75
```

StartServers

StartServers 指示文の使用により、サービスが開始した時点で作成する子プロセスの数を指定できるようになります。以下の形式を取ります:

```
StartServers 数字
```

子プロセスは、現在のトラフィック状況に応じて動的に作成及び終結されるため、通常はこの値を変更する必要はありません。

例15.77 StartServers 指示文の使用

```
StartServers 8
```

ThreadsPerChild

ThreadsPerChild 指示文の使用で、子プロセスが作成できるスレッドの数を指定できるようになります。以下の形式を取ります:

```
ThreadsPerChild 数字
```

この指示文は、**worker** MPM によってのみ使用されます。

例15.78 ThreadsPerChild 指示文の使用

```
ThreadsPerChild 25
```

15.1.6. モジュールでの作業

モジュラーアプリケーションである **httpd** サービスは、必要に応じてランタイムに動的にロード、またはアンロードできる多数の *Dynamic Shared Objects (動的共有オブジェクト)* (DSOs) と一緒に配布されています。デフォルトではこれらのモジュールは 32-bit システムでは、**/usr/lib/httpd/modules/** 内に、64-bit システムでは、**/usr/lib64/httpd/modules/** 内に配置されています。

15.1.6.1. モジュールのロード

特定の DSO モジュールをロードするには、「一般的な httpd.conf の指示文」 で説明してあるように、**LoadModule** 指示文を使用します。個別のパッケージで提供されるモジュールでは多くの場合、それ自身の設定ファイルが **/etc/httpd/conf.d/** ディレクトリに配置してあることに注意して下さい。

例15.79 mod_ssl DSO のロード

```
LoadModule ssl_module modules/mod_ssl.so
```

操作が終了したら、ウェブサーバーを再開始して設定を再ロードします。**httpd** サービスの再開始の方法については 「サービスの再開始」 を参照して下さい。

15.1.6.2. モジュールの書き込み

新規の DSO モジュールを作成する予定がある場合は、httpd-devel パッケージを最初にインストールしていることを確認して下さい。インストールするには、シェルプロンプトで以下を入力します:

```
~]# yum install httpd-devel
```

このパッケージには、モジュールをコンパイルするために必要なファイル、ヘッダーファイル、及び **APache eXtenSion** (**apxs**) ユーティリティが含まれています。

書き込みが終了したら、以下のコマンドでモジュールをビルドできます:

```
~]# apxs -i -a -c module_name.c
```

ビルドが成功すると、Apache HTTP サーバーで配布されている他のモジュールと同様にこのモジュールをロードできるはずです。

15.1.7. 仮想ホストのセットアップ

Apache HTTP サーバーに組み込まれている仮想ホスト設定は、要求されている IP アドレス、ホスト名、あるいはポートに基づいてサーバーが異なる情報を提供できるようにします。

名前ベースの仮想ホストを作成するには、まず **/etc/httpd/conf/httpd.conf** 内に提供してある仮想ホストコンテナを例として見つけます。そしてハッシュマーク (#) を各行の先頭から削除して、例 15.80「仮想ホスト設定のサンプル」 に示してあるような方法で必要に応じてオプションをカスタマイズします。

例15.80 仮想ホスト設定のサンプル

```
NameVirtualHost penguin.example.com:80

<VirtualHost penguin.example.com:80>
    ServerAdmin webmaster@penguin.example.com
    DocumentRoot /www/docs/penguin.example.com
    ServerName penguin.example.com:80
    ErrorLog logs/penguin.example.com-error_log
    CustomLog logs/penguin.example.com-access_log common
</VirtualHost>
```

ServerName はマシンに割り当てられている有効な DNS 名でなければならないことに注意して下さい。**<VirtualHost>** コンテナは高度にカスタマイズ可能で、主なサーバー設定内で利用できるほとんどの指示文を受け付けます。このコンテナ内でサポート**されていない**指示文の中に、**User** と **Group** があり、これらは **SuexecUserGroup** に入れ替わっています。

注記

仮想ホストがデフォルト以外のポートでリッスンするように設定する場合は、**/etc/httpd/conf/httpd.conf** ファイルのグローバル設定で **Listen** 指示文を適切に更新することを忘れないで下さい。

新規に作成された仮想ホストをアクティベートするには、ウェブサーバーを再開始する必要があります。**httpd** サービスを再開始する方法の説明については「サービスの再開始」 を参照して下さい。

15.1.8. SSL サーバーのセットアップ

Secure Sockets Layer (SSL) (セキュアソケットレイヤー) は暗号化のプロトコルであり、サーバーとクライアントが安全に通信できるようにします。その拡張及び改善バージョンである *Transport Layer Security* (TLS) (トランスポートレイヤーセキュリティ) と共に、SSL はプライバシーとデータの整合性を保証します。Apache HTTP サーバーは、SSL/TLS サポートを提供するために OpenSSL ツールキットを使用するモジュールである **mod_ssl** と一緒に組み合わせて、一般的に *SSL サーバー*と呼ばれます。

傍受できる人は誰でも読み取りと修正が可能な普通の HTTP 接続とは異なり、**mod_ssl** を使用すると、発信されたコンテンツの検査や修正を阻止できます。このセクションでは、Apache HTTP サーバー設定内でこのモジュールを有効にする方法について基本情報を提供し、プライベートキーと自己署名の証明書の生成プロセスをガイドしていきます。

15.1.8.1. 証明書とセキュリティの概要

安全な通信はキーの使用に基づいています。常習的な、即ち *symmetric cryptography (対称型暗号化)* では、トランザクションの両端でお互いの通信を解読するのに同じキーを使用します。それに対し、パブリック、即ち *asymmetric cryptography (非対称型暗号化)* では、２つのキーが共存します。*プライベートキー* は秘密として守られて、*パブリックキー* は通常、公共と共有されます。パブリックキーでエンコードされたデータはプライベートキーでのみデコードされるのに対し、プライベートキーでエンコードされたデータは逆にパブリックキーでのみデコードできます。

SSL を使用して安全な通信を提供するためには、SSL サーバーは *Certificate Authority* (CA) (認証局) で署名された電子証明書を使用しなければなりません。証明書はサーバーの様々な属性 (サーバーのホスト名、企業の名前、その住所など) と CA のプライベートキーで生成した署名を一覧表示します。この証明書は特定の認証局が証明書を発行したこと、及び証明書がいかなる方法でも修正されていないことを保証するものです。

ウェブブラウザーは新規の SSL 接続を確立する際に、ウェブサーバーから提供される証明書をチェックします。証明書が信頼できる CA からの署名を持たない場合や、または証明書に表記してあるホスト名が接続の確立で使用されたホスト名と一致しない場合は、ウェブブラウザーはそのサーバーとの交信を拒否して、通常はユーザーに適切なエラーメッセージを表示します。

デフォルトでは、ほとんどのウェブブラウザーが広域で使用されている認証局のグループを信用するように設定されています。そのため、セキュアサーバーをセットアップする時には適切な CA (認証局) を選択すべきです。そうすることで、対象ユーザーは接続を信頼できますが、そうでない場合はエラーメッセージが表示されて、手動で証明書を受理する必要があります。証明書のエラーメッセージを無視するようにユーザーに推奨することは攻撃者の侵略を許すことになりますので、できる限り信頼できる CA を使用しなければなりません。この件の詳細については、表15.22「最も一般的なウェブブラウザー用の CA 一覧」 をご覧下さい。

表15.22 最も一般的なウェブブラウザー用の CA 一覧

ウェブブラウザー	リンク
Mozilla Firefox	Mozilla root CA 一覧.
Opera	The Opera Rootstore.
Internet Explorer	Windows ルート証明書プログラムのメンバー.

SSL サーバーをセットアップする際には、証明書要求とプライベートキーを生成してから、証明書要求と企業の識別証明と支払いを認証局に送ります。認証局は証明書要求と識別を確証すると、サーバーで使用できる署名付きの証明書を送ってきます。その他の方法として、CA 署名のない自己署名の証明書を作成することができますが、これはテスト目的のみに使用すべきものです。

15.1.8.2. mod_ssl モジュールを有効にする

SSL サーバーをセットアップする予定の場合は、mod_ssl (**mod_ssl** モジュール) とopenssl (OpenSSL ツールキット) のパッケージがインストールされていることを確認して下さい。インストールするには、シェルプロンプトで以下を入力します:

```
~]# yum install mod_ssl openssl
```

これにより、中心となる Apache HTTP サーバー設定ファイルにデフォルトで含まれている **/etc/httpd/conf.d/ssl.conf** に **mod_ssl** 設定ファイルが作成されます。このモジュールをロードするには、「サービスの再開始」 で説明してあるように、**httpd** サービスを再開始します。

15.1.8.3. 既存のキーと証明書の使用

以前にキーと証明書を作成している場合は、新規で作成するのではなく、既存のファイルを使用するように SSL サーバーを設定することができます。但しこれが実行できないケースが 2 つだけ存在します。

1. **IP アドレス、またはドメイン名を変更しようとしているケース。**

   証明書は特定の IP アドレスとドメイン名のペアに発行されるものです。これらの値の 1 つが変更になると証明書は無効になります。

2. **VeriSign からの証明書があって、サーバーソフトウェアを変更しようとしているケース。**

   広範囲に使用されている認証局である VeriSign は、特定のソフトウェア製品、IP アドレス、及びドメイン名に証明書を発行します。ソフトウェア製品を変更すると証明書を無効にします。

上記ケースのいずれの場合も、新しい証明書を取得する必要があります。この件に関する情報については、「新しいキーと証明書の生成」 を参照して下さい。

既存のキーと証明書を使用する場合は、その関連ファイルを **/etc/pki/tls/private/** ディレクトリと **/etc/pki/tls/certs/** ディレクトリにそれぞれ移動します。それを達成するには以下のコマンドを入力します:

```
~]# mv key_file.key /etc/pki/tls/private/hostname.key
~]# mv certificate.crt /etc/pki/tls/certs/hostname.crt
```

それから、以下の行を **/etc/httpd/conf.d/ssl.conf** 設定ファイルに追加します:

```
SSLCertificateFile /etc/pki/tls/certs/hostname.crt
SSLCertificateKeyFile /etc/pki/tls/private/hostname.key
```

更新済みの設定をロードするには、「サービスの再開始」 で説明してあるように、**httpd** サービスを再開始します。

例15.81 Red Hat セキュアウェブサーバーからのキーと証明書の使用

```
~]# mv /etc/httpd/conf/httpsd.key
/etc/pki/tls/private/penguin.example.com.key
~]# mv /etc/httpd/conf/httpsd.crt
/etc/pki/tls/certs/penguin.example.com.crt
```

15.1.8.4. 新しいキーと証明書の生成

新しいキーと証明書のペアを生成するには、システムに crypto-utils パッケージをインストールしておく必要があります。これは、シェルプロンプトで以下を入力するとインストールできます:

```
~]# yum install crypto-utils
```

このパッケージは SSL 証明書とプライベートキーを生成して管理するツールセットを提供すると共に、キー生成のプロセスをガイドしてくれる Red Hat Keypair Generation ユーティリティである **genkey** を収納しています。

重要

有効な証明書を既に所有していて、それを新規のものと入れ替える予定の場合は、異なるシリアル番号を指定します。これにより、クライアントのブラウザーがこの変更の通知を受けて予定どおりに新規の証明書に更新して、このページへのアクセスに失敗しないようにします。カスタムのシリアル番号で新規の証明書を作成するには、**genkey** の代わりに、以下のコマンドを使用します:

```
~]# openssl req -x509 -new -set_serial number -key hostname.key
-out hostname.crt
```

注記

使用システム内に特定のホスト名用のキーファイルが既に存在する場合は、**genkey** は開始を拒否します。その場合は、以下のコマンドを使用して既存のキーを削除します:

```
~]# rm /etc/pki/tls/private/hostname.key
```

ユーティリティを実行するには、**genkey** コマンドの後に該当するホスト名 (例えば、**penguin.example.com**) を付けて使用します:

```
~]# genkey hostname
```

キーと証明書の生成を完了するには、以下の手順に従います:

1. キーと証明書の保存先となる目的地を確認する。

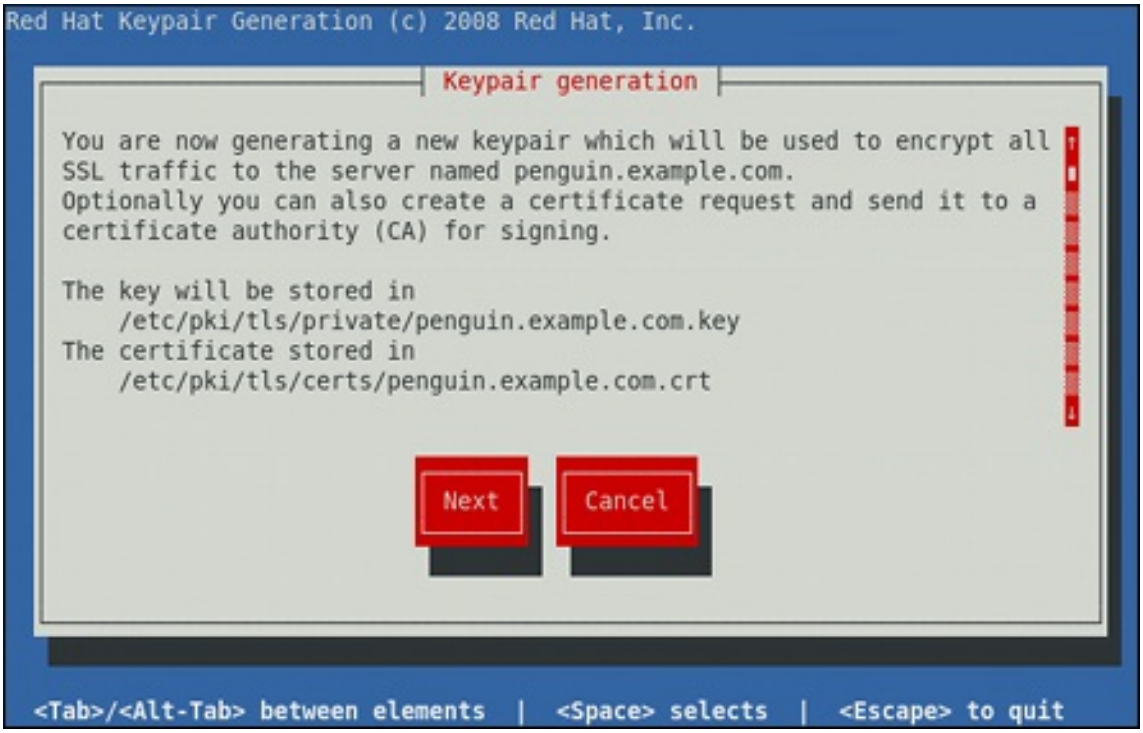


図15.1 genkey ユーティリティの実行

Tab キーを使用して **Next (次へ)** ボタンを選択してから、**Enter** キーを押すと次の画面を進みます。

2. **Up** と **down** の矢印キーを使用して、適切なキーサイズを選択します。大きなキーはセキュリティを増強しますが、サーバーの反応時間も増加することに注意して下さい。このため、推奨できるオプションは、**1024 bits** です。

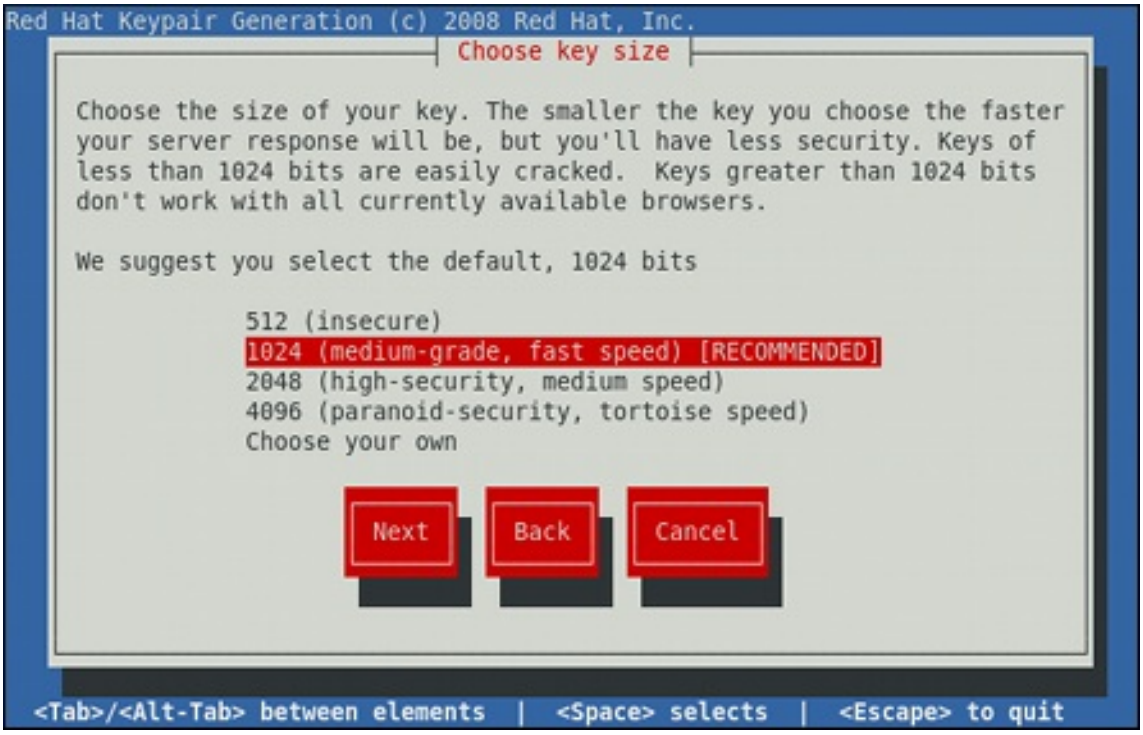


図15.2 キーサイズの選択

終了したら、**Tab** キーを使用して **Next (次へ)** ボタンを選択して **Enter** キーを押すと、ランダムなビットの生成プロセスが開始します。選択したキーサイズに応じてこれには少々時間がかかります。

3. 証明書要求を認証局に送信するかどうかを決定します。

図15.3 証明書要求の生成

Tab キーを使用して **Yes (はい)** を選択し、証明書要求を組み立てるか、または **No (いいえ)** を選択して、自己署名の証明書を生成します。その後に、**Enter** を押して選択を確定します。

4. **Spacebar (スペースバー)** キーを使用すると、プライベートキーの暗号化を有効にする (**[*]**) か、無効にする (**[ ]**) 選択ができます。

図15.4 プライベートキーの暗号化

Tab キーを使用して **Next (次へ)** ボタンを選択してから、**Enter** キーを押すと次の画面を進みます。

5. プライベートキーの暗号化を有効にしている場合は、適切なパスフレーズを入力します。セキュリティの理由で、入力時には文字が表示されませんが、最低でも５文字の長さが必要です。

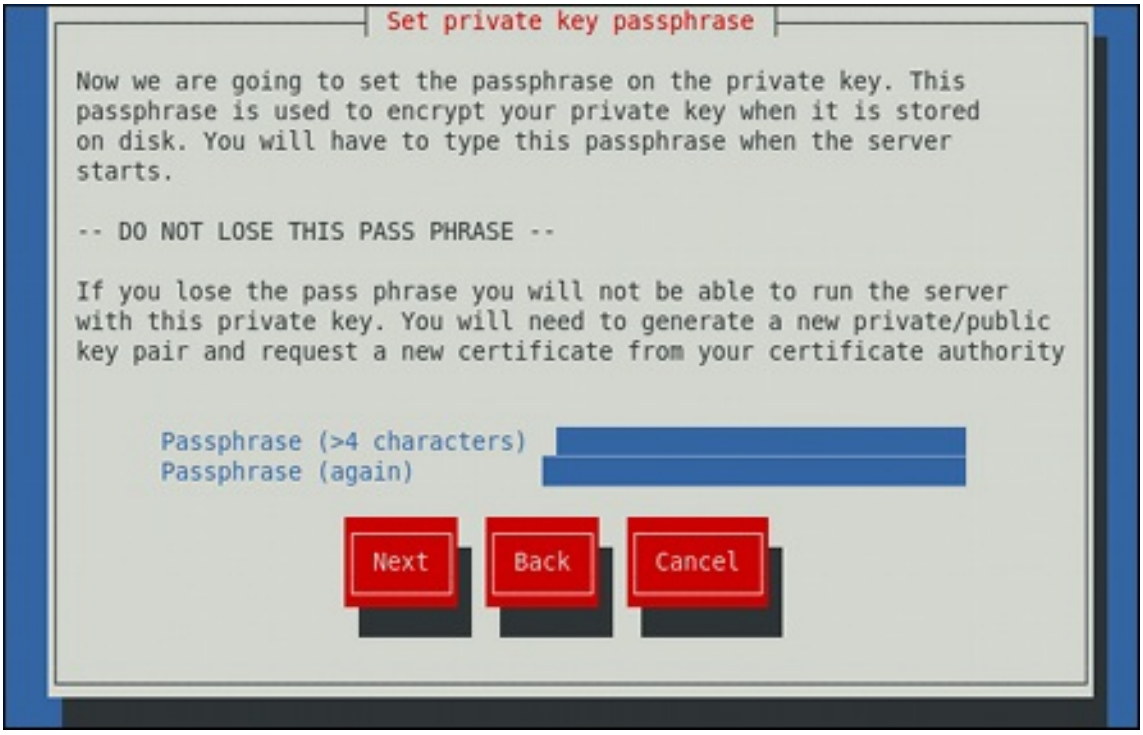


図15.5 パスフレーズの入力

Tab キーを使用して **Next (次へ)** ボタンを選択してから、**Enter** キーを押すと次の画面を進みます。

重要

サーバーを開始するには正しいパスフレーズの入力が必要です。それを紛失したり忘れたりした場合は、新しいキーと証明書を生成しなければなりません。

6. 証明書詳細のカスタマイズ

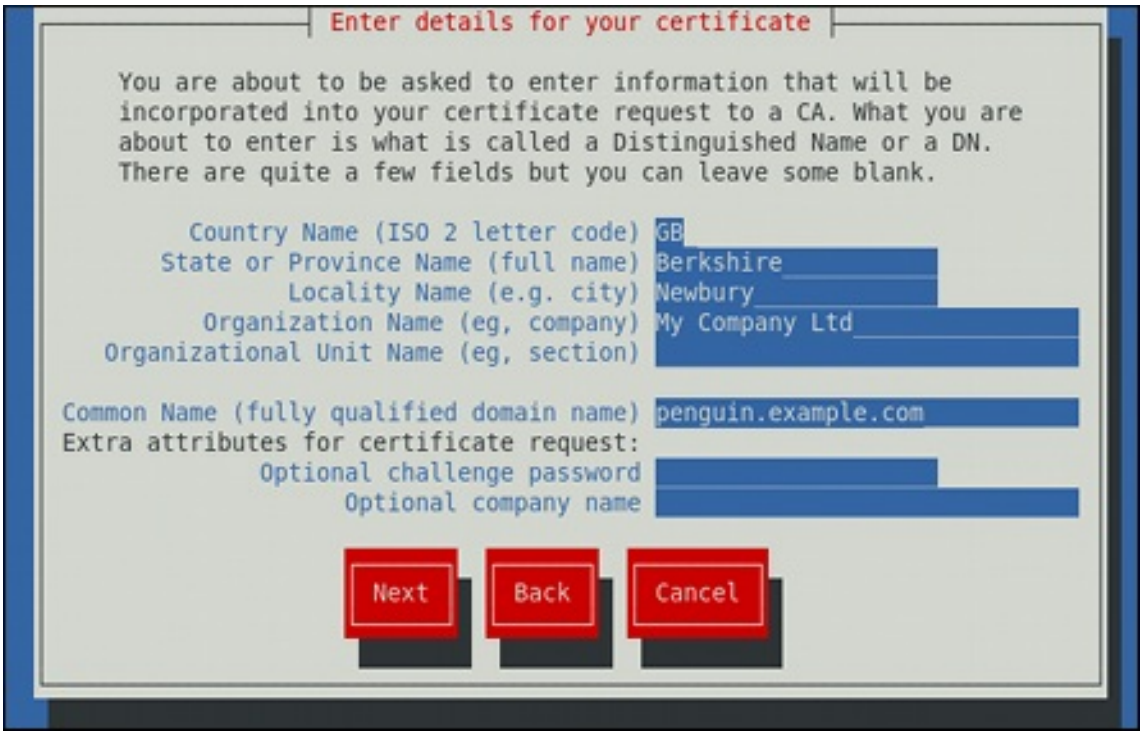


図15.6 証明書情報の指定

Tab キーを使用して **Next** ボタンを選択します。それから **Enter** を押すとキー生成ができます。

7. 以前に証明書要求の生成を有効にしていた場合は、それを認証局に送信するように催促されます。

```
You now need to submit your CSR and documentation to your certificate
authority. Submitting your CSR may involve pasting it into an online
web form, or mailing it to a specific address. In either case, you
should include the BEGIN and END lines.

-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBqjCCARMCAQAwajELMAkGA1UEBhMCR0IxEjAQBgNVBAgTCUJlcmtzaGlyZTEQ
MA4GA1UEBxMHTmV3YnVyeTEXMBUGA1UEChMOTXkgQ29tcGFueSBMdGQxHDAaBgNV
BAMTE3Blbmd1aW4uZXhhbXBsZS5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJ
AoGBAJjw8bXq7WKGGXNZsNZltEe9849wUMc4uAh+X8251b8x+ptJQCanGeNhLlXU
xiL5srY2TjoTSQ5DvyFgPQmFFe3cn7v//bKNgNqd4h0EbRFGaj/hDUG3fXnjujkX
hP+9iY/eIAQZlHQSkABh/2egtIllpfDeRvsTUX376TnkIWLhAgMBAAGgADANBgkq
hkiG9w0BAQQFAAOBgQBUTjgjcnts1hZKO70c5j+b4IfsBCwm4lnvGx3jOwpLdRq/
rHpx5cbHV99vcKnF3CwDrze9DgpTdjdbAccSCVgSG5GE8JZXWYD8EK8p2naJNQl1
YVX1KPi5MPLZuZ9cTb+k4K0cbug0IQiYaKNLNI/0zLE1VEWZXYFXOUBFM2gXYw==
-----END NEW CERTIFICATE REQUEST-----


A copy of this CSR has been saved in the file
/etc/pki/tls/certs/penguin.example.com.1.csr

Press return when ready to continue
```

図15.7 証明書要求を送信する方法の案内

Enter を押してシェルプロンプトに戻ります。

生成が終了したら、キーと証明書の場所を **/etc/httpd/conf.d/ssl.conf** 設定ファイルに追加します:

```
SSLCertificateFile /etc/pki/tls/certs/hostname.crt
SSLCertificateKeyFile /etc/pki/tls/private/hostname.key
```

最後に、「サービスの再開始」 で説明してあるとおりに **httpd** サービスを再開始します。そうすると更新した設定がロードされます。

15.1.9. その他のリソース

Apache HTTP サーバーに関する詳細については、以下のリソースを参照してください。

15.1.9.1. インストールされているドキュメント

http://localhost/manual/

指示文と利用可能なモジュールの全説明を含んでいる、Apache HTTP サーバーの公式ドキュメントです。このドキュメントにアクセスするには、httpd-manual パッケージがインストールしてあり、ウェブサーバーが稼働している必要があります。

man httpd

コマンドラインオプションの全一覧が収納されている **httpd** サービス用の man ページです。

man genkey

genkey の man ページにはその使用法についての全ドキュメントが含まれています。

15.1.9.2. 役立つ Web サイト

http://httpd.apache.org/

すべての指示文とデフォルトのモジュールに関するドキュメントが収納された、Apache HTTP サーバーの公式ウェブサイトです。

http://www.modssl.org/

mod_ssl モジュールに関する公式の Web サイトです。

http://www.openssl.org/

さらに詳しいドキュメントや FAQ、メーリングリストへのリンク、その他役立つリソースを掲載した OpenSSL のホームページです。

第16章 メールサーバー

電子メール は 1960 年代に誕生しました。当時のメールボックスは、ユーザーのホームディレクトリにあるファイルで、そのユーザーのみが読み取り可能でした。初期のメールアプリケーションは、そのファイルの最下部に新規テキストメッセージを追加していたため、ユーザーは、絶えず拡大するファイルの情報をかき分けて特定のメッセージを見つける必要がありました。この方式では、同じシステム上のユーザーに対してしかメッセージを送信できませんでした。

電子メールのメッセージファイルのネットワーク転送は、1971 年に Ray Tomlinson というコンピュータエンジニアが、インターネットの先駆けである ARPANET を介して 2 台のマシン間でテストメッセージを送信したのが初めてです。電子メールによる通信はすぐに広がり、2 年足らずで、ARPANET のトラフィックの 75 パーセントを占めるようになりました。

今日、標準化されたネットワークプロトコルをベースとする電子メールシステムは、インターネット上で最も幅広く使用されているサービスに進化しました。Red Hat Enterprise Linux には、電子メールに対応、アクセスする高度なアプリケーションが数多く備わっています。

本章では、今日使用されている最新の電子メールプロトコルと、電子メールの送受信を行うために設計されたプログラムについて概説します。

16.1. 電子メールプロトコル

今日、電子メールはクライアント/サーバーのアーキテクチャを使用して配信されています。電子メールのメッセージは、メールクライアントプログラムを使用して作成されます。次に、このプログラムが作成されたメッセージをサーバーに送信します。メッセージはサーバーにより受信者の電子メールサーバーに転送され、そこで受信者の電子メールクライアントへと渡されます。

このプロセスを有効にするために、各種の標準のネットワークプロトコルは、様々なマシン (多くの場合、異なるオペレーティングシステムを実行し、異なる電子メールプログラムを使用している) で電子メールの送受信を行えるようにします。

以下は、電子メールの転送に最も一般的に使用されているプロトコルです。

16.1.1. メール転送プロトコル

クライアントアプリケーションからサーバーへのメール配信、及び送信元サーバーから送信先サーバーへのメール配信は、*SMTP* (*Simple Mail Transfer Protocol 簡易メール転送プロトコル*) により処理されます。

16.1.1.1. SMTP

SMTP の第一の目的は、メールサーバー間における電子メールの転送ですが、これは電子メールクライアントにとっても極めて重要となります。電子メールを送信するには、クライアントはメッセージを送信メールサーバーに送信します。送信メールサーバーはこれを受けて、送信先のメールサーバーに配信の連絡を行います。このため、電子メールクライアントを設定する時には SMTP サーバーを指定する必要があります。

Red Hat Enterprise Linux では、ユーザーはメール配信を処理するようローカルマシン上の SMTP サーバーを設定することができます。ただし、送信メール用にリモート SMTP サーバーを設定することも可能です。

SMTP プロトコルに関して重要なのは認証が不要である点です。これにより、インターネット上の誰でも、個人や大規模なグループに対してでも電子メールを送信することができます。迷惑メールや *スパム* が可能になるのは SMTP のこうした特性が原因です。リレー制限を強制すると、インターネット上の

任意のユーザーが、ご使用の SMTP サーバーを介してインターネット上の別のサーバーへ電子メールを送信することを制限できます。リレー制限を強制していないサーバーは、*オープンリレー* サーバーと呼ばれます。

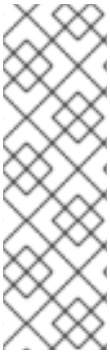
Red Hat Enterprise Linux には、Postfix 及び Sendmail SMTP プログラムが備わっています。

16.1.2. メールアクセスプロトコル

メールサーバーから電子メールを取得するために電子メールクライアントアプリケーションが使用する主要なプロトコルには、*POP (Post Office Protocol ポストオフィスプロトコル)* と *IMAP (Internet Message Access Protocol インターネットメッセージアクセスプロトコル)* の 2 つあります。

16.1.2.1. POP

Red Hat Enterprise Linux のデフォルトの POP サーバーは **Dovecot** で、dovecot パッケージにより提供されています。



注記

Dovecot を使用するには、初めに **dovecot** パッケージがご使用のシステムにインストールされているか次のコマンドを root で実行して確認します:

```
~]# yum install dovecot
```

Yum を使用したパッケージのインストールについての詳細は、「パッケージのインストール」 を参照して下さい。

POP サーバーを使用する場合、電子メールメッセージは電子メールクライアントのアプリケーションからダウンロードされます。デフォルトでは、ほとんどの **POP** 電子メールクライアントでは、電子メールサーバーのメッセージが正しく転送されるとそのメッセージは削除されるように自動的に設定されています。ただし、この設定は通常は変更できます。

POP は、電子メールのファイル添付を可能にする *MIME (Multipurpose Internet Mail Extensions 多目的インターネットメール拡張)* などの重要なインターネットメッセージング標準と完全な互換性があります。

POP が最適に機能する対象は、電子メールを読む単一のシステムを持つユーザーです。また、インターネットやメールサーバーを持つネットワークへの固定接続がないユーザーにも正しく機能します。残念ながら、速度が遅いネットワークを使用するユーザーの場合には、**POP** はクライアントプログラムに対して、認証を行った上で各メッセージのコンテンツ全体をダウンロードするよう要求します。このプロセスは、メッセージに大きなファイルが添付されている場合は長時間かかる場合があります。

標準 **POP** プロトコルの最新版は **POP3** です。

ただし、あまり使用されていない **POP** プロトコルのバリアントにも様々な種類があります:

- **APOP** — **MDS** (Monash Directory Service) 認証を使用した **POP3** です。暗号化されていないパスワードを送信するのではなく、エンコードされたユーザーパスワードのハッシュが電子メールクライアントからサーバーへ送信されます。
- **KPOP** — Kerberos 認証を使用した **POP3** です。

- **RPOP** — **RPOP** 認証を使用した **POP3** です。これは、パスワードに似たユーザーごとの ID を使用し、POP 要求を認証します。ただしこの ID は暗号化されていないため、**RPOP** のセキュリティレベルは標準 **POP** と同程度です。

セキュリティを強化するために、*SSL (Secure Socket Layer セキュアソケットレイヤ)* 暗号化をクライアント認証とデータ転送セッションに使用することが可能です。これは、**pop3s** サービスまたは **/usr/sbin/stunnel** アプリケーションを使用して有効にすることができます。詳細は 「通信のセキュリティ保護」 を参照してください。

16.1.2.2. IMAP

Red Hat Enterprise Linux でのデフォルト **IMAP** サーバーは **Dovecot** で、dovecot パッケージにあります。**Dovecot** のインストール方法については 「POP」 を参照してください。

IMAP メールサーバーを使用する場合、電子メールメッセージはサーバーに残るためユーザーは読み取り、削除を行うことができます。**IMAP** により、クライアントアプリケーションがサーバー上でメールディレクトリを作成、名前変更、削除を行い電子メールを整理、保存することもできます。

IMAP は複数のマシンを使って電子メールにアクセスするユーザーに特に役立ちます。このプロトコルでは、メッセージが開封されるまでは、電子メールのヘッダー情報しかダウンロードされず帯域幅を節減できるため、低速な接続でメールサーバーに接続するユーザーにも便利です。ユーザーは、メッセージを表示またはダウンロードすることなく削除することも可能です。

便宜のため、**IMAP** クライアントアプリケーションはメッセージのコピーをローカルでキャッシュすることが可能です。そのため、ユーザーは **IMAP** サーバーに直接接続していない時でも、既読メッセージを閲覧することができます。

IMAP は **POP** と同様に、電子メールのファイル添付を可能にする MIME などの重要なインターネットメッセージング標準と完全に互換性があります。

セキュリティを強化するために、**SSL** 暗号化をクライアント認証とデータ転送セッションに使用することが可能です。これは、**imaps** サービスまたは **/usr/sbin/stunnel** プログラムを使用して有効にすることができます。電子メール通信をセキュアにする詳細は 「通信のセキュリティ保護」 を参照してください。

無償や商用の IMAP クライアント及びサーバーは他にも提供されています。これらの多くは、IMAP プロトコルを拡張し、追加機能を提供します。

16.1.2.3. Dovecot

IMAP 及び **POP3** プロトコルを実装する **imap-login** 及び **pop3-login** プロセスは、dovecot パッケージに含まれているマスター **dovecot** デーモンにより発生します。**IMAP** 及び **POP** の使用は **/etc/dovecot/dovecot.conf** 設定ファイルで設定します。デフォルトでは **dovecot** は、**SSL** を使用して **IMAP** と **POP3** をセキュアなバージョンとともに実行します。**POP** を使用するよう **dovecot** を設定するには、次のステップを実行します。

1. **protocols** 変数がコメント解除されていて (行頭のハッシュ記号 (#) を削除)、**pop3** 引数を含むよう **/etc/dovecot/dovecot.conf** 設定ファイルを編集します。例えば以下のとおりです。

   ```
   protocols = imap imaps pop3 pop3s
   ```

   protocols 変数がコメントアウトされている場合は、**dovecot** はこの変数に指定されたデフォルト値を使用します。

2. この変更を現行のセッションで有効にするには、以下のコマンドを実行します:

```
~]# service dovecot restart
```

3. この変更を次のブート時に有効にするには、以下のコマンドを実行します:

```
~]# chkconfig dovecot on
```

注記

dovecot が報告するのは **IMAP** サーバーを起動したことだけですが、**POP3** サーバーも起動する点に注意してください。

SMTP とは違い、**IMAP** と **POP3** はユーザー名とパスワードを使用して接続するクライアントを認証する必要があります。デフォルトでは、両方のプロトコルのパスワードとも暗号化されていないネットワーク上で渡されます。

dovecot で **SSL** を編集するには、以下を実行します:

- **/etc/pki/dovecot/dovecot-openssl.conf** 設定ファイルを必要に応じて編集します。ただし、標準的なインストールではこのファイルへの変更は必要ありません。
- **/etc/pki/dovecot/certs/dovecot.pem** 及び **/etc/pki/dovecot/private/dovecot.pem** ファイルの名前変更、移動、削除を行います。
- **/usr/libexec/dovecot/mkcert.sh** のスクリプトを実行して、**dovecot** の自己署名証明書を作成します。証明書は **/etc/pki/dovecot/certs** 及び **/etc/pki/dovecot/private** ディレクトリにコピーされます。変更を有効にするには、**dovecot** を再起動します:

```
~]# service dovecot restart
```

dovecot の詳細は http://www.dovecot.org でオンラインで参照できます。

16.2. 電子メールプログラムの分類

一般的に、すべての電子メールアプリケーションは 3 つのタイプのうち 1 つ以上に分類されます。それぞれの分類は、電子メールメッセージの移動及び管理のプロセスにおいてそれぞれ特定の役割を果たします。大半のユーザーはメッセージの送受信に使用する特定の電子メールプログラムだけを認識しますが、電子メールを正しい送信先に届けるようにするにはすべての電子メールプログラムが重要です。

16.2.1. メール転送エージェント (Mail Transport Agent)

MTA (Mail Transport Agent メール転送エージェント) は、**SMTP** を使用してホスト間で電子メールメッセージを転送します。メッセージは目的の送信先に移動する時、様々な MTA に関わることがあります。

マシン間のメッセージ配信は、簡単に見えるかもしれませんが、配信のためにある MTA がメッセージを受け入れることが可能か、または受け入れるべきかを判断する過程全体は非常に複雑です。その上、スパムの問題により、特定の MTA の使用は通常 MTA の設定または MTA が常駐するネットワークのアクセス設定によって制限されます。

最新の電子メールクライアントプログラムの多くは、電子メールの送信を行う時に MTA として機能できます。ただし、このアクションが真の MTA の役割であると混同してはいけません。電子メールクライアントプログラムで、MTA のような電子メール送信が可能である唯一の理由は、アプリケーションを稼働しているホストに独自の MTA がないためです。これは非 UNIX ベースのオペレーティングシステム上の電子メールクライアントプログラムに特に当てはまります。ただし、これらのクライアントプログラムは、使用を許可されている MTA に対して送信メッセージを送信するだけで、目的の受信者の電子メールサーバーには直接メッセージを配信することはありません。

Red Hat Enterprise Linux では 2 つの MTA、*Postfix* と *Sendmail* が備わっているため、多くの場合、電子メールクライアントプログラムは MTA として機能する必要がありません。Red Hat Enterprise Linux には、*Fetchmail* と呼ばれる特別用途の MTA も装備されています。

Postfix、Sendmail、Fetchmail の詳細は 「メール転送エージェント (MTA)」 を参照してください。

16.2.2. メール配信エージェント (Mail Delivery Agent)

MDA (Mail Delivery Agent メール配信エージェント) は MTA により呼び出され、適切なユーザーのメールボックスに受信メールをファイル保存します。多くの場合、MDA は実際には **mail** や Procmail などの *LDA (Local Delivery Agent ローカル配信エージェント)* です。

電子メールクライアントアプリケーションが読み取り可能なところに配信されるメッセージを実際に処理するプログラムはいずれも MDA と見なすことができます。このため、一部の MTA (Sendmail、Postfix など) は、ローカルユーザーのメールスプールファイルに新規の電子メールメッセージを追加する時に、MDA の役割を果たすことができます。通常、MDA はシステム間でのメッセージの転送やユーザーインターフェースの提供は行いません。MDA は、ローカルマシン上でメッセージの配信と並べ替えを行い、電子メールクライアントアプリケーションがアクセスできるようにします。

16.2.3. メールユーザーエージェント (Mail User Agent)

MUA (Mail User Agent メールユーザーエージェント) は電子メールクライアントアプリケーションと同義語です。MUA により、ユーザーは最低限でも電子メールメッセージの読み取り、作成を行うことができます。多くの MUA では、**POP** または **IMAP** プロトコルを介したメッセージの取得や、メッセージを保管するメールボックスの設定、MTA への送信メッセージの送信ができます。

MUA は、**Evolution** のようなグラフィカルインターフェースの場合と、**pine** のようなシンプルなテキストベースのインターフェースの場合があります。

16.3. メール転送エージェント (MTA)

Red Hat Enterprise Linux には、Postfix と Sendmail の 2 つの主要な MTA が装備されています。Postfix はデフォルトの MTA として設定されていますが、デフォルトの MTA を Sendmail に切り替えることは簡単です。デフォルトの MTA を Sendmail に変更するには、Postfix をアンインストールするか、次のコマンドを使って Sendmail に切り替えます:

```
~]# alternatives --config mta
```

次のコマンドを使って、希望のサービスを有効/無効にすることもできます:

```
~]# chkconfig <service> <on/off>
```

16.3.1. Postfix

当初、IBM のセキュリティエキスパート/プログラマーの Wietse Venema 氏によって開発された Postfix は、Sendmail 互換の MTA で、セキュア、高速、かつ容易に設定できるように設計されています。

セキュリティを向上するために、Postfix ではモジューラ型設計を採用しており、権限が制限された小さなプロセスは、マスターデーモンにより起動されます。より小さく、権限の低いプロセスは、メール配信の様々な段階に関連する非常に特殊なタスクを実行して、ルートディレクトリが変更された環境で稼働し、攻撃の影響を抑えます。

Postfix がローカルコンピュータ以外のホストからのネットワーク接続を受け入れるよう設定するには、設定ファイルを多少変更するだけで可能です。さらに、より複雑なニーズのために、Postfix は様々な設定オプションだけでなくサードパーティのアドオンも提供するため、多用途でフル機能の MTA となっています。

Postfix の設定ファイルは人間に可読で、250 以上の指示文に対応しています。Sendmail とは異なり、変更を反映するためにマクロ処理は必要なく、また最も一般的に使用されるオプションの大部分は、多数のコメントが付いたファイルで説明されています。

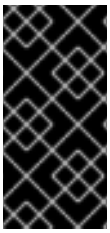
16.3.1.1. Postfix のデフォルトインストール

Postfix 実行可能ファイルは **/usr/sbin/postfix** です。このデーモンは、メール配信の処理に必要なすべての関連プロセスを起動します。

Postfix は設定ファイルを **/etc/postfix/** ディレクトリに格納します。以下は、一般的に使用されるその他のファイルの一覧です:

- **access** — アクセス制御に使用します。このファイルは、Postfix に接続可能なホストを指定します。
- **main.cf** — Postfix のグローバル設定ファイルです。このファイルには、設定オプションの大部分が指定されています。
- **master.cf** — メール配信を達成するために、Postfix が様々なプロセスとやりとりを行う方法を指定します。
- **transport** — 電子メールアドレスをリレーホストにマッピングします。

aliases ファイルは **/etc/** ディレクトリにあります。このファイルは Postfix と Sendmail 間で共有されます。ユーザー ID エイリアスを記述するメールプロトコルが必要な設定可能な一覧です。



重要

デフォルトの **/etc/postfix/main.cf** ファイルでは、Postfix はローカルコンピュータ以外のホストからのネットワーク接続を受け入れないように設定されています。Postfix を他のクライアント用のサーバーとして設定する方法は 「Postfix の基本設定」を参照してください。

/etc/postfix ディレクトリにある設定ファイルのオプションに変更を加えた後には、変更を反映させるために **postfix** サービスを再起動してください:

```
~]# service postfix restart
```

16.3.1.2. Postfix の基本設定

デフォルトでは、Postfix はローカルホスト以外のホストからのネットワーク接続を受け入れないように設定されています。ネットワーク上の他のホストを対象としたメール配信を有効にするには、root で以下の手順を実行します:

- **vi** などのテキストエディタで **/etc/postfix/main.cf** ファイルを編集します。
- **mydomain** 行のハッシュ記号 (#) を削除して、コメント解除してから、*domain.tld* の箇所を、メールサーバーがサービスを提供しているドメイン **example.com** などに置き換えます。
- **myorigin = $mydomain** 行のコメントを解除します。
- **myhostname** 行のコメントを解除し、*host.domain.tld* をマシンのホスト名に置き換えます。
- **mydestination = $myhostname, localhost.$mydomain** 行のコメントを解除します。
- **mynetworks** 行のコメントを解除して、*168.100.189.0/28* の箇所を、サーバーに接続可能なホスト用の有効なネットワーク設定に置き換えます。
- **inet_interfaces = all** 行のコメントを解除します。
- **inet_interfaces = localhost** をコメント化します。
- **postfix** サービスを再起動します。

これらの手順が完了したら、ホストは配信のため外部の電子メールを受け入れるようになります。

Postfix には様々な設定オプションがあります。Postfix の設定方法を学習する最適な方法の 1 つは、**/etc/postfix/main.cf** 設定ファイルのコメントを読むことです。Postfix 設定、SpamAssassin 統合、**/etc/postfix/main.cf** パラメータの詳細などの補足情報は http://www.postfix.org/ でオンラインで参照できます。

16.3.1.3. LDAP での Postfix の使用

Postfix は **LDAP** ディレクトリを様々なルックアップテーブルのソースとして利用できます (例えば **aliases**、**virtual**、**canonical** など)。これにより **LDAP** は階層的なユーザー情報を保存でき、Postfix は **LDAP** クエリの結果を必要な場合にのみ知らされます。この情報をローカルに保存しないことで、管理者は容易に管理を行うことができます。

16.3.1.3.1. /etc/aliases ルックアップのサンプル

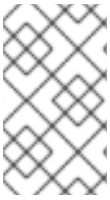
以下は **/etc/aliases** ファイルをルックアップする **LDAP** を使用する基本的なサンプルです。**/etc/postfix/main.cf** に次の内容が含まれていることを確認してください:

```
alias_maps = hash:/etc/aliases, ldap:/etc/postfix/ldap-aliases.cf
```

/etc/postfix/ldap-aliases.cf ファイルがなければ作成します。以下の内容を含むようにしてください:

```
server_host = ldap.example.com
search_base = dc=example, dc=com
```

ldap.example.com、***example***、***com*** パラメータは、既存の利用可能な **LDAP** サーバーの仕様と置き換える必要があります。



注記

/etc/postfix/ldap-aliases.cf ファイルは、**LDAP SSL** と **STARTTLS** を有効にするパラメーターなど様々なパラメーターを指定することができます。詳細は **ldap_table(5)** の man ページを参照してください。

LDAP の詳細は 「OpenLDAP」 を参照してください。

16.3.2. Sendmail

Sendmail の主要な目的は、他の MTA と同様に、通常 **SMTP** プロトコルを使用するホスト間で電子メールを安全に転送することです。ただし、Sendmail は高度に設定可能なため、使用するプロトコルをはじめとする電子メール処理のほぼすべての側面を制御することが可能です。多くのシステム管理者は、能力と拡張性から MTA に Sendmail の使用を選択しています。

16.3.2.1. 用途と制約

認識すべき重要な点は、Sendmail ができないことではなく、Sendmail が何であるか、何ができるのかということです。複数の役割を果たすモノリシックなアプリケーションの時代には、Sendmail は組織内で電子メールサーバーを稼働するのに必要な唯一のアプリケーションに思われるかもしれません。技術的にはこれは真実です。Sendmail はメールを各ユーザーのディレクトリにスプールして、ユーザーに送信メールを配信できるからです。しかし、実際は大半のユーザーは単なるメール配信以上の機能を必要とします。ユーザーは通常、**POP** または **IMAP** を使用する MUA で電子メールとやりとりを行い、ローカルマシンにメッセージをダウンロードする方法を望みます。あるいは、メールボックスへのアクセスに Web インターフェースを好むユーザーもいます。こうした他のアプリケーションを Sendmail と連動させることは可能ですが、実際、それらが存在する理由は異なり、独立して機能することができます。

Sendmail で設定すべきまたは設定できるすべての用途の説明は、本項の範囲外となります。Sendmail には文字通り数百におよぶ様々なオプションやルールセットがあるため、Sendmail のあらゆる能力や問題修正方法に関する専門的な資料があります。Sendmail に関するリソースの一覧は 「その他のリソース」 を参照してください。

本項では、Sendmail と共にデフォルトでインストール済みのファイルを概説し、迷惑メール (スパム) の停止方法や *LDAP* での Sendmail の拡張方法など基本設定の変更について説明します。

16.3.2.2. Sendmail のデフォルトのインストール

Sendmail を使用するには、root で以下を実行して、ご使用のシステムに sendmail パッケージがインストールされていることを確認します:

```
~]# yum install sendmail
```

Sendmail を設定するには、root で以下を実行して、ご使用のシステムに sendmail-cf パッケージがインストールされていることを確認します:

```
~]# yum install sendmail-cf
```

Yum を使用したパッケージのインストールについての詳細は、「パッケージのインストール」 を参照して下さい。

Sendmail を使用する前に、デフォルトの MTA が Postfix から切り替わっている必要があります。デフォルトの MTA の切り替え方法については 「メール転送エージェント (MTA)」 を参照してください。

Sendmail 実行可能ファイルは **/usr/sbin/sendmail** です。

Sendmail の長い詳細設定ファイルは **/etc/mail/sendmail.cf** です。**sendmail.cf** ファイルは直接編集しないようにしてください。Sendmail に設定変更を加えるには、**/etc/mail/sendmail.mc** ファイルを編集して、元の **/etc/mail/sendmail.cf** をバックアップした上で、以下のような別の方法で新規設定ファイルを生成します。

- **/etc/mail** に格納された makefile を使用して (**~]# make all -C /etc/mail/**)、新しい **/etc/mail/sendmail.cf** 設定ファイルを作成します。**/etc/mail** にある生成された他のすべてのファイル (db ファイル) は、必要に応じて再生成されます。旧 makemap コマンドは現在も使用可能です。make コマンドは、**service sendmail start | restart | reload** で自動的に使用されます。

- 別の方法として、**m4** マクロプロセッサを使用して新しい **/etc/mail/sendmail.cf** を作成することも可能です。**m4** マクロプロセッサはデフォルトではインストールされていません。**/etc/mail/sendmail.cf** の作成に使用する前に、root で m4 パッケージをインストールしてください。

  ```
  ~]# yum install m4
  ```

Sendmail の設定に関する詳細は 「Sendmail の一般的な設定変更」 を参照してください。

以下のような様々な Sendmail 設定ファイルが、**/etc/mail/** ディレクトリにインストールされています:

- **access** — Sendmail を送信電子メールに使用できるシステムを指定します。
- **domaintable** — ドメイン名のマッピングを指定します。
- **local-host-names** — ホストのエイリアスを指定します。
- **mailertable** — 特定のドメインのルーティングを上書きする方法を指定します。
- **virtusertable** — ドメイン固有のエイリアシング (aliasing) 形式を指定し、単一のマシン上における複数の仮想ドメインのホスティングを可能にします。

access、**domaintable**、**mailertable**、**virtusertable** など **/etc/mail/** にある設定ファイルの数点は、Sendmail が設定変更を利用できる前に、実際にそれらの情報をデータベースファイルに保存する必要があります。データベースファイル内のそうした設定への変更を含めるには、root で以下のコマンドを実行します:

```
~]# makemap hash /etc/mail/<name> < /etc/mail/<name>
```

<name> は更新される設定ファイル名を表します。場合によっては、変更を反映させるために以下を実行して **sendmail** サービスも再起動させる必要があります:

```
~]# service sendmail restart
```

例えば、**example.com** ドメイン宛のすべての電子メールが **bob@other-example.com** に配信されるようにするには、**virtusertable** ファイルに以下の行を追加します:

```
@example.com bob@other-example.com
```

変更を完了するには、**virtusertable.db** ファイルを更新する必要があります:

```
~]# makemap hash /etc/mail/virtusertable < /etc/mail/virtusertable
```

Sendmail は、新規設定を格納している更新済みの **virtusertable.db** ファイルを作成します。

16.3.2.3. Sendmail の一般的な設定変更

Sendmail 設定ファイルを変更する場合は、既存ファイルを編集せずに、全く新しい **/etc/mail/sendmail.cf** ファイルを生成するのが最適な方法です。

警告

sendmail.cf ファイルを変更する前に、バックアップコピーを作成しておいた方が良いでしょう。

希望する機能を Sendmail に追加したい場合は、root として **/etc/mail/sendmail.mc** ファイルを編集します。編集が終了したら、**sendmail** サービスを再起動します。m4 パッケージがインストールされている場合は、**m4** マクロプロセッサにより新しい **sendmail.cf** 設定ファイルが自動的に生成されます。

```
~]# service sendmail restart
```

重要

デフォルトの **sendmail.cf** ファイルでは、Sendmail はローカルコンピューター以外のホストからのネットワーク接続を受け入れないように設定されています。Sendmail を他のクライアント用のサーバーとして設定するには、**/etc/mail/sendmail.mc** ファイルを編集して、**DAEMON_OPTIONS** 指示文の **Addr=** オプションで指定されているアドレスを **127.0.0.1** からアクティブなネットワークデバイスの IP アドレスに変更するか、行頭に **dnl** を付けて、 **DAEMON_OPTIONS** 指示文を一斉にコメントアウトします。終了したら、サービスを再起動して **/etc/mail/sendmail.cf** を再生成します:

```
~]# service sendmail restart
```

Red Hat Enterprise Linux に同梱されているデフォルトの設定ファイルは、大半の **SMTP** 専用サイトで機能します。ただし、*UUCP* (*UNIX-to-UNIX Copy Protocol ユニックストゥユニックスコピープロトコル*) サイトでは機能しません。UUCP メール転送を使用する場合は、**/etc/mail/sendmail.mc** ファイルを再設定して、新しい **/etc/mail/sendmail.cf** ファイルを生成する必要があります。

/usr/share/sendmail-cf ディレクトリ下のディレクトリにあるファイルを編集する前には、**/usr/share/sendmail-cf/README** ファイルを確認してください。**/etc/mail/sendmail.cf** ファイルの今後の設定に影響を及ぼす場合があるためです。

16.3.2.4. マスカレード

一般的な Sendmail の設定の 1 つとして、単一のマシンがネットワーク上の全マシンのメールのゲートウェイとして機能するように設定する方法があります。例えば、ある企業が **mail.example.com** とい

う名前のマシンで、すべての電子メールを処理して、すべての送信メールに対して一貫した返信アドレスを割り当てたいとしましょう。

このような状況では、Sendmail サーバーは、返信アドレスが **user@host.example.com** ではなく **user@example.com** となるように、その企業のネットワーク上のマシン名をマスカレードする必要があります。

これを行うには、**/etc/mail/sendmail.mc** に以下の行を追加します:

```
FEATURE(always_add_domain)dnl
FEATURE(`masquerade_entire_domain')dnl
FEATURE(`masquerade_envelope')dnl
FEATURE(`allmasquerade')dnl
MASQUERADE_AS(`bigcorp.com.')dnl
MASQUERADE_DOMAIN(`bigcorp.com.')dnl
MASQUERADE_AS(bigcorp.com)dnl
```

m4 マクロプロセッサを使用して、新しい **sendmail.cf** を生成した後には、この設定によりネットワーク内のメールはすべて **bigcorp.com** から送信されたかのように表示されます。

16.3.2.5. スパムの防止

電子メールのスパムは、通信を要求したことがないユーザーから受信した、不要な迷惑メールと定義することができます。これは、破壊的で高コストを伴う、広く蔓延したインターネット通信標準の悪用です。

Sendmail を使うと、迷惑メールの送信に使用されている新たなスパム技術を比較的に簡単にブロックすることができます。さらに、数多くの一般的なスパム手法もデフォルトでブロックします。Sendmail で利用できる主要なアンチスパム機能は *ヘッダーのチェック*、*リレーの否認* (バージョン 8.9 からデフォルト)、*アクセスのデータベース*、*送信者情報の確認* です。

例えば、リレーとも呼ばれる **SMTP** メッセージの転送は、Sendmail バージョン 8.9 以降デフォルトでは無効になっています。このように変更される前には、Sendmail はメールホスト (**x.edu**) に対して、ある当事者 (**y.com**) からのメッセージを受け入れるよう指示し、そのメッセージを別の当事者 (**z.net**) に送信していました。しかし、現在は任意のドメインがサーバーを介してメールをリレーするよう Sendmail を設定する必要があります。リレードメインを設定するには、**/etc/mail/relay-domains** ファイルを編集して Sendmail を再起動してください。

```
~]# service sendmail restart
```

しかし、ユーザーはインターネット全体で他のサーバーから大量のスパムを受け取ることが何度もあります。そのような場合は、**/etc/mail/access** ファイルで利用可能な Sendmail のアクセス制御機能を使用して、迷惑なホストからの接続を阻止することができます。以下の例は、このファイルを使用したブロックと Sendmail サーバーへのアクセスを具体的に許可する方法を示しています:

```
badspammer.com ERROR:550 "Go away and do not spam us anymore"
tux.badspammer.com OK 10.0 RELAY
```

この例では、**badspammer.com** から送信された電子メールはいずれも 550 RFC-821 準拠のエラーコードでブロックされ、メッセージはスパム送信者に送り返されます。**tux.badspammer.com** のサブドメインから送信される電子メールは受け入れられます。最後の行には、10.0.*.* ネットワークから送信された電子メールがメールサーバーを通してリレー可能であることを示しています。

/etc/mail/access.db ファイルはデータベースであるため、**makemap** コマンドを使用して変更を有効にします。これを行うには、root として以下のコマンドを使用します:

```
~]# makemap hash /etc/mail/access < /etc/mail/access
```

メッセージヘッダーの分析により、ヘッダーのコンテンツに基づいたメール拒否が可能となります。**SMTP** サーバーは、電子メールの送信経路に関する情報をメッセージヘッダー内に保管します。メッセージがある MTA から別の MTA に送られると、その他すべての **Received** ヘッダーの上にそれぞれ **Received** ヘッダーを付けます。ただし、注意すべき重要なポイントは、この情報はスパム送信者が変更できるという点です。

上記の例は、アクセスの許可や阻止に関する Sendmail の機能のほんの一部です。詳細情報と例は **/usr/share/sendmail-cf/README** を参照してください。

Sendmail は、メールを配信する時に Procmail MDA を呼び出すため、SpamAssassin のようなスパムフィルタリングプログラムを使用して、ユーザー用にスパムを識別してファイル保存することも可能です。SpamAssassin 詳細については 「スパムフィルター」 を参照してください。

16.3.2.6. LDAP での Sendmail の使用

LDAP の使用は、大規模なグループからある特定のユーザーに関する特定の情報を検索する、非常に迅速かつ強力な方法です。例えば、**LDAP** サーバーを使用すると、一般的な企業ディレクトリから特定の電子メールアドレスをユーザーの姓で検索することができます。この種の実装では、**LDAP** はほどんど Sendmail から分離されており、**LDAP** が階層別のユーザー情報を保存して、Sendmail は事前にアドレスが入力された電子メールメッセージの形式で **LDAP** のクエリ結果を知らされるだけです。

ただし、Sendmail は **LDAP** との非常に大規模な統合をサポートします。この統合では、Sendmail は **LDAP** を使用して、中規模からエンタープライズレベルの組織をサポートするために連携する異なるメールサーバー上の **/etc/aliases** や **/etc/mail/virtusertables** などの個別に管理されているファイルを置き換えます。一言でいうならば、**LDAP** はメールルーティングレベルを Sendmail とその別個の設定ファイルから多くの様々なアプリケーションで活用できる強力な **LDAP** クラスタに抽象化します。

Sendmail の現行版は **LDAP** に対応しています。**LDAP** を使用して Sendmail を拡張するには、最初に **OpenLDAP** などの **LDAP** サーバーを稼働して、適切な設定を行います。次に、**/etc/mail/sendmail.mc** を編集して以下を追加します。

```
LDAPROUTE_DOMAIN('yourdomain.com')dnl
FEATURE('ldap_routing')dnl
```

注記

これは **LDAP** を使った非常に基本的な Sendmail の設定にすぎません。実際の設定は **LDAP** の実装によりこれとは大幅に異なる可能性があります。特に、共通の **LDAP** サーバーを使用するために数種の Sendmail マシンを設定する場合がそうです。

詳しい **LDAP** のルーティング設定に関する説明と例は、 **/usr/share/sendmail-cf/README** を参照してください。

次に、**m4** マクロプロセッサを実行し、再び Sendmail を再起動して、**/etc/mail/sendmail.cf** ファイルを再作成します。この方法については、「Sendmail の一般的な設定変更」 を参照してください。

LDAP の詳細は 「OpenLDAP」 を参照してください。

16.3.3. Fetchmail

Fetchmail は、リモートサーバーから電子メールを取得してローカルの MTA に配信する MTA です。多くのユーザーは、リモートサーバー上にあるメッセージをダウンロードするプロセスと、MUA で電子メールを読み取り、整理するプロセスを別々にする機能性を評価してます。ダイアルアップユーザーのニーズを踏まえて設計されている Fetchmail は、**POP3** や **IMAP** などの数々のプロトコルを使用して、メールスプールファイルに接続し、すべての電子メールメッセージを迅速にダウンロードします。また、必要に応じて、電子メールメッセージを **SMTP** サーバーに転送することもできます。

注記

Fetchmail を使用するには、root で以下を実行して、最初にご使用のシステムに fetchmail パッケージがインストールされていることを確認します:

```
~]# yum install fetchmail
```

Yum を使用したパッケージのインストールについての詳細は、「パッケージのインストール」 を参照して下さい。

Fetchmail は、ユーザーのホームディレクトリ内の **.fetchmailrc** ファイルを使用して各ユーザー用に設定されています。ない場合は、ホームディレクトリに **.fetchmailrc** ファイルを作成してください。

Fetchmail は **.fetchmailrc** ファイル内の詳細設定を使用して、リモートサーバー上にある電子メールを確認し、ダウンロードします。次に、電子メールをローカルマシン上のポート **25** に配信し、ローカルの MTA を使用して電子メールを適正なユーザーのスプールファイルに配置します。Procmail が利用できる場合は起動して電子メールをフィルターし、MUA が読み込むことができるようにメールボックスに配置します。

16.3.3.1. Fetchmail の設定オプション

Fetchmai を実行する時にすべての必要なオプションをコマンドライン上で渡し、リモートサーバー上の電子メールを確認することは可能ですが、**.fetchmailrc** ファイルを使用した方がはるかに簡単です。希望の設定オプションを **.fetchmailrc** ファイル内に配置し、それらのオプションが、**fetchmail** コマンドを発行する時に毎回使用されるようにします。Fetchmail の実行時にオプションを上書きしたい場合は、コマンドライン上でそのオプションを指定します。

ユーザーの **.fetchmailrc** ファイルには、3 つのクラスの設定オプションが格納されています。

- **グローバルオプション** — Fetchmail に、プログラムの動作を制御する、または電子メールを確認する全接続の設定を提供する指示をします。
- **サーバーオプション** — ポーリングされるサーバーに関する必要情報を指定します。ホスト名をはじめ、確認するポートやタイムアウトになるまでの秒数などの特定の電子メールサーバーの設定などです。こうしたオプションは、該当するサーバーを使用する全ユーザーに影響を及ぼします。
- **ユーザーオプション** — 指定された電子メールサーバーを使用して、電子メールの認証や確認を行うにあたって必要なユーザー名、パスワードなどの情報を格納します。

グローバルオプションは、**.fetchmailrc** ファイルの最上部に表示されます。その次には 1 つまたは複数のサーバーオプションがあり、それぞれが Fetchmail が確認すべき異なる電子メールサーバーを指定します。サーバーオプションの次には、その電子メールサーバーを確認する各ユーザーアカウントの

ユーザーオプションがあります。サーバーオプションと同様に、複数のユーザーオプションを指定することで特定のサーバーでの使用、同一サーバー上の複数の電子メールアカウントの確認を行うことができます。

サーバーオプションを **.fetchmailrc** ファイルで利用するには、サーバーの情報の先頭に **poll** または **skip** などの特別なオプションの動詞を使用します。**poll** アクションは、Fetchmail の実行時にこのサーバーオプションを使用して、指定されたユーザーオプションで電子メールを確認するよう Fetchmail に指示します。ただし、**skip** アクションの後にあるサーバーオプションは、Fetchmail が呼び出された時にサーバーのホスト名が指定されていない限り確認されません。**skip** オプションは、特定して呼び出された時にスキップされたサーバーのみを確認し、現在稼働中の設定には影響を及ぼさないため **.fetchmailrc** ファイルの設定をテストする時に役立ちます。

サンプルの **.fetchmailrc** ファイルは以下のとおりです:

```
set postmaster "user1"
set bouncemail

poll pop.domain.com proto pop3
    user 'user1' there with password 'secret' is user1 here

poll mail.domain2.com
    user 'user5' there with password 'secret2' is user1 here
    user 'user7' there with password 'secret3' is user1 here
```

この例では、グローバルオプションにより、最後手段としてユーザーに電子メールが送信されるように指定されており (**postmaster** オプション)、すべての電子メールエラーは送信者ではなく、ポストマスターに送信されます (**bouncemail** オプション)。**set** アクションは、この行にグローバルオプションが含まれていることを Fetchmail に伝えます。次に、2 つの電子メールサーバーが指定されます。1 つは **POP3** を使用して確認するように設定され、もう 1 つは様々なプロトコルを試みて機能するものを見つけるように設定されます。第 2 のサーバーオプションを使用して 2 人のユーザーが確認されますが、どのユーザー宛でも見つかった電子メールはすべて、**user1** のメールスプールに送信されます。これにより、単一の MUA 受信ボックスに表示させながら、複数のサーバー上で複数のメールボックスをチェックすることが可能となります。各ユーザーの固有の情報は、**user** アクションで開始します。

注記

ユーザーはパスワードを **.fetchmailrc** ファイルに配置する必要はありません。**with password '<password>'** のセクションを省略した場合は、Fetchmail は起動時にパスワードを要求するようになります。

Fetchmail には、数々のグローバルオプション、サーバーオプション、ローカルオプションがあります。これらの多くのオプションは、ほとんど使用されないか、非常に特定の状況のみに適用します。**fetchmail** の man ページに、各オプションの詳細が記載されていますが、最も一般的なものを以下の 3 つの項で説明します。

16.3.3.2. グローバルオプション

グローバルオプションは、**set** アクションの後に、それぞれ 1 行ずつ配置する必要があります。

- **daemon <*seconds*>** — Fetchmail がバックグラウンドに残るデーモンモードを指定します。*<seconds>* の箇所は、Fetchmail がサーバーをポーリングするまでの待機時間の秒数に置き換えます。

- **postmaster** — 配信に問題が生じた場合にローカルユーザーがメールを送信するよう指定します。

- **syslog** — エラーとステータスメッセージのログファイルを指定します。デフォルトは **/var/log/maillog** です。

16.3.3.3. サーバーオプション

サーバーオプションは **.fetchmailrc** 内で、**poll** または **skip** アクションの後にそれぞれの行に配置される必要があります。

- **auth *<auth-type>*** — *<auth-type>* の箇所を、使用する認証のタイプに置き換えます。デフォルトでは、**password** 認証が使用されますが、一部のプロトコルは **kerberos_v5**、**kerberos_v4** 及び **ssh** を含む他のタイプの認証をサポートしています。**any** の認証タイプを使用した場合、Fetchmail は最初にパスワードを必要としない方法を試みます。次に、パスワードをマスクする方法を試みた後、最後にサーバーに暗号化されていないパスワードを送信して認証を試みます。

- **interval *<number>*** — 指定されたサーバーを ***<number>*** の時間間隔でポーリングし、設定された全サーバー上の電子メールを確認します。このオプションは、通常ユーザーがほとんどメッセージを受信しない電子メールサーバーに使用されます。

- **port *<port-number>*** — *<port-number>* をポート番号に置き換えます。この値は、指定されたプロトコルのデフォルトのポート番号を上書きします。

- **proto *<protocol>*** — *<protocol>* の箇所は、サーバー上のメッセージを確認する時に使用する **pop3** や **imap** などのプロトコルに置き換えます。

- **timeout *<seconds>*** — *<seconds>* の箇所を、Fetchmail が接続の試行を打ち切った後でサーバーが非アクティブとなる秒数に置き換えます。この値が設定されていない場合は、デフォルトの **300** 秒が仮定されます。

16.3.3.4. ユーザーオプション

ユーザーオプションは、サーバーオプションの下の各行に配置される場合と、サーバーオプションと同じ行に配置される場合があります。いずれの場合も、定義されるオプションは **user** オプション (以下で説明) に従う必要があります。

- **fetchall** — 既読メッセージを含め Fetchmail がキューにあるすべてのメッセージをダウンロードするように命令します。デフォルトでは、Fetchmail は新規メッセージのみをダウンロードするようになっています。

- **fetchlimit *<number>*** — *<number>* の箇所は、停止する前に取得されるメッセージ数に置き換えます。

- **flush** — 新規メッセージを取得する前に、キューにあるすべての既読メッセージを削除します。

- **limit *<max-number-bytes>*** — *<max-number-bytes>* の箇所を、Fetchmail で取得する時に許容されているメッセージの最大バイトサイズに置き換えます。このオプションは、大型のメッセージのダウンロードに時間がかかりすぎる場合の低速のネットワークリンクに有用です。

- **password '*<password>*'** — *<password>* の箇所を、ユーザーのパスワードに置き換えます。

- **preconnect "<*command*>"** — *<command>* の箇所を、ユーザー宛のメッセージを取得する前に実行するコマンドに置き換えます。
- **postconnect "<*command*>"** — *<command>* の箇所を、ユーザー宛のメッセージを取得した後に実行するコマンドに置き換えます。
- **ssl** — SSL 暗号化をアクティベートします。
- **user "<*username*>"** — *<username>* の箇所を、Fetchmail がメッセージの取得に使用するユーザー名に置き換えます。**このオプションは、他のすべてのユーザーオプションの前に付ける必要があります。**

16.3.3.5. Fetchmail のコマンドオプション

fetchmail コマンドの実行時にコマンドライン上で使用される Fetchmail オプションの大半は **.fetchmailrc** 設定オプションを反映します。この方法では、Fetchmail は設定ファイルの有無を問わず使用することができます。 **.fetchmailrc** 設定オプションは、**.fetchmailrc** ファイルに残しておいた方が簡単なため、大半のユーザーはコマンドライン上では使用しません。

fetchmail コマンドは、特定の用途のオプションと併せて実行した方が望ましい場合もあります。コマンドラインで指定されるオプションはいずれも設定ファイルオプションを上書きするため、エラーが発生した場合は、コマンドオプションを使用してエラーの原因になっている **.fetchmailrc** 設定を一時的に上書きすることが可能です。

16.3.3.6. 情報提供またはデバッグのオプション

fetchmail コマンドの後に使用されるオプションの一部は、重要な情報を提供する場合があります。

- **--configdump** — **.fetchmailrc** 及び Fetchmail のデフォルト値からの情報に基づいた、取り得るオプションをすべて表示します。このオプションを使用すると、どのユーザーの電子メールも取得されません。
- **-s** — Fetchmail をサイレントモードで実行し、**fetchmail** コマンドの後にエラー以外のメッセージが表示されないようにします。
- **-v** — Fetchmail を verbose モードで実行し、Fetchmail とリモート電子メールサーバー間のすべての通信を表示します。
- **-V** — 詳細なバージョン情報の表示、グローバルオプションの一覧表示、電子メールプロトコルや認証メソッドなどの各ユーザーと使用する設定の表示を行います。このオプションを使用すると、どのユーザーの電子メールも取得されません。

16.3.3.7. 特別なオプション

これらのオプションは **.fetchmailrc** ファイルによく見られるデフォルト値を上書きする時に役立つ場合があります。

- **-a** — Fetchmail は、新規または既読を問わず、すべてのメッセージをリモートの電子メールサーバーからダウンロードします。デフォルトでは、Fetchmail は新規メッセージのみをダウンロードします。
- **-k** — Fetchmail はメッセージをダウンロードした後、リモートの電子メールサーバー上にメッセージを残します。このオプションを使用すると、メッセージをダウンロード後に削除するデフォルトの動作は上書きされます。

- **-l *<max-number-bytes>*** — Fetchmail は一定のサイズを超えるメッセージはダウンロードせず、リモートの電子メールサーバー上に残します。

- **--quit** — Fetchmail デーモンのプロセスを終了します。

その他のコマンドと **.fetchmailrc** オプションについては、**fetchmail** の man ページを参照してください。

16.3.4. MTA の設定

Mail Transport Agent (メール転送エージェント)(MTA) は電子メールの送信に不可欠です。**Evolution**、**Thunderbird**、**Mutt** などの *MUA* を使用して、メールの読み取り、作成を行うことができます。ユーザーが MUA から電子メールを送信すると、メッセージは MTA に引き渡されます。MTA は一連の MTA を通じて、メッセージが送信先に到着するまで送信します。

ユーザーがシステムから電子メールを送信する予定でなくても、一部の自動化されたタスクまたはシステムプログラムは **/bin/mail** コマンドを使用して、ログメッセージを含む電子メールをローカルシステムの root ユーザーに送信する可能性があります。

Red Hat Enterprise Linux 6 は、Sendmail と Postfix の 2 つの MTA を提供しています。どちらもインストールされている場合は、Postfix がデフォルトの MTA となります。

16.4. メール配信エージェント(MAIL DELIVERY AGENTS)(MDA)

Red Hat Enterprise Linux には Procmail と **mail** の 2 つの主要な MDA が含まれています。どちらのアプリケーションも LDA とみなされ、電子メールを MTA のスプールファイルからユーザーのメールボックスに移動します。ただし、Procmail の方が堅牢なフィルタリングシステムを提供します。

本項では、Procmail についてのみ詳しく説明します。**mail** コマンドの詳細は、man ページ (**man mail**) を参照してください。

電子メールがローカルホストのメールスプールファイルに配置されると、Procmail が配信とフィルタリングを行います。Procmail は強力な上、システムリソースの使用が低いため、幅広く利用されています。Procmail は電子メールクライアントアプリケーションによって読み取る電子メールを配信するという重要な役割を果たすことができます。

Procmail は、様々な方法で呼び出されることができます。MTA が電子メールをメールスプールファイルの中に配置すると常にProcmail が起動します。次に、Procmail は電子メールを MUA のためにフィルタリング、ファイル保存して、終了します。別の方法としては、メッセージを受信すると常に Procmail を実行するように MUA を設定し、メッセージが正しいメールボックスに移動するようにできます。デフォルトでは、**/etc/procmailrc** または **~/.procmailrc** ファイル (別名 *rc* ファイル) がユーザーのホームディレクトリに存在していると、MTA が新規メッセージを受信するといつでも Procmail を呼び出すようになっています。

デフォルトでは、**/etc/** ディレクトリにはシステム全体の **rc** ファイルは存在せず、ユーザーのホームディレクトリに **.procmailrc** ファイルは存在しません。このため、Procmail を使用するには、各ユーザーが特定の環境変数とルールを用いて **.procmailrc** ファイルを構築する必要があります。

Procmail が電子メールメッセージに対応するかどうかは、そのメッセージが **rc** ファイルの特定の条件または *レシピ* と適合するかどうかによって決まります。あるメッセージが任意のレシピと適合する場合は、電子メールは特定のファイルに配置、削除、それ以外は処理されます。

Procmail が起動すると、電子メールメッセージを読み込み、ヘッダー情報から本文を切り離します。次に、Procmail は **/etc/procmailrcs** ディレクトリ内の **/etc/procmailrc** ファイルと **rc** ファイルで、デフォルトのシステム全体の Pcocmail 環境用変数とレシピを探します。その後 Procmail は、ユー

ザーのホームディレクトリ内で **.procmailrc** ファイルを探します。多くのユーザーは、Procmail 用に追加の **rc** ファイルも作成します。これは、ホームディレクトリの **.procmailrc** ファイル内で参照されます。

16.4.1. Procmail の設定

Procmail の設定ファイルには、重要な環境変数が含まれています。これらの変数は、並べ替えするメッセージ、及びどのレシピとも適合しないメッセージの処理を指定します。

これらの環境変数は通常 **~/.procmailrc** ファイルの冒頭に、以下のような形式で表示されます。

```
<env-variable>="<value>"
```

この例では、***<env-variable>*** が変数の名前で、***<value>*** が変数を定義します。

大半の Procmail ユーザーが使用しない環境変数が多々あります。重要な環境変数の多くは、既にデフォルト値で定義されています。大抵の場合は、以下のような変数が使用されます。

- **DEFAULT** — どのレシピにも適合しないメッセージが配置された場合のデフォルトのメールボックスを設定します。

  デフォルトの **DEFAULT** 値は、**$ORGMAIL** と同じです。

- **INCLUDERC** — 照合するメッセージに対する多くのレシピを格納する追加の **rc** ファイルを指定します。これにより、Procmail レシピの一覧は、スパムのブロック、電子メール一覧の管理など異なる役割を果たす個別のファイルに分割されます。その結果、そうしたファイルは、ユーザーの **~/.procmailrc** ファイル内のコメント文字を使用して、オン/オフにすることができます。

  例えば、ユーザーの **.procmailrc** ファイル内の行は以下のようになります:

  ```
  MAILDIR=$HOME/Msgs INCLUDERC=$MAILDIR/lists.rc
  INCLUDERC=$MAILDIR/spam.rc
  ```

  電子メールの一覧の Procmail フィルターをオフにしつつスパム制御を維持したい場合は、最初の **INCLUDERC** の行をハッシュ記号 (#) でコメントアウトします。

- **LOCKSLEEP** — Procmail が特定のロックファイルの使用を試みる時間間隔を秒単位で設定します。デフォルトは **8** 秒です。

- **LOCKTIMEOUT** — ロックファイルが最後に修正された後、Procmail がそれは古くて削除可能であると見なすまでに経過する必要のある時間を秒単位で設定します。デフォルトは **1024** 秒です。

- **LOGFILE** — Procmail の情報やエラーメッセージが書き込まれるファイルです。

- **MAILDIR** — Procmail 用の現在作業中のディレクトリを設定します。設定されると、他の Procmail のパスはすべてこのディレクトリに対する相対パスになります。

- **ORGMAIL** — 元のメールボックス、またはデフォルトやレシピで必要な場所にメッセージを配置できなかった場合にメッセージを配置する別の場所を指定します。

  デフォルトでは、**/var/spool/mail/$LOGNAME** の値が使用されます。

- **SUSPEND** — スワップ領域など必要なリソースが利用できない場合に、Procmail が一時停止する時間を秒単位で設定します。
- **SWITCHRC** — 追加の Procmail レシピが格納されている外部ファイルをユーザーが指定できるようにします。これは、**INCLUDERC** オプションとよく似ていますが、レシピのチェックが参照先の設定ファイル上で実際に停止され、**SWITCHRC** の指定するファイル上のレシピのみが使用される点が異なります。
- **VERBOSE** — Procmail が詳細な情報をログ記録するようにします。このオプションはデバッグに役立ちます。

その他の重要な環境変数は、シェルから引き出されます。例えば、ログイン名の **LOGNAME**、ホームディレクトリの場所である **HOME**、デフォルトのシェルである **SHELL** などです。

すべての環境変数についての包括的な説明やデフォルト値については、**procmailrc** の man ページを参照してください。

16.4.2. Procmail のレシピ

多くの場合、新規ユーザーが Procmail の使用法を学習するにあたって最も難しいと感じるのは、レシピの構築です。レシピは、適合するストリングの条件を指定するために使用される特定の書式である *正規表現* を使用してメッセージ照合を行うため、これはある程度理解できます。ただ、正規表現の構築はそれほど難しくなく、読んで理解することも簡単です。その上、Procmail のレシピを書く方法は、正規表現にかかわらず一貫性があるため、例を使って学習すると簡単です。Procmail のレシピの例は、「レシピの例」 を参照してください。

Procmail レシピは以下の書式を使用します:

```
:0<flags>: <lockfile-name> * <special-condition-character>
        <condition-1> * <special-condition-character>
        <condition-2> * <special-condition-character>
        <condition-N>
        <special-action-character>
        <action-to-perform>
```

Procmail レシピの最初の 2 文字は、コロンとゼロです。ゼロの後に様々なフラグを配置して、Procmail がレシピを処理する方法を制御します。**<flags>** セクションの後ろにコロンを付けると、このメッセージ用にロックファイルが作成されることを示しています。ロックファイルが作成されると、その名前は **<lockfile-name>** の箇所を置き換えて指定することが可能です。

レシピは、メッセージと適合するための様々な条件を格納できます。条件がない場合は、あらゆるメッセージがレシピと適合することになります。正規表現は、メッセージ照合を容易にするために、一部の条件で使用されます。複数の条件を使用する場合、アクションが実行されるためにはすべてが適合しなければなりません。条件は、レシピの 1 行目に設定されているフラグに基づいてチェックされます。アスタリスク文字 (*) の後にオプションの特殊文字を配置すると、さらに条件を制御できます。

<action-to-perform> 引数は、メッセージが条件の 1 つに適合する場合にアクションを実行するよう指定します。1 つのレシピにつき 1 つのアクションのみとなります。多くの場合、メールボックスの名前がここで使用され、適合するメッセージをファイルに誘導し、電子メールを効果的に並べ替えます。特別なアクションの文字は、アクションが指定される前に使用することもできます。詳細は 「特別な条件とアクション」 を参照してください。

16.4.2.1. 配信レシピと非配信レシピの比較

レシピがある特定のメッセージと適合した場合に使用されるアクションにより、それが *配信* レシピ、

または *非配信* レシピと見なされるかが決定されます。配信レシピには、ファイルへのメッセージの書き込み、別のプログラムへのメッセージ送信、別の電子メールアドレスへのメッセージ転送などのアクションが含まれています。非配信レシピは、*ネストされたブロック* などその他のアクションを網羅します。ネストされたブロックは、中括弧 **{ }** で囲まれたアクションセットで、レシピの条件に適合するメッセージで実行されます。ネストされたブロックは、互いにネストさせることができるため、メッセージに対するアクションを特定、実行するにあたっての制御力が強化されます。

メッセージが配信レシピと適合すると、Procmail は指定されたアクションを実行し、その他のレシピとメッセージとの比較を停止します。非配信レシピと適合するメッセージの場合は、他のレシピに対する照合は継続されます。

16.4.2.2. フラグ

フラグは、レシピの条件をメッセージに照合する方法、またはそれを行うかどうかを決定するにあたって不可欠です。一般的に使用されるフラグは以下の通りです:

- **A** — **A** や **a** のフラグが付いていない以前のレシピもこのメッセージに適合する場合にのみ、このレシピが使用されることを指定します。
- **a** — **A** や **a** のフラグが付いた以前のレシピもこのメッセージに適合し、**かつ** 正常に完了した場合にのみこのレシピが使用されることを指定します。
- **B** — メッセージの本文を解析し、適合する条件を検索します。
- **b** — ファイルへのメッセージの書き込みや転送など、結果として生じるアクションにその本文を使用します。これはデフォルトの動作です。
- **c** — 電子メールのカーボンコピーを生成します。必要なアクションをメッセージで実行し、メッセージのコピーは **rc** のファイル内で引き続き処理することができるため、レシピの配信に役立ちます。
- **D** — **egrep** の照合で大文字と小文字を区別します。デフォルトでは、照合プロセスでは大文字と小文字を区別していません。
- **E** — **A** フラグと類似していますが、レシピ内の条件は、直前にある **E** フラグなしのレシピが適合しない場合のみに、メッセージと照合されます。これは *else* アクションと類似しています。
- **e** — 直前のレシピで指定されたアクションが失敗した場合のみ、レシピがメッセージに照合されます。
- **f** — フィルターとしてパイプを使用します。
- **H** — メッセージのヘッダーを解析し、適合する条件を検索します。これはデフォルトの動作です。
- **h** — 結果として生じるアクションでヘッダーを使用します。これはデフォルトの動作です。
- **w** — Procmail に対して、指定されたフィルターまたはプログラムが終了するのを待ち、メッセージがフィルターされたと見なす前に正常に終了したかどうかを報告するよう指示します。
- **W** — 「プログラム障害」のメッセージが抑制されている点を除いては **w** と全く同じです。

その他のフラグの詳細な一覧は、**procmailrc** の man ページを参照してください。

16.4.2.3. ローカルロックファイルの指定

ロックファイルは、Procmail で複数のプロセスが 1 つのメッセージを同時に変更しないようにするために非常に役立ちます。ローカルロックファイルを指定するには、レシピの 1 行目の任意のフラグの後にコロン (**:**) を配置します。これにより、送信先のファイル名に基づいたローカルロックファイルと、**LOCKEXT** のグローバル環境変数で設定されたものすべてが作成されます。

別の方法としては、このレシピで使用するローカルロックファイルの名前をコロンの後に指定します。

16.4.2.4. 特別な条件とアクション

Procmail レシピの条件とアクションの前に使用される特殊文字により、解釈の仕方が変わります。

以下の文字は、レシピの条件の行頭でアスタリスク文字 (*****) の後に使用できます:

- **!** — 条件の行では、この文字により条件が反転し、条件がメッセージに一致しない場合のみに適合が発生するようになります。
- **<** — メッセージが指定されたバイト数内に収まっているかどうかを確認します。
- **>** — メッセージが指定されたバイト数を超えているかどうかを確認します。

以下の文字は、特別なアクションを実行するために使用されます:

- **!** — アクションの行では、この文字は Procmail にメッセージを指定された電子メールアドレスに転送するように指示します。
- **$** — **rc** ファイルで以前に設定された変数を参照します。多くの場合は、様々なレシピによって参照される共通のメールボックスを設定するのに使用されます。
- **|** — メッセージを処理するための特定のプログラムを起動します。
- **{** と **}** — 適合するメッセージに適用する追加のレシピを格納するのに使用される、ネストされたブロックを構築します。

アクションの行頭に特殊文字を使用しない場合、Procmail はアクションの行がメッセージを書き込むためのメールボックスを指定していると仮定します。

16.4.2.5. レシピの例

Procmail は極めて柔軟性の高いプログラムですが、この柔軟性が原因で、新規ユーザーが Procmail のレシピを一から作成するのが難しい場合があります。

Procmail レシピの条件を構築するスキルを向上させる最適な方法は、正規表現をしっかり理解することに加えて、他の人が構築した多くの例を参照することから始まります。正規表現についての詳細な説明は、本項の範囲外となります。Procmail のレシピの構造と役立つ Procmail のサンプルレシピは、インターネット上の様々なところに掲載されています (http://www.iki.fi/era/procmail/links.html など)。正規表現の適切な使用、調整方法は、これらのレシピ例を参照してください。また、基礎的な正規表現のルールに関する初歩的な情報は、**grep** の man ページを参照してください。

以下にあげる簡単な例は、Procmail のレシピの基本構造を記載しており、構造をさらに複雑にするための基盤を示しています。

以下の例に示したように、基本的なレシピには条件さえも含まれていません:

```
:0: new-mail.spool
```

1 行目では、ローカルロックファイルが作成されたことは指定していますが、名前は指定していません。そのため、Procmail は送信先のファイル名を使用して **LOCKEXT** 環境変数で指定された値を追加します。条件が指定されていないため、すべてのメッセージがこのレシピに適合し、**MAILDIR** 環境変数によって指定されたディレクトリ内にある **new-mail.spool** と呼ばれる単一のスプールファイルに配置されます。その後、MUA はこのファイル内のメッセージを閲覧できるようになります。

このような基本レシピは、**rc** ファイルの末尾に配置され、メッセージをデフォルトの場所に誘導します。

以下の例では、特定の電子メールアドレスからのメッセージを照合して、削除します。

```
:0 * ^From: spammer@domain.com /dev/null
```

この例では、**spammer@domain.com** から送信されたメッセージはすべて **/dev/null** デバイスに送信され、削除されます。

警告

メッセージを **/dev/null** に送信して永久に削除してしまう前に、ルールが目的通りに機能していることを確認してください。レシピが間違えて目的以外のメッセージを対象にすると、それらのメッセージは消えてしまい、ルールのトラブルシューティングが困難になります。

この問題に対処する優れた方法としては、レシピのアクションを特別なメールボックスに移動させることです。これで、メールボックスを時折確認して、誤検知を探すことができます。メッセージが間違って適合されることがなく満足できる状態になったら、そのメールボックスは削除して、メッセージを **/dev/null** に送信するよう指示します。

以下のレシピでは、特定のメーリングリストから送信された電子メールを取得して、特定のフォルダに配置します。

```
:0: * ^(From|Cc|To).*tux-lug tuxlug
```

tux-lug@domain.com のメーリングリストから送信されたメッセージはすべて、MUA 用に自動的に **tuxlug** メールボックスに配置されます。**From**、**Cc**、**To** の行にメーリングリストの電子メールアドレスが入っている場合は、この例の条件がメッセージに適合する点に注意してください。

さらに詳しい強力なレシピについては、「その他のリソース」 の Procmail に関する数々のオンライン資料を参照してください。

16.4.2.6. スパムフィルター

Procmail は、新規の電子メールを受信すると Sendmail、Postfix、Fetchmail によって呼び出されるため、スパム対策の強力なツールとして使用できます。

これは、Procmail が SpamAssassin と併用された場合に特に有効です。これらの 2 つのアプリケーションを併用すると、スパムメールを迅速に特定して、並び替えまたは破棄することができます。

SpamAssassin はヘッダー分析、テキスト分析、ブラックリスト、スパム追跡データベース、自己学習型 Bayesian スパム分析を使用して、迅速かつ正確にスパムの特定とタグ付けを行います。

注記

SpamAssassin を使用するには、root で以下を実行して、最初にご使用のシステムに spamassassin パッケージがインストールされていることを確認します:

```
~]# yum install spamassassin
```

Yum を使用したパッケージのインストールについての詳細は、「パッケージのインストール」 を参照して下さい。

ローカルユーザーが SpamAssassin を使用する最も簡単な方法は、**~/.procmailrc** ファイルの最上部付近に以下の行を配置することです:

```
INCLUDERC=/etc/mail/spamassassin/spamassassin-default.rc
```

/etc/mail/spamassassin/spamassassin-default.rc には、シンプルな Procmail ルールが格納されており、受信するすべての電子メールに対して SpamAssassin をアクティベートします。電子メールがスパムであると判断された場合には、ヘッダー内でタグ付けされて、タイトルの先頭には以下のようなパターンが追加されます。

```
*****SPAM*****
```

電子メールのメッセージ本文にも、スパム診断の理由となった要素の継続的な記録が先頭に追加されます。

スパムとしてタグ付けされた電子メールをファイル保存するには、以下と同様のルールを使用することができます:

```
:0 Hw * ^X-Spam-Status: Yes spam
```

このルールにより、スパムとしてヘッダーにタグ付けされた電子メールはすべて、**spam** と呼ばれるメールボックスにファイル保存されます。

SpamAssassin は Perl スクリプトであるため、ビジー状態のサーバーではバイナリ SpamAssassin デーモン (**spamd**) とクライアントアプリケーション (**spamc**) を使用する必要がある場合があります。ただし、SpamAssassin をこのように設定するには、ホストへの root のアクセスが必要です。

spamd デーモンを起動するには、以下のコマンドを入力します:

```
~]# service spamassassin start
```

システムのブート時に SpamAssassin デーモンを起動するには、**サービス設定ツール** (**system-config-services**) などの initscript ユーティリティを使用して、**spamassassin** サービスを有効にします。サービスの起動と停止に関する詳細は、10章 *サービスとデーモン* を参照してください。

Procmail が Perl スクリプトではなく、SpamAssassin クライアントアプリケーションを使用するように設定するには、**~/.procmailrc** ファイルの最上部付近に以下の行を配置します。システム全体の設定の場合は、**/etc/procmailrc** に配置してください。

```
INCLUDERC=/etc/mail/spamassassin/spamassassin-spamc.rc
```

16.5. メールユーザーエージェント (MAIL USER AGENTS)

Red Hat Enterprise Linux には、多数の利用可能な電子メールプログラムがあります。**Evolution** のようなグラフィカル電子メールクライアントプログラムと、**mutt** のようなテキストベースの電子メールプログラムがあります。

本項ではこれ以降、クライアントとサーバー間の通信のセキュリティ保護について重点的に説明していきます。

16.5.1. 通信のセキュリティ保護

Evolution や **mutt** などの Red Hat Enterprise Linux に装備されている評判の高い MUA は、SSL 暗号化電子メールセッションを提供します。

暗号化されていないネットワークを行き来するその他のサービスと同様に、ユーザー名、パスワード、メッセージ全体などの電子メールに関する重要な情報は、ネットワーク上のユーザーによって傍受、閲覧される可能性があります。また、標準の **POP** 及び **IMAP** プロトコルは、認証情報を暗号化せずに渡すため、ユーザー名とパスワードはネームサーバー上で渡される時に攻撃者はそれらを収集して、ユーザーのアカウントに侵入することが可能です。

16.5.1.1. セキュアな電子メールクライアント

リモートサーバー上の電子メールを確認するように設計されている Linux MUA のほとんどは、SSL 暗号化に対応しています。電子メールを取得する時に SSL を使用するためには、SSL は電子メールクライアントとサーバーの両方で有効である必要があります。

SSL をクライアント側で有効にするのは簡単です。多くの場合は、MUA の設定ウインドウでボタンをクリックするか、MUA の設定ファイルのオプションで有効にすることができます。セキュアな **IMAP** と **POP** には既知のポート番号 (それぞれ **993** と **995**) があり、MUA はそれらを使用してメッセージの認証、ダウンロードを行います。

16.5.1.2. 電子メールクライアントの通信のセキュリティ保護

電子メールサーバー上の **IMAP** 及び **POP** ユーザーに SSL 暗号化を行うことはごく簡単です。

最初に SSL 証明書を作成します。これは、*認証局* (*CA*) に SSL 証明書を申請するか、自己署名証明書を作成するか、2 つの方法で行うことができます。

警告

自己署名証明書は、テスト目的のみで使用することをお勧めします。実稼働環境で使用するサーバーは、CA により交付された SSL 証明書を使用した方が良いでしょう。

IMAP や **POP** に対し自己署名 SSL 証明書を作成するには、**/etc/pki/dovecot/** ディレクトリに移動し、**/etc/pki/dovecot/dovecot-openssl.conf** 設定ファイルの証明書パラメータを希望に応じて編集し、root で次のコマンドを入力します:

```
dovecot]# rm -f certs/dovecot.pem private/dovecot.pem
dovecot]# /usr/libexec/dovecot/mkcert.sh
```

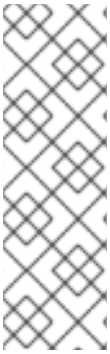
終了したら、**/etc/dovecot/conf.d/10-ssl.conf** ファイルに次の設定ファイルがあることを確認してください。

```
ssl_cert = </etc/pki/dovecot/certs/dovecot.pem
ssl_key = </etc/pki/dovecot/private/dovecot.pem
```

service dovecot restart コマンドを実行して、**dovecot** デーモンを再起動します。

別の方法として、**stunnel** コマンドを **IMAP** または **POP** サービスへの標準的なセキュアでない接続の周りに SSL 暗号化ラッパーとして使用することも可能です。

stunnel ユーティリティは、Red Hat Enterprise Linux に装備されている外部の OpenSSL ライブラリを使用して、強力な暗号化を実現し、ネットワーク接続を保護します。SSL 証明書を取得するためには、CA を申請することを推奨しますが、自己署名証明書を作成することも可能です。



注記

stunnel を使用するには、root で以下を実行して、最初にご使用のシステムに stunnel パッケージがインストールされていることを確認します:

```
~]# yum install stunnel
```

Yum を使用したパッケージのインストールについての詳細は、「パッケージのインストール」 を参照して下さい。

自己署名 SSL 証明書を作成するには、**/etc/pki/tls/certs/** ディレクトリに移動して、以下のコマンドを入力します:

```
certs]# make stunnel.pem
```

すべての質問に回答して、プロセスを完了します。

証明書が生成されたら、以下のコンテンツを含む **/etc/stunnel/mail.conf** のような **stunnel** 設定ファイルを作成します:

```
cert = /etc/pki/tls/certs/stunnel.pem

[pop3s]
accept  = 995
connect = 110

[imaps]
accept  = 993
connect = 143
```

/usr/bin/stunnel /etc/stunnel/mail.conf コマンドを使用して作成済みの設定ファイルで **stunnel** を起動したら、**IMAP** または **POP** 電子メールクライアントを使用し、SSL 暗号化によって電子メールサーバーに接続することが可能となります。

stunnel の詳細は **stunnel** の man ページ、または *<version-number>* が **stunnel** のバージョン番号である **/usr/share/doc/stunnel-<*version-number*>** ディレクトリのドキュメントを参照してください。

16.6. その他のリソース

以下は、電子メールアプリケーションに関する補足のドキュメントの一覧です。

16.6.1. インストールされているドキュメント

- Sendmail の設定に関する情報は、**sendmail** 及び **sendmail-cf** パッケージに含まれています。
  - **/usr/share/sendmail-cf/README** — **m4** マクロプロセッサの情報、Sendmail のファイルの場所、対応するメーラー、強化機能へのアクセス方法などの情報が記載されています。

  さらに、**sendmail** 及び **aliases** の man ページには役立つ情報が記載されており、Sendmail の様々なオプションや、Sendmail **/etc/mail/aliases** ファイルの適切な設定について説明しています。

- **/usr/share/doc/postfix-<*version-number*>** — Postfix の設定方法に関する多くの情報が含まれています。*<version-number>* の箇所を Postfix のバージョン番号に置き換えてください。

- **/usr/share/doc/fetchmail-<*version-number*>** — Fetchmail の機能の全一覧を記載した **FEATURES** ファイルと初歩的な **FAQ** ドキュメントが含まれています。*<version-number>* の箇所を Fetchmail のバージョン番号に置き換えます。

- **/usr/share/doc/procmail-<*version-number*>** — Procmail の概要を記載した **README** ファイル、各プログラム機能を詳細に記した **FEATURES** ファイル、設定に関する多数のよくある質問に対する回答をまとめた **FAQ** が含まれています。*<version-number>* の箇所を Procmail のバージョン番号に置き換えてください。

  Procmail の仕組みや新しいレシピの作成方法を学習する場合は、以下にあげる Procmail の man ページが非常に役立ちます:

  - **procmail** — Procmail の仕組みと電子メールのフィルタリングに必要な手順について概説しています。
  - **procmailrc** — レシピの構築に使用される **rc** のファイル形式について説明しています。
  - **procmailex** — 実環境向けの役立つ Procmail レシピの例を多数記載しています。
  - **procmailsc** — 特定のレシピとメッセージを適合するために Procmail で使用される重みつきスコアリング手法について説明しています。
  - **/usr/share/doc/spamassassin-<*version-number*>/** — SpamAssassin に関する多くの情報が含まれています。*<version-number>* の箇所を **spamassassin** パッケージのバージョン番号に置き換えてください。

16.6.2. 役立つ Web サイト

- http://www.sendmail.org/ — Sendmail の機能に関する完全な技術詳細、ドキュメント、設定例が記載されています。
- http://www.sendmail.com/ — Sendmail に関連したニュース、インタビュー、記事が掲載されており、利用可能な数多くのオプションの幅広い詳細が含まれています。
- http://www.postfix.org/ — Postfix プロジェクトのホームページで、Postfix に関する豊富な情報が掲載されています。メーリングリストは、特に情報検索に役立ちます。
- http://fetchmail.berlios.de/ — Fetchmail のホームページです。オンラインマニュアルと詳細な FAQ を参照できます。
- http://www.procmail.org/ — Procmail のホームページです。Procmail に特化した各種メーリングリストへのリンクや、様々な FAQ ドキュメントが記載されています。
- http://partmaps.org/era/procmail/mini-faq.html — Procmail の優れた FAQ です。トラブルシューティングのヒント、ファイルのロック、ワイルドカード文字の使用の詳細が記載されています。
- http://www.uwasa.fi/~ts/info/proctips.html — Procmail をさらに使い易くするための数々のヒントが掲載されています。**`.procmailrc`** ファイルのテスト方法や、Procmail スコアリングを使用して特定のアクションを実行すべきかどうかを判断する手順の説明が含まれています。
- http://www.spamassassin.org/ — SpamAssassin プロジェクトの公式サイトです。

16.6.3. 関連資料

- 『Sendmail Milters: A Guide for Fighting Spam』 (Bryan Costales、Marcia Flynt 共著、Addison-Wesley 社出版) — Sendmail の優れたガイドで、メールフィルターのカスタマイズに役立ちます。
- 『Sendmail』 (Bryan Costales、Eric Allman 他共著、O'Reilly & Associates 社出版) — Delivermail と Sendmail の考案者のサポートを受け執筆された優れた Sendmail の参考書籍です。
- 『Removing the Spam: Email Processing and Filtering』 (Geoff Mulligan 著、Addison-Wesley 社出版) — Sendmail や Procmail などの確立されたツールを用いてスパム問題に対処する電子メール管理者が使用する様々な手法を考察した書籍です。
- 『Internet Email Protocols: A Developer's Guide』 (Kevin Johnson 著、Addison-Wesley 社出版) — 主要な電メールプロトコルとそれらによるセキュリティに関する非常に詳細なレビューをまとめています。
- 『Managing IMAP』 (Dianna Mullet、Kevin Mullet 共著、O'Reilly & Associates 社出版) — IMAP サーバーの設定に必要な手順が詳述されています。

第17章 ディレクトリサーバー

17.1. OPENLDAP

LDAP (Lightweight Directory Access Protocol) は、ネットワーク上で中央に格納されている情報へアクセスするために使用するオープンプロトコルのセットです。これは、ディレクトリ共有のために **X.500** 標準をベースとしていますが、それほど複雑ではなくリソース集約型です。このため LDAP は 「X.500 Lite」 と呼ばれることがあります。

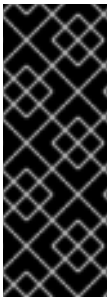
X.500 のように、LDAP はディレクトリを使用して階層式で情報を編成します。ディレクトリには、名前やアドレス、電話番号などの各種情報を格納することができ、また *ネットワーク情報サービス* (NIS) と似た方法で使用することもできます。このため、誰でも LDAP 対応ネットワーク上にあるマシンから自分のアカウントにアクセスできます。

LDAP の一般的な使用目的は、中央で管理されるユーザーとグループ、ユーザー認証、システム設定です。また、仮想の電話帳としても使用できるため、ユーザーは他のユーザーの連絡先情報に容易にアクセスできます。更に、LDAP は世界各地にある他の LDAP サーバーにユーザーを照会して、グローバルレポジトリに保存されているアドホックな情報を提供することが可能です。しかし LDAP は、大学や政府省庁、民間企業など個々の組織内で最もよく使用されます。

本項では、LDAPv2 及び LDAPv3 プロトコルのオープンソース実装である **OpenLDAP 2.4** をインストール/設定する方法について取り上げます。

17.1.1. LDAP の概要

LDAP は、クライアント/サーバーのアーキテクチャーを使用して、ネットワークからアクセス可能な中央情報ディレクトリを作成する信頼できる手段を提供します。クライアントがこのディレクトリ内の情報を修正しようとすると、サーバーはユーザーが修正できる権限を持っていることを確認します。その後、要求された通りにエントリを追加/更新します。通信の安全性を確保するために、*Secure Sockets Layer* (SSL) または *Transport Layer Security* (TLS) 暗号化プロトコルを使用して攻撃者が送信を傍受できないようにします。



重要

Red Hat Enterprise Linux 6 の OpenLDAP スイートは OpenSSL を使用しなくなりました。代わりに *ネットワークセキュリティサービス* (NSS) の Mozilla 実装を使用します。OpenLDAP は現行の証明書、キー、その他の TLS 設定による機能を継続します。Mozilla 証明書とキーデータベースを使用するための Mozilla NSS の設定方法については、『How do I use TLS/SSL with Mozilla NSS (Mozilla NSS で TLS/SSL を使用する方法)』 を参照して下さい。

LDAP サーバーは複数のデータベースシステムをサポートするため、管理者は適用する予定の情報のタイプに最適なソリューションを柔軟に選択できます。明確に定義された *アプリケーションプログラミングインターフェース* (API) クライアントにより、LDAP サーバーと通信できるアプリケーションは多く、その数と質ともに向上しています。

17.1.1.1. LDAP の用語

以下は、本章で使用されている LDAP 特有の用語一覧です。

エントリ

LDAP ディレクトリ内の単一ユニットです。各エントリは一意の DN *(Distinguished Name: 識別名)* で識別されます。

属性

エントリに直接関連付けられた情報です。例えば、ある組織が LDAP エントリとして表示されている場合、アドレス、ファックス番号などがこの組織に関連付けられた属性です。同様に、人も電話番号や電子メールアドレスのような共通の属性を持つエントリとして表示させることができます。

属性には、単一値か順不同の空白で区切られた値の一覧のどちらかがあります。一部の属性はオプションですが、その他は必須です。必須の属性は **objectClass** 定義を使用して指定され、**/etc/openldap/slapd.d/cn=config/cn=schema/** ディレクトリ内のスキーマファイル内にあります。

属性とそれに対応する値のアサーションは、RDN *(Relative Distinguished Name: 相対識別名)* とも呼ばれます。グローバルで一意の識別名とは異なり、相対識別名は 1 エントリごとに一意であるのみです。

LDIF

LDAP データ交換形式 (LDIF) は LDAP エントリをプレーンテキスト形式で表示したものです。以下のような形式を取ります:

```
[id] dn: distinguished_name
attribute_type: attribute_value…
attribute_type: attribute_value…
…
```

オプションの *id* は、エントリの編集に使用されるアプリケーションにより決定される番号です。*attribute_type* と *attribute_value* のペアが該当するスキーマファイルですべて定義されている限り、各エントリはそれらを必要な数だけ含むことができます。空白の行はエントリの終了を意味しています。

17.1.1.2. OpenLDAP の機能

OpenLDAP スイートは多くの重要な機能を提供します。

- **LDAPv3 サポート** — LDAP バージョン 2 以降に行われたプロトコルへの変更の多くは、LDAP をよりセキュアにするように設計されています。その他の改善点としては、SASL (Simple Authentication and Security Layer)、TLS 及び SSL プロトコルに対するサポートがあります。
- **IPC 上の LDAP** — プロセス間通信 (IPC) の使用により、ネットワーク上で通信する必要性をなくすことでセキュリティを強化します。
- **IPv6 サポート** — OpenLDAP は、次世代のインターネットプロトコルである Internet Protocol version 6 (IPv6) に準拠しています。
- **LDIFv1 サポート** — OpenLDAP は LDIF バージョン 1 に完全に準拠しています。
- **更新された C API** — 最新の C API はプログラマーが LDAP ディレクトリサーバーに接続して使用する方法を改善します。
- **強化されたスタンドアロン LDAP サーバー** — これには更新されたアクセス制御システム、スレッドプーリング、改善されたツールの他、多数が含まれます。

17.1.1.3. OpenLDAP サーバーの設定

Red Hat Enterprise Linux における LDAP サーバーの標準的な設定手順は、以下のとおりです。

1. OpenLDAP スイートをインストールします。必須パッケージの詳細については、「OpenLDAP スイートのインストール」 を参照して下さい。
2. 「OpenLDAP サーバーの設定」 で説明のとおりに設定をカスタマイズします。
3. 「OpenLDAP サーバーの実行」 で説明のとおりに **slapd** サービスを開始します。
4. **ldapadd** ユーティリティを使用して、エントリを LDAP ディレクトリに追加します。
5. **ldapsearch** ユーティリティを使用して、**slapd** サービスが情報に正しくアクセスしていることを確認します。

17.1.2. OpenLDAP スイートのインストール

OpenLDAP ライブラリとツールのスイートは、以下のパッケージにより提供されます。

表17.1 OpenLDAP パッケージの一覧

パッケージ	詳細
openldap	OpenLDAP サーバー/クライアントアプリケーションを実行するために必要なライブラリを含むパッケージです。
openldap-clients	LDAP サーバー上のディレクトリを表示/修正するためのコマンドラインユーティリティを含むパッケージです。
openldap-servers	LDAP サーバーを設定して実行するためのサービスとユーティリティの両方を含むパッケージです。これには、*スタンドアロン LDAP デーモン* である **slapd** が含まれます。
openldap-servers-sql	SQL サポートモジュールを含むパッケージです。
compat-openldap	OpenLDAP 互換ライブラリを含むパッケージです。

さらに、LDAP サーバーと共によく使用されるパッケージは以下のとおりです。

表17.2 一般的にインストールされるその他の LDAP パッケージの一覧

パッケージ	詳細
nss-pam-ldapd	ユーザーがローカルの LDAP クエリを実行できるようにするローカル LDAP ネームサービスである、**nslcd** を含むパッケージです。
mod_authz_ldap	Apache HTTP サーバー用の LDAP 認証モジュールである **mod_authz_ldap** を含むパッケージです。このモジュールは、サブジェクト用の短縮した識別名とクライアント SSL 証明書の発行者を使用して、LDAP ディレクトリ内のユーザーの識別名を決定します。また、ユーザーの LDAP ディレクトリエントリの属性を基にしてユーザーを承認して、アセットのユーザー/グループの権限を基にしてアセットへのアクセスを決定し、パスワードの有効期限が切れたユーザーのアクセスを拒否することもできます。**mod_ssl** モジュールは、**mod_authz_ldap** モジュールの使用時に必要である点に注意して下さい。

これらのパッケージをインストールするには、以下の形式で **yum** コマンドを使用します:

```
yum install package…
```

例えば、基本的な LDAP サーバーをインストールするには、シェルプロンプトで以下を入力します:

```
~]# yum install openldap openldap-clients openldap-servers
```

このコマンドを実行するには、スーパーユーザー権限 (つまり **root** でログインしている) が必要である点に注意して下さい。Red Hat Enterprise Linux に新しいパッケージをインストールする方法の詳細については、「パッケージのインストール」 を参照して下さい。

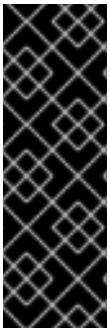
17.1.2.1. OpenLDAP サーバーユーティリティの概要

管理タスクを実行するために、openldap-servers パッケージは **slapd** サービスと共に以下のユーティリティをインストールします。

表17.3 OpenLDAP サーバーユーティリティの一覧

コマンド	詳細
slapacl	属性の一覧へのアクセスをチェックできるようにします。
slapadd	LDIF ファイルから LDAP ディレクトリへエントリを追加できるようにします。
slapauth	認証/承認のパーミッション用に ID の一覧をチェックできるようにします。
slapcat	LDAP ディレクトリからデフォルト形式でエントリをプルして、LDIF ファイル内に保存できるようにします。
slapdn	利用可能なスキーマ構文を基にして識別名 (DN) の一覧をチェックできるようにします。
slapindex	現在の内容を基にして **slapd** ディレクトリのインデックスを再構築できるようにします。設定ファイル内のインデックスオプションを変更する時は常にこのユーティリティを実行します。
slappasswd	**ldapmodify** ユーティリティと共に、あるいは **slapd** 設定ファイル内で使用される暗号化されたユーザーパスワードを作成できるようにします。
slapschema	データベースの整合性を対応するスキーマとチェックできるようにします。
slaptest	LDAP サーバーの設定をチェックできるようにします。

これらのユーティリティとその使用方法の詳しい説明については、「インストールされているドキュメント」 で参照されている該当の各 man ページをご覧下さい。



重要

root のみが **slapadd** を実行できますが、**slapd** サービスは **ldap** ユーザーとして実行します。このため、ディレクトリサーバーは **slapadd** によって作成されたファイルは修正することができません。この問題を解決するには、**slapd** ユーザーティリティの実行後に、シェルプロンプトで以下を入力します:

```
~]# chown -R ldap:ldap /var/lib/ldap
```

警告

データの整合性を維持するために、**slapadd**、**slapcat**、**slapindex** を使用する前に **slapd** サービスを停止します。シェルプロンプトで以下を入力します:

```
~]# service slapd stop
Stopping slapd:
[  OK  ]
```

slapd サービスの起動、停止、再起動及び現在のステータスの確認を行う方法の詳細については 「OpenLDAP サーバーの実行」 を参照して下さい。

17.1.2.2. OpenLDAP クライアントユーティリティの概要

openldap-clients パッケージは、LDAP ディレクトリ内のエントリを追加/修正/削除するために使用できる以下のようなユーティリティをインストールします。

表17.4 OpenLDAP クライアントユーティリティの一覧

コマンド	詳細
ldapadd	ファイルまたは標準入力からエントリを LDAP ディレクトリに追加できるようにします。これは **ldapmodify -a** へのシンボリックリンクです。
ldapcompare	任意の属性と LDAP ディレクトリのエントリを比較できるようにします。
ldapdelete	LDAP ディレクトリからエントリを削除できるようにします。
ldapexop	LDAP の拡張操作を実行できるようにします。
ldapmodify	LDAP ディレクトリ内のエントリをファイルまたは標準入力から修正できるようにします。
ldapmodrdn	LDAP ディレクトリエントリの RDN 値を修正できるようにします。

コマンド	詳細
ldappasswd	LDAP ユーザー用のパスワードを設定/変更できるようにします。
ldapsearch	LDAP ディレクトリエントリを検索できるようにします。
ldapurl	LDAP URL を生成/分解できるようにします。
ldapwhoami	LDAP サーバー上で **whoami** 動作を実行できるようにします。

ldapsearch は例外ですが、その他のユーティリティをより簡単に使用するためには、LDAP ディレクトリ内で変更される各エントリ用にコマンドを入力する代わりに、行われる変更を含むファイルを参照します。このようなファイルの形式は、各ユーティリティの man ページに要約されています。

17.1.2.3. 一般的な LDAP クライアントアプリケーションの概要

サーバー上のディレクトリを作成/修正できるグラフィカルな LDAP クライアントは各種存在しますが、どれも Red Hat Enterprise Linux には含まれていません。読み取り専用モードでディレクトリにアクセスできる人気のアプリケーションとしては、**Mozilla Thunderbird**、**Evolution**、**Ekiga** があります。

17.1.3. OpenLDAP サーバーの設定

デフォルトでは、OpenLDAP 設定は **/etc/openldap/** ディレクトリに格納されています。以下の表は、このディレクトリ内の最も重要なディレクトリとファイルを示しています:

表17.5 OpenLDAP 設定ファイルとディレクトリの一覧

パス	詳細
/etc/openldap/ldap.conf	OpenLDAP ライブラリを使用するクライアントアプリケーション用の設定ファイルです。これには **ldapadd**、**ldapsearch**、**Evolution** などが含まれます。
/etc/openldap/slapd.d/	**slapd** 設定を含むディレクトリです。

OpenLDAP はその設定を **/etc/openldap/slapd.conf** ファイルから読み取らなくなった点に注意して下さい。代わりに、**/etc/openldap/slapd.d/** ディレクトリに存在する設定データベースを使用します。以前のインストールから **slapd.conf** ファイルが残存している場合は、以下のコマンドを実行することで新しい形式に変換できます:

```
~]# slaptest -f /etc/openldap/slapd.conf -F /etc/openldap/slapd.d/
```

slapd 設定は、階層ディレクトリ構造で編成されている LDIF エントリで構成されています。これらのエントリを編集するための推奨される方法は 「OpenLDAP サーバーユーティリティの概要」 で説明のとおりサーバーユーティリティを使用することです。

重要

LDIF ファイル内でエラーが発生すると **slapd** サービスが起動できなくなることがあります。このため、**/etc/openldap/slapd.d/** 内の LDIF ファイルを直接編集しないことが強く推奨されます。

17.1.3.1. グローバル設定の変更

LDAP サーバーのグローバル設定オプションは、**/etc/openldap/slapd.d/cn=config.ldif** ファイル内に格納されています。一般的に使用される指示文は、以下のとおりです:

olcAllows

olcAllows 指示文を使用すると、有効にする機能を指定できます。以下の形式を取ります:

```
olcAllows: feature…
```

これは、表17.6「利用可能な olcAllows のオプション」 で説明のとおり空白で区切られた機能の一覧を使用します。デフォルトオプションは **bind_v2** です。

表17.6 利用可能な olcAllows のオプション

オプション	詳細
bind_v2	LDAP バージョン 2 のバインド要求の受け入れを有効にします。
bind_anon_cred	識別名(DN) が空欄である場合の匿名バインドを有効にします。
bind_anon_dn	識別名(DN) が空欄で **ない** 場合の匿名バインドを有効にします。
update_anon	匿名による更新操作の処理を有効にします。
proxy_authz_anon	匿名によるプロキシ認証制御の処理を有効にします。

例17.1 olcAllows 指示文の使用

```
olcAllows: bind_v2 update_anon
```

olcConnMaxPending

olcConnMaxPending 指示文を使用すると、匿名セッションに対する保留中の要求の最大数を指定できます。以下の形式を取ります:

```
olcConnMaxPending: number
```

デフォルトオプションは **100** です。

例17.2 olcConnMaxPending 指示文の使用

```
olcConnMaxPending: 100
```

olcConnMaxPendingAuth

olcConnMaxPendingAuth 指示文を使用すると、認証済みセッションに対する保留中の要求の最大数を指定できます。以下の形式を取ります:

```
olcConnMaxPendingAuth: number
```

デフォルトオプションは **1000** です。

例17.3 olcConnMaxPendingAuth 指示文の使用

```
olcConnMaxPendingAuth: 1000
```

olcDisallows

olcDisallows を使用すると、無効にする機能を指定できます。以下の形式を取ります:

```
olcDisallows: feature…
```

これは、表17.7「利用可能な olcDisallows のオプション」 で説明のとおり空白で区切られた機能の一覧を使用します。デフォルトでは、どの機能も無効ではありません。

表17.7 利用可能な olcDisallows のオプション

オプション	詳細
bind_anon	匿名バインド要求の受け入れを無効にします。
bind_simple	簡易バインド認証のメカニズムを無効にします。
tls_2_anon	STARTTLS コマンドの受け取り時に、匿名セッションの強制を無効にします。
tls_authc	認証時に、STARTTLS コマンドを許可しません。

例17.4 olcDisallows 指示文の使用

```
olcDisallows: bind_anon
```

olcIdleTimeout

olcIdleTimeout 指示文を使用すると、アイドル状態の接続を閉じるまでの待機秒数を指定できます。以下の形式を取ります:

```
olcIdleTimeout: number
```

このオプションは、デフォルトで無効です (すなわち **0** に設定されています)。

例17.5 olcIdleTimeout 指示文の使用

```
olcIdleTimeout: 180
```

olcLogFile

olcLogFile 指示文を使用すると、ログメッセージを書き込むファイルを指定できます。以下の形式を取ります:

```
olcLogFile: file_name
```

ログメッセージは、デフォルトで標準エラーに書き込まれます。

例17.6 olcLogFile 指示文の使用

```
olcLogFile: /var/log/slapd.log
```

olcReferral

olcReferral オプションを使用すると、サーバーが要求を処理できない場合に処理する別のサーバーの URL を指定できます。以下の形式を取ります:

```
olcReferral: URL
```

このオプションは、デフォルトで無効です。

例17.7 olcReferral 指示文の使用

```
olcReferral: ldap://root.openldap.org
```

olcWriteTimeout

olcWriteTimeout オプションを使用すると、未処理の書き込み要求がある接続を閉じるまでの待機秒数を指定できます。以下の形式を取ります:

```
olcWriteTimeout
```

このオプションは、デフォルトで無効です (すなわち **0** に設定されています)。

例17.8 olcWriteTimeout 指示文の使用

```
olcWriteTimeout: 180
```

17.1.3.2. データベース特有の設定の変更

デフォルトでは、OpenLDAP サーバーは Berkeley DB (BDB) をデータベースバックエンドとして使用

します。このデータベース用の設定は **/etc/openldap/slapd.d/cn=config/olcDatabase={1}bdb.ldif** ファイルに格納されています。データベース特有の設定でよく使用される指示文は、以下のとおりです:

olcReadOnly

olcReadOnly 指示文を使用すると、データベースを読み取り専用モードで使用できます。以下の形式を取ります:

```
olcReadOnly: boolean
```

これは、**TRUE** (読み取り専用モードを有効) / **FALSE** (データベースの修正を有効) を使用します。デフォルトのオプションは **FALSE** です。

例17.9 olcReadOnly 指示文の使用

```
olcReadOnly: TRUE
```

olcRootDN

olcRootDN 指示文を使用すると、アクセス制御により制限されないユーザーまたは LDAP ディレクトリの動作用に設定される管理制限パラメーターを指定できます。以下の形式を取ります:

```
olcRootDN: distinguished_name
```

これは、*識別名* (DN) を使用します。デフォルトのオプションは **cn=Manager,dn=my-domain,dc=com** です。

例17.10 olcRootDN 指示文の使用

```
olcRootDN: cn=root,dn=example,dn=com
```

olcRootPW

olcRootPW 指示文を使用すると、**olcRootDN** 指示文を使って指定されたユーザー用のパスワードを設定できます。以下の形式を取ります:

```
olcRootPW: password
```

これは、プレーンテキスト文字列かハッシュを使用します。ハッシュを生成するためには、シェルプロンプトで以下を入力します:

```
~]$ slappaswd
New password:
Re-enter new password:
{SSHA}WczWsyPEnMchFf1GRTweq2q7XJcvmSxD
```

例17.11 olcRootPW 指示文の使用

```
olcRootPW: {SSHA}WczWsyPEnMchFf1GRTweq2q7XJcvmSxD
```

olcSuffix

olcSuffix 指示文を使用すると、情報を提供するドメインを指定できます。以下の形式を取ります:

```
olcSuffix: domain_name
```

これは、*完全修飾ドメイン名* (FQDN) を使用します。デフォルトのオプションは **dc=my-domain,dc=com** です。

例17.12 olcSuffix 指示文の使用

```
olcSuffix: dc=example,dc=com
```

17.1.3.3. スキーマの拡張

OpenLDAP 2.3 以降、**/etc/openldap/slapd.d/** ディレクトリには、以前 **/etc/openldap/schema/** に配置されていた LDAP 定義も含まれています。デフォルトのスキーマファイルをガイドとして使用して追加の属性タイプとオブジェクトクラスをサポートするために、OpenLDAP により使用されるスキーマを拡張することができます。ただし、このタスクは本章の範囲外です。このトピックの詳細については http://www.openldap.org/doc/admin/schema.html を参照して下さい。

17.1.4. OpenLDAP サーバーの実行

本項では、**Standalone LDAP Daemon** (スタンドアロン LDAP デーモン) を起動、停止、再起動、現在のステータスを確認する方法について説明しています。システムサービスの管理方法全般については、10章 *サービスとデーモン* を参照して下さい。

17.1.4.1. サービスの開始

slapd サービスを実行するには、シェルプロンプトで以下を入力します:

```
~]# service slapd start
Starting slapd:                                            [  OK  ]
```

ブート時にサービスを自動的に起動させたい場合には、以下のコマンドを入力します:

```
~]# chkconfig slapd on
```

「サービスの有効化と無効化」 で説明のとおり、**サービスの設定** ユーティリティを使用することもできます。

17.1.4.2. サービスの停止

実行中の **slapd** サービスを停止するには、シェルプロンプトで以下を入力します:

```
~]# service slapd stop
Stopping slapd:                                            [  OK  ]
```

ブート時にサービスの自動開始を阻止したい場合には、以下を入力します:

```
~]# chkconfig slapd off
```

別の方法として、「サービスの有効化と無効化」 で説明のとおり **サービスの設定** ユーティリティを使用することもできます。

17.1.4.3. サービスの再開始

実行中の **slapd** サービスを再起動するには、シェルプロンプトで以下を入力します:

```
~]# service slapd restart
Stopping slapd:                                            [  OK  ]
Starting slapd:                                            [  OK  ]
```

これにより、サービスを停止して、再度開始します。設定を再ロードするには、このコマンドを使用します。

17.1.4.4. サービスステータスの確認

サービスが実行中であることを確認するには、シェルプロンプトで次のコマンドを入力します:

```
~]# service slapd status
slapd (pid  3672) is running...
```

17.1.5. OpenLDAP を使用した認証用のシステムの設定

OpenLDAP を使用して認証するようシステムを設定するためには、LDAP サーバーとクライアントマシンの両方に適切なパッケージがインストールされていることを確認します。サーバーの設定方法に関する詳細は、「OpenLDAP スイートのインストール」 と 「OpenLDAP サーバーの設定」 の手順に従って下さい。クライアントで、以下をシェルプロンプトで入力します:

```
~]# yum install openldap openldap-clients nss-pam-ldapd
```

11章*認証の設定* には、認証のために LDAP を使用するようアプリケーションを設定する方法の詳しい手順が記載されています。

17.1.5.1. 旧認証情報を LDAP 形式に移行

migrationtools パッケージは、認証情報を LDAP 形式に移行する時に役立つシェルスクリプトと Perl スクリプトのセットを提供します。このパッケージをインストールするには、シェルプロンプトで以下を入力します:

```
~]# yum install migrationtools
```

これにより、スクリプトを **/usr/share/migrationtools/** ディレクトリにインストールします。インストール後、**/usr/share/migrationtools/migrate_common.ph** ファイルを編集して、以下の行を変更して正しいドメインを反映させます。例えば:

```
# Default DNS domain
$DEFAULT_MAIL_DOMAIN = "example.com";
```

```
# Default base
$DEFAULT_BASE = "dc=example,dc=com";
```

別の方法として、コマンドライン上で環境変数を直接指定することもできます。例えば、**dc=example,dc=com** に設定されているデフォルトベースを使って **migrate_all_online.sh** スクリプトを実行するには、以下を入力します:

```
~]# export DEFAULT_BASE="dc=example,dc=com" \
/usr/share/migrationtools/migrate_all_online.sh
```

ユーザーデータベースを移行するために実行するスクリプトを決定するには、表17.8「一般的に使用される LDAP 移行スクリプト」 を参照して下さい。

表17.8 一般的に使用される LDAP 移行スクリプト

既存のネームサービス	LDAP は実行中ですか?	使用するスクリプト
/etc フラットファイル	はい	**migrate_all_online.sh**
/etc フラットファイル	いいえ	**migrate_all_offline.sh**
NetInfo	はい	**migrate_all_netinfo_online.sh**
NetInfo	いいえ	**migrate_all_netinfo_offline.sh**
NIS (YP)	はい	**migrate_all_nis_online.sh**
NIS (YP)	いいえ	**migrate_all_nis_offline.sh**

これらのスクリプトを使用する方法の詳細については、**/usr/share/doc/migrationtools-*version*/** ディレクトリ内の **README** 及び **migration-tools.txt** ファイルを参照して下さい。

17.1.6. その他のリソース

以下のリソースは、LDAP (Lightweight Directory Access Protocol) に関する追加情報を提供します。システムに LDAP を設定する前に、以下の資料、特に 『OpenLDAP Software Administrator's Guide (OpenLDAP ソフトウェア管理者ガイド)』 を確認することを強く推奨します。

17.1.6.1. インストールされているドキュメント

以下のドキュメントは、openldap-servers パッケージでインストールされます。

/usr/share/doc/openldap-servers-*version*/guide.html

『OpenLDAP Software Administrator's Guide』 のコピー

/usr/share/doc/openldap-servers-*version*/README.schema

インストールされているスキーマファイルの説明を含む README ファイル

更に、openldap、openldap-servers、openldap-clients パッケージでインストールされる多くの man ページがあります。

クライアントアプリケーション

- **man ldapadd** — LDAP ディレクトリにエントリを追加する方法を説明しています。
- **man ldapdelete** — LDAP ディレクトリからエントリを削除する方法を説明しています。
- **man ldapmodify** — LDAP ディレクトリ内でエントリを修正する方法を説明しています。
- **man ldapsearch** — LDAP ディレクトリ内でエントリを検索する方法を説明しています。
- **man ldappasswd** — LDAP ユーザーのパスワードを設定/変更する方法を説明しています。
- **man ldapcompare** — **ldapcompare** ツールの使用方法を説明しています。
- **man ldapwhoami** — **ldapwhoami** ツールの使用方法を説明しています。
- **man ldapmodrdn** — エントリの RDN を修正する方法を説明しています。

サーバーアプリケーション

- **man slapd** — LDAP サーバー用のコマンドラインオプションを説明しています。

管理アプリケーション

- **man slapadd** — **slapd** データベースにエントリを追加するために使用するコマンドラインオプションを説明しています。
- **man slapcat** — **slapd** データベースから LDIF ファイルを生成するために使用するコマンドラインオプションを説明しています。
- **man slapindex** — **slapd** データベースのコンテンツを基にしたインデックスの再生成に使用するコマンドラインオプションを説明しています。
- **man slappasswd** — LDAP ディレクトリ用のユーザーパスワードを生成するために使用するコマンドラインオプションを説明しています。

設定ファイル

- **man ldap.conf** — LDAP クライアント用の設定ファイル内で利用可能な形式とオプションを説明しています。
- **man slapd-config** — 設定ディレクトリ内で利用可能な形式とオプションを説明しています。

17.1.6.2. 役立つ Web サイト

http://www.openldap.org/doc/admin24/

『OpenLDAP Software Administrator's Guide (OpenLDAP ソフトウェア管理者ガイド)』 の現行バージョンです。

http://www.kingsmountain.com/ldapRoadmap.shtml

LDAP プロトコルに関する役立つリソースと最新ニュースへのリンクが含まれる Jeff Hodges 著の『LDAP Roadmap & FAQ』 です。

http://www.ldapman.org/articles/

ディレクトリツリーの設計とディレクトリ構造のカスタマイズを行う方法が含まれた、LDAP に関する優れた概要を提供する記事を集めたものです。

http://www.padl.com/

役立つ LDAP ツールが記載された開発者の Web サイトです。

17.1.6.3. 関連資料

『OpenLDAP by Example』 (John Terpstra、Benjamin Coles 共著、Prentice Hall 社出版)

OpenLDAP 導入における実用的な練習を集めたものです。

『Implementing LDAP』 (Mark Wilcox 著、Wrox Press, Inc. 社出版)

システム管理者とソフトウェア開発者の両者の観点から LDAP について取り上げた本です。

『Understanding and Deploying LDAP Directory Services』 (Tim Howes 他共著、Macmillan Technical Publishing 社出版)

実稼働環境における LDAP の導入とその設計原理を説明した本です。

第18章 ファイルとプリントサーバー

18.1. SAMBA

Samba は、*Server Message Block* (**SMB**) プロトコルのオープンソース実装です。Samba は、Microsoft Windows®、Linux、UNIX 及びその他のオペレーティングシステムが混在するネットワークの構築を可能とし、Windows ベースのファイルへのアクセス及びプリンター共有を可能にします。Samba は **SMB** を採用しているため、Windows クライアントに対しては Windows サーバーとして表示されます。

注記

Samba を使用するには、まず最初に root として以下のコマンドを実行し、samba パッケージをご使用のシステムに確実にインストールします:

```
~]# yum install samba
```

Yum を使用したパッケージのインストールについての詳細は、「パッケージのインストール」 を参照して下さい。

18.1.1. Samba の概要

Samba の第 3 のメジャーリリースであるバージョン 3.0.0 では、これまでのバージョンと比べて以下のような様々な点が改善されています。

- *Lightweight Directory Access Protocol* (**LDAP**) 及び *Kerberos* を使用した Active Directory ドメインへの参加
- 国際化のためのビルトイン Unicode サポート
- Microsoft Windows サーバー及びクライアントのすべての最新バージョンで、 ローカルレジストリのハッキングを必要としない Samba サーバーへの接続をサポート
- Samba.org チームが執筆した 400 ページ強のリファレンスマニュアルと、300 ページ強の実装/統合マニュアルの 2 つの新しいドキュメント。詳しい情報は、「関連資料」 を参照して下さい。

18.1.1.1. Samba の機能

Samba は強力かつ多用途なサーバーアプリケーションです。熟練のシステム管理者であっても、インストールと設定を行う前に、Samba の能力と限界について知っておく必要があります。

Samba の特徴

- Linux、UNIX、Windows クライアントに対するディレクトリツリーとプリンターのサービス
- ネットワークブラウジングのサポート (NetBIOS を使用または不使用)
- Windows ドメインログインの認証
- *Windows Internet Name Service* (**WINS**) ネームサーバー解決
- Windows NT® 式の Primary Domain Controller (PDC) として機能
- Samba ベースの PDC に対し *Backup Domain Controller* (BDC) として機能

- Active Directory ドメインメンバーサーバーとして機能
- Windows NT/2000/2003/2008 PDC への参加

Samba に搭載されていない機能

- Windows PDC に対する BDC としての機能 (及びその逆)
- Active Directory ドメインコントローラーとしての機能

18.1.2. Samba のデーモンと関連サービス

Samba の各種デーモンとサービスの概要は以下の通りです。

18.1.2.1. Samba のデーモン

Samba は、3 つのデーモン (**smbd**、**nmbd** 及び **winbindd**) で構成されています。3 つのサービス (**smb**、**nmb** 及び **winbind**) は、デーモンの起動、停止、その他のサービス関連機能を制御します。各デーモンの詳細やそのデーモンを制御する特定のサービスについて、以下に記載します。

smbd

smbd サーバーのデーモンは、Windows クライアントに対してファイル共有と印刷サービスを提供します。又、**SMB** プロトコルを介したユーザー認証、リソースのロック、データ共有も実行します。サーバーが **SMB** トラフィックをリッスンするデフォルトのポートは、**TCP** ポート、**139**、**445** です。

smbd デーモンは、**smb** サービスによって制御されます。

nmbd

nmbd サーバーのデーモンは、Windows ベースのシステムにおいて SMB/*Common Internet File System* (CIFS) によって生成されるような、NetBIOS ネームサービス要求を認識してそれに応答します。これらのシステムとして、Windows 95/98/ME、Windows NT、Windows 2000、Windows XP、LanManager クライアントがあります。又、このデーモンは、Windows **ネットワークコンピュータ (Network Neighborhood)** ビューを構成するブラウジングプロトコルにも関与しています。サーバーが **NMB** トラフィックをリッスンするデフォルトのポートは、**UDP** ポート **137** です。

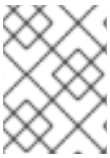
nmbd デーモンは、**nmb** サービスによって制御されます。

winbindd

winbind サービスは、Windows NT、2000、2003 または Windows Server 2008 を実行しているサーバー上のユーザーとグループの情報を解決するため、Windows ユーザー/グループの情報が、UNIX プラットフォームで認識可能となります。これは、Microsoft RPC コール、*Pluggable Authentication Modules* (PAM)、*Name Service Switch* (NSS) を使用することで可能です。これにより、Windows NT ドメインユーザーは、UNIX マシン上の UNIX ユーザーとして表示され、操作を行うこともできます。**winbind** サービス、Samba ディストリビューションにバンドルされていますが、**smb** サービスとは別に制御されます。

winbindd デーモンは **winbind** サービスによって制御されており、稼働するために **smb** サービスを起動する必要はありません。**winbindd** は、Samba が Active Directory メンバーである場合にも使用される他、Samba ドメインコントローラー上で (ネストされたグループ及び/またはドメイン間信頼関係の実装のために) 使用することも可能です。**winbind** は、Windows NT ベースのサーバーへの接続に使用されるクライアント側のサービスであるため、**winbind** の更に詳しい説明は、本マニュアルの範囲外となります。

認証に対する **winbind** の設定方法については、「Winbind 認証の設定」 を参照して下さい。



注記

Samba ディストリビューションに含まれているユーティリティの一覧は、「Samba ディストリビューションプログラム」 を参照して下さい。

18.1.3. Samba 共有への接続

Nautilus を使用して、ネットワーク上で利用可能な Samba 共有を表示することができます。ご使用のネットワーク上の Samba ワークグループとドメインの一覧を表示するには、GNOME パネルから **Places (場所)** → **Network (ネットワーク)** の順にクリックして、任意のネットワークを選択します。又、**Nautilus** の **File (ファイル)** → **Open Location (場所を開く)** のバーで **smb:** を入力して、ワークグループ/ドメインを表示することも可能です。

図18.1「Nautilus で表示される SMB ワークグループ」 に示されているように、ネットワーク上で利用可能な **SMB** ワークグループまたはドメインはそれぞれアイコンで表示されます。

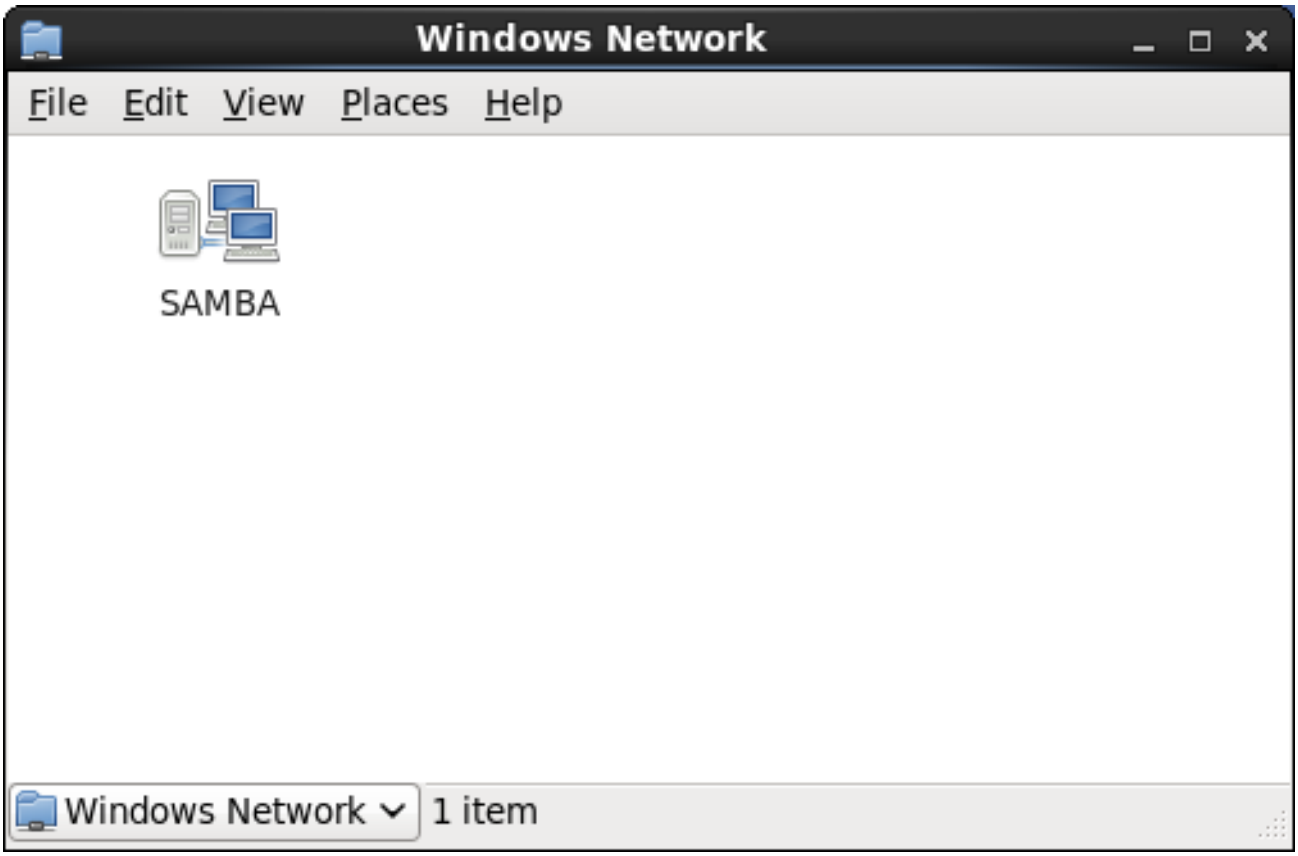


図18.1 Nautilus で表示される SMB ワークグループ

ワークグループ/ドメインのアイコンの一つをダブルクリックすると、そのワークグループ/ドメイン内にあるコンピューターの一覧が表示されます。

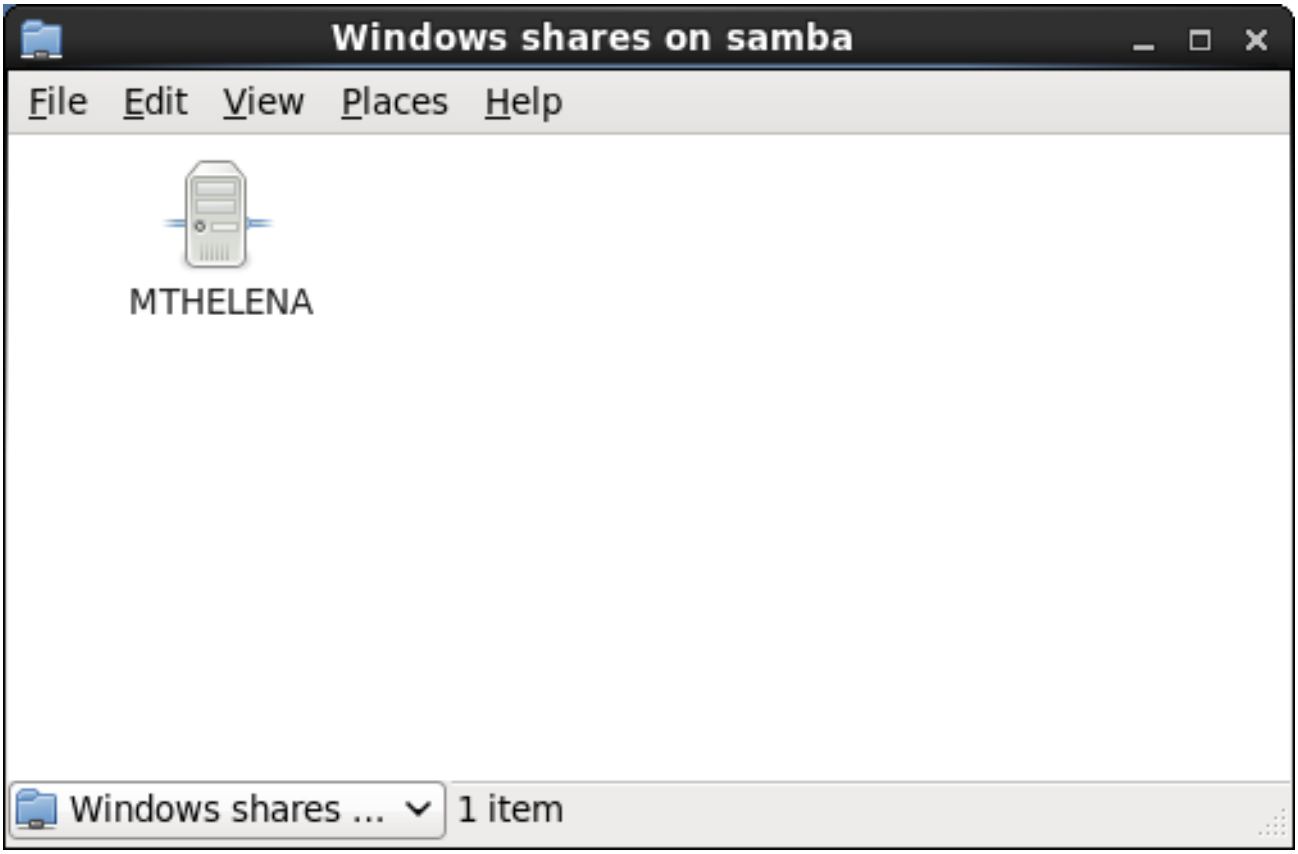


図18.2 Nautilus で表示される SMB マシン

図18.2「Nautilus で表示される SMB マシン」 に示されているように、ワークグループ内のマシン 1 台につき 1 つのアイコンが表示されます。アイコンをダブルクリックすると、そのマシン上の Samba 共有が表示されます。ユーザー名とパスワードの組み合わせが必要な場合は、プロンプトが表示されます。

又、以下のような構文を使用して、**Nautilus** の **場所** のバーで Samba サーバーと共有名を指定することも可能です。(*<servername>* と *<sharename>* は、適切な値に置き換えて下さい)。

```
smb://<servername>/<sharename>
```

18.1.3.1. コマンドライン

Samba サーバーのネットワークにクエリを行うには、**findsmb** コマンドを使用します。検出された各サーバーの **IP** アドレス、NetBIOS 名、ワークグループ名、オペレーティングシステム、**SMB** サーバーのバージョンが表示されます。

シェルプロンプトから Samba 共有に接続するには、以下のコマンドを入力します:

```
smbclient //<hostname>/<sharename> -U <username>
```

<hostname> の箇所を、接続先の Samba サーバーのホスト名または **IP** アドレスに置き換えます。
又、*<sharename>* は閲覧する共有ディレクトリ名に、*<username>* は対象システムの Samba ユーザー名に置き換えます。正しいパスワードを入力して下さい。パスワードが不要な場合には、**Enter** キーを押します。

smb:\> のプロンプトが表示されたら、ログインに成功したことになります。ログインしたら、**help** と入力して、コマンドの一覧を表示します。自分のホームディレクトリの内容を確認したい場合は、*sharename* を自分のユーザー名に置き換えて下さい。**-U** スイッチが使用されていない場合は、現

在のユーザーのユーザー名が Samba サーバーに渡されます。

smbclient を終了するには、**smb:\>** プロンプトで **exit** と入力します。

18.1.3.2. 共有のマウント

Samba 共有をディレクトリにマウントすると、そのディレクトリ内のファイルは、ローカルファイルシステムの一部であるかのように扱うことができるので、その方が便利な場合があります。

Samba 共有をディレクトリにマウントするには、マウントするためのディレクトリを新規作成し (まだ存在していない場合)、以下のコマンドを root で実行します:

```
~]# mount -t cifs //<servername>/<sharename> /mnt/point/ -o
username=<username>,password=<password>
```

このコマンドにより、 ローカルディレクトリ */mnt/point/* 内の *<servername>* から *<sharename>* がマウントされます。

注記

mount.cifs ユーティリティは、 (Samba には依存しない) 別の RPM です。**mount.cifs** を使用するには、まず最初に root として以下のコマンド実行し、cifs-utils パッケージをご使用のシステムに確実にインストールします:

```
~]# yum install cifs-utils
```

Yum を使用したパッケージのインストールについての詳細は、「パッケージのインストール」 を参照して下さい。

cifs-utils パッケージには、Kerberos 化した CIFS マウントを実行するためカーネルから呼び出された **cifs.upcall** バイナリも含まれます。**cifs.upcall** の詳細については、**man cifs.upcall** を参照して下さい。

Samba 共有のマウントに関する詳細は、**man mount.cifs** を参照して下さい。

警告

認証のためにプレーンテキストパスワードが必要な CIFS サーバーもあります。プレーンテキストパスワード認証のサポートを有効にするには、次のコマンドを実行します:

```
~]# echo 0x37 > /proc/fs/cifs/SecurityFlags
```

警告: この操作により、パスワードの暗号化を解除することでパスワードが漏洩する可能性があります。

18.1.4. Samba サーバーの設定

デフォルトの設定ファイル (**/etc/samba/smb.conf**) では、ユーザーは自分のホームディレクトリを Samba 共有として表示できるように設定されています。又、システム用に Samba 共有プリンターとして設定したすべてのプリンターを共有することもできます。つまり、システムにプリンターを接続して、ご使用のネットワーク上の Windows マシンから印刷することができます。

18.1.4.1. グラフィカル設定

グラフィカルインターフェースを使用して Samba を設定するには、利用可能な Samba のグラフィカルユーザーインターフェースを使用して下さい。現在利用可能な GUI の一覧は http://www.samba.org/samba/GUI/ をご覧下さい。

18.1.4.2. コマンドラインからの設定

Samba では、**/etc/samba/smb.conf** を設定ファイルとして使用します。この設定ファイルを変更した場合には、root として以下のコマンドを実行して Samba デーモンを再起動しないと、変更内容は有効になりません:

```
~]# service smb restart
```

Windows ワークグループと Samba サーバーの簡略な説明を指定するには、**/etc/samba/smb.conf** ファイルで以下の行を編集します。

```
workgroup = WORKGROUPNAME
server string = BRIEF COMMENT ABOUT SERVER
```

WORKGROUPNAME の箇所は、このマシンが所属すべき Windows ワークグループの名前に置き換えます。*BRIEF COMMENT ABOUT SERVER* は、Samba システムについての Windows のコメントとして使用されるオプションです。

ご使用の Linux システム上に Samba 共有ディレクトリを作成するには、**/etc/samba/smb.conf** ファイルに以下のセクションを (ご自分のニーズとご使用のシステムを反映させるよう変更してから) 追加して下さい:

```
[sharename]
comment = Insert a comment here
path = /home/share/
valid users = tfox carole
public = no
writable = yes
printable = no
create mask = 0765
```

上記の例では、**tfox** と **carole** の 2 人のユーザーが、Samba クライアントから Samba サーバー上の **/home/share** ディレクトリの読み取り/書き込みを行うことができます。

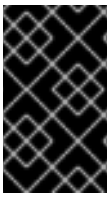
18.1.4.3. 暗号化されたパスワード

暗号化されたパスワードは、よりセキュアであるため、デフォルトで有効になっています。暗号化されたパスワードを使用するユーザーを作成するには、**smbpasswd -a *<username>*** のコマンドを使用します。

18.1.5. Samba の起動と停止

Samba サーバーを起動するには、root として、シェルプロンプトで以下のコマンドを入力します:

```
~]# service sshd start
```



重要

ドメインメンバーのサーバーを設定するには、**smb** サービスを開始する **前に net join** のコマンドを使用して、まずドメインまたは Active Directory に参加する必要があります。

サーバーを停止するには、root としてログインし、シェルプロンプトで以下のコマンドを入力します:

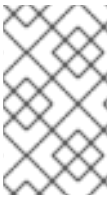
```
~]# service smb stop
```

restart オプションは、Samba を停止して起動する、迅速な方法です。これは、Samba の設定ファイルを編集した後に設定の変更内容を有効にする、最も信頼できる方法です。最初にデーモンが稼働していなかった場合でも、restart のオプションにより起動される点に注意して下さい。

サーバーを再起動するには、シェルプロンプトで root として以下のコマンドを入力します:

```
~]# service smb restart
```

condrestart (*条件付き再起動*) のオプションは、現在稼働中であるという条件でのみ **smb** を起動させます。このオプションは、デーモンが稼働していない場合には開始しないため、スクリプトに有用です。



注記

/etc/samba/smb.conf ファイルに変更が加えられた場合、Samba は数分後にそのファイルを自動的に再読み込みします。手動で **restart** または **reload** のコマンドを発行して設定ファイルを再読み込みすることもできます。

サーバーを条件付きで再起動させるには、root として以下のコマンドを入力します:

```
~]# service smb condrestart
```

手動での **/etc/samba/smb.conf** ファイルの再読み込みは、**smb** サービスによる自動再読み込みが失敗した場合に役立ちます。サービスを再起動させずに Samba サーバーの設定ファイルが確実に再読み込みされるようにするためには、root として以下のコマンドを入力します:

```
~]# service smb reload
```

デフォルトでは、**smb** サービスはブート時には自動的に起動 **しません**。Samba がブート時に起動するように設定するには、**/sbin/chkconfig**、**/usr/sbin/ntsysv** または **サービス設定ツール** プログラムなどの initscript ユーティリティを使用します。これらのツールに関する詳細は 10章 *サービスとデーモン* を参照して下さい。

18.1.6. Samba サーバーのタイプと smb.conf ファイル

Samba の設定はごく簡単です。Samba への変更はすべて **/etc/samba/smb.conf** 設定ファイルで行われます。デフォルトの **smb.conf** ファイルはしっかりと文書化されていますが、LDAP、Active Directory、数々のドメインコントローラー実装などの複雑なトピックには対応していません。

以下のセクションでは、Samba サーバーの異なる構成方法について説明しています。適切な構成を行うには、どのような要件があり、**/etc/samba/smb.conf** ファイルにどのような変更を加える必要があるかを考慮して下さい。

18.1.6.1. スタンドアロンサーバー

スタンドアロンサーバーは、ワークグループサーバーまたはワークグループ環境のメンバーにすることができます。スタンドアロンサーバーは、ドメインコントローラーではなく、ドメインには決して参加しません。以下の例には、複数の匿名の共有レベルセキュリティ設定と単一ユーザーレベルのセキュリティ設定が含まれています。共有レベル及びユーザーレベルのセキュリティモードに関する詳細は、「Samba のセキュリティモード」 を参照して下さい。

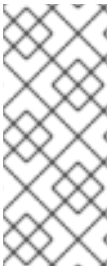
18.1.6.1.1. 匿名の読み取り専用

以下の **/etc/samba/smb.conf** ファイルには、匿名の読み取り専用のファイル共有を実装するにあたって必要な設定の例を示しています。**security = share** のパラメーターによって共有が匿名となります。単一の Samba サーバーでは、異なるセキュリティレベルを併用することはできない点に留意して下さい。**security** 指示文は、**/etc/samba/smb.conf** ファイルの **[global]** 設定セクションにある Samba のグローバルパラメーターです。

```
[global]
workgroup = DOCS
netbios name = DOCS_SRV
security = share
[data]
comment = Documentation Samba Server
path = /export
read only = Yes
guest only = Yes
```

18.1.6.1.2. 匿名の読み取り/書き込み

以下の **/etc/samba/smb.conf** ファイルには、匿名の読み取り/書き込みファイル共有を実装するにあたって必要な設定の例を示しています。匿名の読み取り/書き込みファイル共有を有効にするには、**read only** 指示文を **no** に設定します。**force user** と **force group** の指示文も追加して、共有内で指定されている、新たに配置されたファイルのオーナーシップを強制します。



注記

匿名の読み取り/書き込みサーバーを設定するのは可能ですが、推奨されません。共有スペースに配置されたファイルにはいずれも、ユーザーを問わず、**/etc/samba/smb.conf** ファイルで一般ユーザー (**force user**) とグループ (**force group**) によって指定されているユーザー/グループの組み合わせが割り当てられます。

```
[global]
workgroup = DOCS
netbios name = DOCS_SRV
security = share
```

```
[data]
comment = Data
path = /export
force user = docsbot
force group = users
read only = No
guest ok = Yes
```

18.1.6.1.3. 匿名のプリントサーバー

以下の **/etc/samba/smb.conf** ファイルには、匿名のプリントサーバーを実装するにあたって必要な設定の例を示しています。以下のように、**browseable** を **no** に設定すると、Windows の **ネットワークコンピュータ (Network Neighborhood)** には、そのプリンターは表示されません。このプリンターは、一覧では隠されていますが、明示的に設定することができます。NetBIOS を使用して、**DOCS_SRV** に接続することにより、クライアントは、プリンターにアクセスすることができます。但し、そのクライアントは **DOCS** ワークグループの一部でもあることが条件です。又、**use client driver** 指示文が **Yes** に設定されているため、クライアントに正しいローカルプリンタードライバーがインストール済みであることも前提となっています。この場合、Samba サーバーには、クライアントに対してプリンタードライバーを共有する責任はありません。

```
[global]
workgroup = DOCS
netbios name = DOCS_SRV
security = share
printcap name = cups
disable spools= Yes
show add printer wizard = No
printing = cups
[printers]
comment = All Printers
path = /var/spool/samba
guest ok = Yes
printable = Yes
use client driver = Yes
browseable = Yes
```

18.1.6.1.4. セキュアな読み取り/書き込みファイルとプリントサーバー

以下の **/etc/samba/smb.conf** ファイルでは、セキュアな読み取り/書き込みファイルとプリントサーバーの実装に必要な設定の例を示しています。**security** 指示文を **user** に設定すると、Samba はクライアントの接続の認証を行うように強制されます。**[homes]** の共有には、**[public]** 共有のような **force user** や **force group** の指示文はない点に注意して下さい。**[public]** の場合の **force user** と **force group** とは対照的に、**[homes]** の共有では、作成されるあらゆるファイルに認証済みユーザーの詳細情報が使用されます。

```
[global]
workgroup = DOCS
netbios name = DOCS_SRV
security = user
printcap name = cups
disable spools = Yes
show add printer wizard = No
printing = cups
[homes]
```

```
comment = Home Directories
valid users = %S
read only = No
browseable = No
[public]
comment = Data
path = /export
force user = docsbot
force group = users
guest ok = Yes
[printers]
comment = All Printers
path = /var/spool/samba
printer admin = john, ed, @admins
create mask = 0600
guest ok = Yes
printable = Yes
use client driver = Yes
browseable = Yes
```

18.1.6.2. ドメインメンバーサーバー

ドメインメンバーはスタンドアロンサーバーと類似していますが、ドメインコントローラー (Windows または Samba) にログインし、ドメインのセキュリティルールの対象となります。ドメインメンバーサーバーの例としては、Primary Domain Controller (PDC) 上にマシンアカウントを持つ、Samba を稼働させている部門サーバーがあげられます。その部門の全クライアントは依然として PDC で認証を行い、デスクトッププロファイル及びすべてのネットワークポリシーファイルが含まれます。相異点は、部門サーバーにはプリンターとネットワーク共有を制御する能力があるという点です。

18.1.6.2.1. Active Directory ドメインメンバーサーバー

以下の **/etc/samba/smb.conf** ファイルでは、Active Directory ドメインメンバーサーバーの実装に必要な設定の例を示しています。この例では、Samba がローカルで実行されているサービスに対してユーザー認証を行いますが、Active Directory のクライアントでもあります。Kerberos **realm** パラメーターは、必ずすべて大文字で記載してください (例：**realm = EXAMPLE.COM**)。Windows 2000/2003/2008 では、Active Directory の認証に Kerberos が必要とされるため、**realm** 指示文が必要です。Active Directory と Kerberos が異なるサーバーで稼働している場合は、それらを区別するために、**password server** 指示文が必要となる場合があります。

```
[global]
realm = EXAMPLE.COM
security = ADS
encrypt passwords = yes
# Optional. Use only if Samba cannot determine the Kerberos server
automatically.
password server = kerberos.example.com
```

メンバーサーバーを Active Directory ドメインに参加させるには、以下の手順を完了させる必要があります。

- メンバーサーバー上の **/etc/samba/smb.conf** ファイルの設定
- メンバーサーバー上の **/etc/krb5.conf** ファイルを含む Kerberos の設定
- Active Directory ドメインサーバー上のマシンアカウントの作成

- メンバーサーバーの Active Directory ドメインへの関連付け

マシンアカウントを作成して、Windows 2000/2003/2008 Active Directory に参加するには、Active Directory ドメインに参加させたいメンバーサーバーに対して、Kerberos を最初に初期化しておく必要があります。管理用の Kerberos チケットを作成するには、メンバーサーバー上で root として、以下のコマンドを入力します:

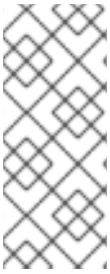
```
kinit administrator@EXAMPLE.COM
```

kinit コマンドは、Active Directory 管理者アカウントと Kerberos レルムを参照する Kerberos 初期化スクリプトです。Active Directory には Kerberos チケットが必要なため、**kinit** はクライアント/サーバー認証用に Kerberos チケット保証チケットを取得して、キャッシュします。
Kerberos、**/etc/krb5.conf** ファイル、**kinit** コマンドの詳細については、Red Hat Enterprise Linux 6 『シングルサインオンとスマートカードの管理』 ガイドの 『Kerberos の使用』 の項を参照して下さい。

Active Directory サーバー (windows1.example.com) に参加するには、メンバーサーバー上で root として以下のコマンドを入力します:

```
net ads join -S windows1.example.com -U administrator%password
```

マシン **windows1** が、対応する Kerberos レルム内で (**kinit** のコマンドが成功して) 自動的に検出された後には、**net** コマンドで、必要な管理者アカウントとパスワードを使用して Active Directory サーバーに接続します。これによって、Active Directory 上に適切なマシンアカウントが作成され、Samba ドメインメンバーサーバーに対して、ドメインに参加するパーミッションが付与されます。



注記

security = user ではなく、**security = ads** が使用されるため、**smbpasswd** のようなローカルのパスワードバックエンドは必要ありません。**security = ads** をサポートしていない旧式のクライアントは、**security = domain** が設定されているかのように認証されます。この変更により、機能に影響を及ぼすことはなく、以前ドメインには入っていなかったローカルユーザーが許可されます。

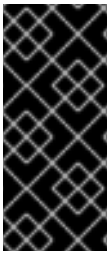
18.1.6.2.2. Windows NT4 ベースのドメインメンバーサーバー

以下の **/etc/samba/smb.conf** ファイルでは、Windows NT4 ベースのドメインメンバーサーバーの実装に必要な設定の例を示しています。NT4 ベースのドメインメンバーサーバーとなるのは、Active Directory への接続と類似しています。主な相異点は、NT4 ベースのドメインでは、認証方法として Kerberos を使用しないため、**/etc/samba/smb.conf** ファイルがより単純である点です。この例では、Samba メンバーサーバーが NT4 ベースのドメインサーバーへのパススルーとして機能します。

```
[global]
workgroup = DOCS
netbios name = DOCS_SRV
security = domain
[homes]
comment = Home Directories
valid users = %S
read only = No
browseable = No
[public]
comment = Data
path = /export
```

```
force user = docsbot
force group = users
guest ok = Yes
```

Samba をドメインメンバーサーバーとして使用すると、様々な状況で役立ちます。ファイルやプリンター共有以外の用途で Samba サーバーを使う場合があります。ドメイン環境において、Linux 専用アプリケーションが必要とされる場合、Samba をドメインメンバーサーバーとすることによってメリットがもたらされる場合があります。Windows ベースでなくとも、ドメイン内のマシンはすべて追跡記録されるため、管理者にとって好都合です。Windows ベースのサーバーハードウェアが推奨されない場合には、**/etc/samba/smb.conf** ファイルを修正してそのサーバーを Samba ベースの PDC に容易に変換することができます。又、Windows NT ベースのサーバーが Windows 2000/2003/2008 にアップグレードされた場合には、**/etc/samba/smb.conf** ファイルを修正して、必要に応じて、Active Directry へのインフラストラクチャーの変更を容易に組み込むことができます。



重要

/etc/samba/smb.conf ファイルの設定が終了したら、Samba を起動する **前に** root として以下のコマンドを入力して、ドメインに参加します。

```
net rpc join -U administrator%password
```

ドメインサーバーのホスト名を指定する **-S** オプションは、 **net rpc join** コマンドでは指定する必要がない点に注意して下さい。Samba では、ホスト名を明示的に記載するのではなく、**/etc/samba/smb.conf** ファイル内の **workgroup** 指示文で指定されているホスト名を使用します。

18.1.6.3. ドメインコントローラー

Windows NT のドメインコントローラーは、Linux 環境における Network Information Service (NIS) サーバーと類似した機能です。ドメインコントローラーと NIS サーバーはいずれもユーザー/グループ情報データベースと関連サービスのホスティングを行います。ドメインコントローラーは、ドメインリソースにアクセスするユーザーの認証などのセキュリティ目的で主に使用されます。ユーザー/グループデータベースの整合性を維持するサービスは、*Security Account Manager* (SAM) と呼ばれています。SAM データベースの保管方法は、Windows と Linux の Samba ベースシステムでは異なるため、SAM の複製を実行したり、PDC/BDC 環境でプラットフォームを混在させることはできません。

Samba 環境で使用できる PDC は一つのみ、BDC はゼロ以上です。

重要

Samba/Windows ドメインコントローラーが混在する環境では、Samba を使用することはできません (Samba は、Windows PDC の BDC もしくは Windows BDC の PDC にはなりません)。その代わりに、Samba PDC と BDC は**共存させることができます**。

18.1.6.3.1. tdbsam を使用する Primary Domain Controller (PDC)

最もシンプルかつ一般的な Samba PDC 実装は、新しいデフォルトの **tdbsam** パスワードデータベースバックエンドを使用する方法です。エージング **smbpasswd** バックエンドである **tdbsam** の交換により、数々の点が改善されます。詳細については、「Samba アカウント情報データベース」 を参照して下さい。**passdb backend** 指示文は、PDC にどのバックエンドを使用するかを制御します。

以下のファイル **/etc/samba/smb.conf** には、**tdbsam** パスワードデータベースバックエンドを実装するために必要なサンプル設定を示しています。

```
[global]
workgroup = DOCS
netbios name = DOCS_SRV
passdb backend = tdbsam
security = user
add user script = /usr/sbin/useradd -m "%u"
delete user script = /usr/sbin/userdel -r "%u"
add group script = /usr/sbin/groupadd "%g"
delete group script = /usr/sbin/groupdel "%g"
add user to group script = /usr/sbin/usermod -G "%g" "%u"
add machine script = /usr/sbin/useradd -s /bin/false -d /dev/null  -g
machines "%u"
# The following specifies the default logon script
# Per user logon scripts can be specified in the user
# account using pdbedit logon script = logon.bat
# This sets the default profile path.
# Set per user paths with pdbedit
logon drive = H:
domain logons = Yes
os level = 35
preferred master = Yes
domain master = Yes
[homes]
 comment = Home Directories
 valid users = %S
 read only = No
[netlogon]
 comment = Network Logon Service
 path = /var/lib/samba/netlogon/scripts
 browseable = No
 read only = No
# For profiles to work, create a user directory under the
# path shown.
mkdir -p /var/lib/samba/profiles/john
[Profiles]
 comment = Roaming Profile Share
 path = /var/lib/samba/profiles
 read only = No
 browseable = No
 guest ok = Yes
 profile acls = Yes
# Other resource shares ... ...
```

tdbsam を使用する機能的な PDC システムを提供するには、次のステップを実行します:

1. 上記の例のように **smb.conf** ファイルの設定を使用します。
2. Samba パスワードデータベースに root ユーザーを追加します。

```
~]# smbpasswd -a root
ここで、パスワードを入力します。
```

3. **smb** サービスを起動します。

4. すべてのプロファイル、ユーザー、netlogon ディレクトリが作成されていることを確認します。

5. ユーザーがメンバーであるグループを追加します。

```
~]# groupadd -f users
~]# groupadd -f nobody
~]# groupadd -f ntadmins
```

6. UNIX グループとそれぞれの Windows グループを関連付けます。

```
~]# net groupmap add ntgroup="Domain Users" unixgroup=users
~]# net groupmap add ntgroup="Domain Guests" unixgroup=nobody
~]# net groupmap add ntgroup="Domain Admins" unixgroup=ntadmins
```

7. ユーザーまたはグループにアクセス権を付与します。例えば、クライアントマシンを Samba ドメインコントローラーのドメインに追加するアクセス権を Domain Admins グループへのメンバーに付与するには、次のコマンドを実行します:

```
~]# net rpc rights grant 'DOCS\Domain Admins'
SetMachineAccountPrivilege -S PDC -U root
```

Windows システムには、Domain Users のようなドメイングループにマッピングされるプライマリグループがある方が望ましい点に注意して下さい。

Windows グループとユーザーは、同じ名前空間を使用するため、UNIX のように同じ名前を持つグループとユーザーが存在することはできません。

注記

複数のドメインコントローラーが必要な場合やユーザーが 250 人以上の場合には、**tdbsam** 認証バックエンドは **使用しないで下さい**。そのような場合には、LDAP が推奨されます。

18.1.6.3.2. Active Directory を使用する Primary Domain Controller (PDC)

Samba を Active Directory のメンバーにすることは可能ですが、Samba を Active Directory のドメインコントローラーとして稼働させることはできません。

18.1.7. Samba のセキュリティモード

Samba には **share-level** と **user-level** の 2 つのセキュリティモードがあり、これらはまとめて **セキュリティレベル** と総称されています。共有レベルのセキュリティの実装方法は 1 つのみとなっていますが、ユーザーレベルのセキュリティの実装には 4 つの異なる方法の中の一つを使用できます。セキュリティレベルの異なる実装方法は、**セキュリティモード** と呼ばれます。

18.1.7.1. ユーザーレベルのセキュリティ

ユーザーレベルのセキュリティは、Samba のデフォルト設定です。**security = user** 指示文が **/etc/samba/smb.conf** ファイル内に表示されていない場合でも、Samba はこの設定を使用します。サーバーがクライアントのユーザー名/パスワードを受け入れると、クライアントは各インスタンスでパ

スワードを入力する必要なく、複数の共有をマウントすることができます。又、Samba はセッションベースのユーザー名/パスワード要求を受け入れることもできます。クライアントは、各ログインに対して一意の UID を使用して、複数の認証コンテキストを維持します。

/etc/samba/smb.conf ファイル内でユーザーレベルのセキュリティを設定する **security = share** 指示文は以下の通りです。

```
[GLOBAL]
...
security = user
...
```

以下のセクションでは、その他のユーザーレベルのセキュリティ実装について説明します。

18.1.7.1.1. ドメインセキュリティモード (ユーザーレベルのセキュリティ)

ドメインセキュリティモードでは、Samba サーバーにはマシンアカウント (ドメインセキュリティトラストアカウント) があり、これによって、すべての認証要求がドメインコントローラーにパススルーされます。Samba サーバーをドメインメンバーサーバーにするには、 **/etc/samba/smb.conf** ファイル内で以下のような指示文を使用します。

```
[GLOBAL]
...
security = domain
workgroup = MARKETING
...
```

18.1.7.1.2. Active Directory セキュリティモード (ユーザーレベルのセキュリティ)

Active Directory 環境がある場合には、ネイティブの Active Directory メンバーとしてドメインに参加することが可能です。セキュリティポリシーによって NT 対応の認証プロトコルの使用が制限されている場合でも、Samba サーバーは Kerberos を使用して ADS に参加することができます。Active Directory メンバーモードの Samba は、Kerberos チケットを受け入れることが可能です。

/etc/samba/smb.conf ファイルに、以下のような指示文を追加すると、Samba は Active Directory のメンバーサーバーになります:

```
[GLOBAL]
...
security = ADS
realm = EXAMPLE.COM
password server = kerberos.example.com
...
```

18.1.7.1.3. サーバーセキュリティモード (ユーザーレベルのセキュリティ)

サーバーセキュリティモードは、以前 Samba がドメインメンバーサーバーとして機能することができなかった時に使用されていました。

注記

このモードにはセキュリティ上の欠点が多数あるため、 **使用しない** ことを強く推奨します。

/etc/samba/smb.conf ファイル内に以下の指示文を追加すると、Samba をサーバーセキュリティモードで稼働させることができます。

```
[GLOBAL]
...
encrypt passwords = Yes
security = server
password server = "NetBIOS_of_Domain_Controller"
...
```

18.1.7.2. 共有レベルのセキュリティ

共有レベルのセキュリティでは、サーバーはクライアントから、明示的なユーザー名なしにパスワードのみを受け入れます。サーバーは、ユーザー名とは別に各共有のパスワードを求めます。Microsoft Windows のクライアントで共有レベルのセキュリティサーバーとの互換性の問題があることが最近報告されています。Samba の開発者は、共有レベルのセキュリティを使用しないことを強く推奨しています。

/etc/samba/smb.conf ファイル内で共有レベルのセキュリティを設定する **security = share** 指示文は以下の通りです。

```
[GLOBAL]
...
security = share
...
```

18.1.8. Samba アカウント情報データベース

Samba の最新リリースには、以前提供されていなかった新たなパスワードデータベースバックエンドを含む数多くの新機能が搭載されています。Samba のバージョン 3.0.0 は、旧バージョンで使用されていたすべてのデータベースを完全にサポートしています。但し、バックエンドの多くは、サポートされていても実稼働での使用には適切でない場合があります。

Samba で使用できる異なるバックエンドの一覧は以下の通りです。ここには記載されていないバックエンドでも、使用可能なものもあります。

プレーンテキスト

プレーンテキストバックエンドは、**/etc/passwd** タイプのバックエンドにすぎません。プレーンテキストバックエンドでは、クライアントと Samba サーバー間ですべてのユーザー名とパスワードが暗号化されずに送信されます。この方法は、非常に不安定なため、使用は決して推奨されません。プレーンテキストのパスワードを使用して Samba サーバーに接続する異なる Windows クライアントは、この認証方法をサポートできない可能性があります。

smbpasswd

以前の Samba パッケージで使用されていた一般的なバックエンドである **smbpasswd** バックエンドは、MS Windows LanMan や NT アカウントを含むプレーンの ASCII テキストレイアウトと暗号化されたパスワード情報を使用します。**smbpasswd** バックエンドは、Windows NT/2000/2003 SAM 拡張コントロールのストレージが欠如しています。**smbpasswd** バックエンドは、拡張性や NT ベースグループ用の RID などの Windows の情報の保持能力が乏しいため推奨されません。**tdbsam** バックエンドは、小規模なデータベース (ユーザー 250 人) における使用での問題を解決しますが、エンタープライズ級のソリューションではありません。

ldapsam_compat

ldapsam_compat バックエンドは、アップグレードした Samba のバージョンと併用する場合の継続的な OpenLDAP サポートが可能です。このオプションは、通常 Samba 3.0 への移行時に使用されます。

tdbsam

新しいデフォルトの **tdbsam** パスワードバックエンドは、ローカルサーバーやビルトインデータベース複製を必要としないサーバー、スケーラビリティや LDAP の複雑性を必要としないサーバーに向けて理想的なデータベースバックエンドを提供します。**tdbsam** バックエンドには、**smbpasswd** データベース情報や以前は除外されていた SAM の情報がすべて含まれています。拡張された SAM データが含まれたことで、Samba は、Windows NT/2000/2003/2008 ベースのシステムで使用されているのと同じアカウント及びシステムへのアクセス制御を実装することが可能です。

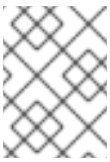
tdbsam バックエンドは、最大で 250 人のユーザー規模に推奨されます。これより大きな組織の場合は、拡張性やネットワークインフラストラクチャーの問題が懸念されるため、Active Directory または LDAP の統合が必要です。

ldapsam

ldapsam バックエンドは、Samba に最適な分散アカウントインストール方法を提供します。LDAP が最適な理由は、そのデータベースを **Red Hat Directory Server** や **OpenLDAP Server** などの任意の数のサーバーに複製できる点です。LDAP データベースは軽量でスケーラブルなため、大企業に好まれます。ディレクトリサーバーのインストールと設定は、本章の対象外となります。**Red Hat Directory Server** の詳細については、『Red Hat Directory Server 9.0 Deployment Guide』 を参照して下さい。LDAP については、 「OpenLDAP」 をご覧下さい。

旧バージョンの Samba から 3.0 にアップグレードする場合には、OpenLDAP スキーマファイル (**/usr/share/doc/samba-*<version>*/LDAP/samba.schema**) 及び Red Hat Directory Server スキーマファイル (**/usr/share/doc/samba-*<version>*/LDAP/samba-schema-FDS.ldif**) が変更されている点に注意して下さい。これらのファイルには、**ldapsam** バックエンドが適正に機能するために必要な *属性構文の定義* と *オブジェクトクラス定義* が含まれています。

このため、**ldapsam** バックエンドを Samba サーバーに使用する場合は、**slapd** にこのスキーマファイルが含まれるように設定する必要があります。その方法については、 「スキーマの拡張」 を参照して下さい。



注記

ldapsam バックエンドを使用する場合には、**openldap-server** パッケージをインストールする必要があります。

18.1.9. Samba のネットワークブラウジング

ネットワークブラウジング により、Windows と Samba のサーバーが Windows の **ネットワークコンピュータ (Network Neighborhood)** に表示されるようになります。**ネットワークコンピュータ** には、アイコンがサーバーとして表示され、それを開くとそのサーバーの利用可能な共有やプリンターが表示されます。

ネットワークブラウジングの機能には **TCP**/**IP** 上の NetBIOS が必要です。NetBIOS ベースのネットワーキングでは、ブロードキャスト (**UDP**) メッセージングを使用してブラウズリスト管理を行います。NetBIOS と WINS を **TCP**/**IP** ホスト名解決の第一の方法として使用しない場合は、静的ファイル (**/etc/hosts**) や **DNS** などの他の方法を使用する必要があります。

ドメインマスターブラウザーは、すべてのサブネット上のローカルマスターブラウザーからブラウズリストを照合して、ワークグループとサブネット間のブラウジングを可能にします。独自のサブネットにはドメインマスタブラウザーをローカルマスターブラウザーにするのが望ましいでしょう。

18.1.9.1. ドメインブラウジング

デフォルトでは、ドメイン用の Windows サーバーの PDC は、そのドメイン用のドメインマスタブラウザーでもあります。この種の状況においては、Samba サーバーをドメインマスターサーバーとして **設定してはなりません**。

Windows サーバーの PDC が含まれていないサブネットの場合、Samba サーバーはローカルマスターブラウザーとして実装することができます。ドメインコントローラー環境での **/etc/samba/smb.conf** ファイルのローカルマスターブラウザー用 (もしくはブラウジング一切なし) 設定は、ワークグループ設定と同様になります (「Samba サーバーの設定」 を参照してください)。

18.1.9.2. WINS (Windows Internet Name Server)

Samba サーバーまたは Windows NT サーバーは、WINS サーバーとして機能させることが可能です。NetBIOS が有効な状態で WINS サーバーを使用する場合には、UDP ユニキャストをルーティングして、ネットワーク全体にわたる名前解決を可能にすることができます。WINS サーバーを使用しない場合は、UDP ブロードキャストがローカルのサブネットに限定されるため、他のサブネット、ワークグループ、ドメインにルーティングすることはできません。Samba は現在 WINS の複製に対応していないため、WINS の複製が必要な場合には、Samba をプライマリ WINS サーバーとして使用しないで下さい。

NT/2000/2003/2008 サーバーと Samba が混在する環境においては、Microsoft WINS 機能を使用することを推奨します。Samba のみの環境では、WINS には **単一の** Samba サーバーを使用することを推奨します。

以下は、Samba サーバーが WINS サーバーとして機能する場合の **/etc/samba/smb.conf** ファイルの一例です:

```
[global]
wins support = Yes
```

注記

NetBIOS 名を解決するには、全サーバー (Samba を含む) を WINS サーバーに接続する必要があります。WINS がない場合は、ローカルのサブネット上でしかブラウジングは行われません。又、ドメイン全体の一覧を取得したとしても、WINS を使用しないクライアントに対しては、ホストを解決できません。

18.1.10. CUPS 印刷サポートを実装した Samba

Samba により、クライアントマシンは Samba サーバーに接続されたプリンターを共有することができます。又、Samba により、クライアントマシンは Linux に内蔵されているドキュメントを Windows のプリンター共有に送信することもできます。Red Hat Enterprise Linux で機能する印刷システムは他にもありますが、Samba との緊密な統合性から CUPS (Common UNIX Print System) が印刷システムとして推奨されます。

18.1.10.1. smb.conf の簡易設定

以下の例は、CUPS サポート用の基本的な **/etc/samba/smb.conf** 設定です。

-

```
[global]
load printers = Yes
printing = cups
printcap name = cups
[printers]
comment = All Printers
path = /var/spool/samba
browseable = No
public = Yes
guest ok = Yes
writable = No
printable = Yes
printer admin = @ntadmins
[print$]
comment = Printer Drivers Share
path = /var/lib/samba/drivers
write list = ed, john
printer admin = ed, john
```

その他の印刷設定も可能です。機密書類の印刷における安全性とプライバシーを強化するために、ユーザーはパブリックパス以外の場所に配置された独自の印刷スプーラーを使用することができます。ジョブが失敗した場合には、他のユーザーはファイルにアクセスできません。

print$** 指示文には、ローカルで利用できない場合にクライアントがアクセスできるプリンタードライバーが格納されています。**print$ 指示文はオプションであり、組織によっては必要ない場合があります。

browseable を **Yes** に設定すると、プリンターは Windows ネットワークコンピュータに表示されます。但し、Samba サーバーがドメイン/ワークグループ内で正しく設定されていることが条件です。

18.1.11. Samba ディストリビューションプログラム

findsmb

findsmb *<subnet_broadcast_address>*

findsmb プログラムは Perl スクリプトで、特定のサブネット上の **SMB** 対応システムに関する情報をレポートします。サブネットが指定されていない場合には、ローカルサブネットが使用されます。表示される項目には、**IP** アドレス、NetBIOS 名、ワークグループ名またはドメイン名、オペレーティングシステム、バージョンなどが含まれます。

以下にあげる例は、システム上の任意の有効なユーザーとして **findsmb** を実行した際の出力を示しています。

```
~]$ findsmb
IP ADDR       NETBIOS NAME  WORKGROUP/OS/VERSION
------------------------------------------------------------------
10.1.59.25    VERVE          [MYGROUP] [Unix] [Samba 3.0.0-15]
10.1.59.26    STATION22      [MYGROUP] [Unix] [Samba 3.0.2-7.FC1]
10.1.56.45    TREK          +[WORKGROUP] [Windows 5.0] [Windows 2000 LAN
Manager]
10.1.57.94    PIXEL          [MYGROUP] [Unix] [Samba 3.0.0-15]
10.1.57.137   MOBILE001      [WORKGROUP] [Windows 5.0] [Windows 2000 LAN
Manager]
10.1.57.141   JAWS          +[KWIKIMART] [Unix] [Samba 2.2.7a-security-
rollup-fix]
```

```
10.1.56.159   FRED          +[MYGROUP] [Unix] [Samba 3.0.0-14.3E]
10.1.59.192   LEGION        *[MYGROUP] [Unix] [Samba 2.2.7-security-rollup-
fix]
10.1.56.205   NANCYN        +[MYGROUP] [Unix] [Samba 2.2.7a-security-
rollup-fix]
```

net

net *<protocol> <function> <misc_options> <target_options>*

net ユーティリティは、Windows や MS-DOS で使用する **net** ユーティリティと似ています。第一引数は、コマンドを実行する時にプロトコルを指定するのに使用されます。***<protocol>*** オプションには、**ads**、**rap** または **rpc** を用いて、サーバー接続のタイプを指定することができます。Active Directory では **ads**、Win9x/NT3 では **rap**、Windows NT4/2000/2003/2008 では **rpc** を使用します。プロトコルが省略されている場合には、**net** は自動的に特定を試みます。

以下の例では、**wakko** という名前のホストが利用可能な共有の一覧を示しています。

```
~]$ net -l share -S wakko
Password:
Enumerating shared resources (exports) on remote server:
Share name   Type     Description
----------   ----     -----------
data         Disk     Wakko data share
tmp          Disk     Wakko tmp share
IPC$         IPC      IPC Service (Samba Server)
ADMIN$       IPC      IPC Service (Samba Server)
```

以下の例は、**wakko** という名前のホストの Samba ユーザーの一覧を示しています。

```
~]$ net -l user -S wakko
root password:
User name             Comment
-----------------------------
andriusb              Documentation
joe                   Marketing
lisa                  Sales
```

nmblookup

nmblookup *<options> <netbios_name>*

nmblookup プログラムは、NetBIOS 名を **IP** アドレスに解決します。このプログラムは、ターゲットマシンが応答するまで、ローカルサブネット上のクエリをブロードキャストします。

以下の例では、NetBIOS 名 **trek** の IP アドレスを表示しています:

```
nmblookup trek
querying trek on 10.1.59.255
10.1.56.45 trek<00>
```

pdbedit

pdbedit *<options>*

pdbedit プログラムは、SAM データベース内にあるアカウントを管理します。**smbpasswd**、LDAP、**tdb** データベースライブラリを含むすべてのバックエンドがサポートされています。

以下はユーザーの追加、削除、一覧表示の例です

```
~]$ pdbedit -a kristin
new password:
retype new password:
Unix username:        kristin
NT username:
Account Flags:        [U          ]
User SID:             S-1-5-21-1210235352-3804200048-1474496110-2012
Primary Group SID:    S-1-5-21-1210235352-3804200048-1474496110-2077
Full Name: Home Directory:       \\wakko\kristin
HomeDir Drive:
Logon Script:
Profile Path:         \\wakko\kristin\profile
Domain:               WAKKO
Account desc:
Workstations: Munged
dial:
Logon time:           0
Logoff time:          Mon, 18 Jan 2038 22:14:07 GMT
Kickoff time:         Mon, 18 Jan 2038 22:14:07 GMT
Password last set:    Thu, 29 Jan 2004 08:29:28
GMT Password can change:  Thu, 29 Jan 2004 08:29:28 GMT
Password must change: Mon, 18 Jan 2038 22:14:07 GMT
~]$ pdbedit -v -L kristin
Unix username:        kristin
NT username:
Account Flags:        [U          ]
User SID:             S-1-5-21-1210235352-3804200048-1474496110-2012
Primary Group SID:    S-1-5-21-1210235352-3804200048-1474496110-2077
Full Name:
Home Directory:       \\wakko\kristin
HomeDir Drive:
Logon Script:
Profile Path:         \\wakko\kristin\profile
Domain:               WAKKO
Account desc:
Workstations: Munged
dial:
Logon time:           0
Logoff time:          Mon, 18 Jan 2038 22:14:07 GMT
Kickoff time:         Mon, 18 Jan 2038 22:14:07 GMT
Password last set:    Thu, 29 Jan 2004 08:29:28 GMT
Password can change:  Thu, 29 Jan 2004 08:29:28 GMT
Password must change: Mon, 18 Jan 2038 22:14:07 GMT
~]$ pdbedit -L
andriusb:505:
joe:503:
lisa:504:
kristin:506:
~]$ pdbedit -x joe
~]$ pdbedit -L
andriusb:505: lisa:504: kristin:506:
```

rpcclient

rpcclient *<server> <options>*

rpcclient プログラムは、システム管理用の Windows 管理グラフィカルユーザーインターフェース (GUI) へのアクセスを提供する、Microsoft RPC を使用した管理コマンドを発行します。ほとんどの場合、このプログラムを使用するのは、Microsoft RPC の複雑性を完全に理解している上級ユーザーです。

smbcacls

smbcacls *<//server/share> <filename> <options>*

smbcacls プログラムは、Samba サーバーまたは Windows サーバーより共有されているファイルやディレクトリ上の Windows ACL を変更します。

smbclient

smbclient *<//server/share> <password> <options>*

smbclient プログラムは、**ftp** と同様の機能を提供する多用途の UNIX クライアントです。

smbcontrol

smbcontrol -i *<options>*

smbcontrol *<options> <destination> <messagetype> <parameters>*

smbcontrol プログラムは、実行中の **smbd**、**nmbd** または **winbindd** デーモンに制御メッセージを送信します。**smbcontrol -i** を実行すると、空白行または ***'q'*** と入力するまでコマンドはインタラクティブに実行されます。

smbpasswd

smbpasswd *<options> <username> <password>*

smbpasswd プログラムは暗号化されたパスワードを管理します。このプログラムを実行して、スーパーユーザーが任意のユーザーのパスワードを変更したり、一般ユーザーが自分の Samba パスワードを変更することができます。

smbspool

smbspool *<job> <user> <title> <copies> <options> <filename>*

smbspool プログラムは、Samba への CUPS 対応印刷インターフェースです。**smbspool** は CUPS プリンターでの使用に設計されていますが、非 CUPS プリンターでも機能します。

smbstatus

smbstatus *<options>*

smbstatus プログラムは、Samba サーバーへの現在の接続状態を表示します。

smbtar

smbtar *<options>*

smbtar プログラムは、Windows ベースの共有ファイル及びディレクトリのローカルテープアーカイブへのバックアップと復元を実行します。**tar** コマンドと似ていますが、これらの 2 つのコマンドには互換性はありません。

testparm

testparm *<options> <filename> <hostname IP_address>*

testparm プログラムは、**/etc/samba/smb.conf** ファイルの構文を確認します。ご使用の **/etc/samba/smb.conf** ファイルがデフォルトの場所 (**/etc/samba/smb.conf**) にある場合は、ロケーションを指定する必要はありません。**testparm** プログラムにホスト名と IP アドレスを指定すると、**hosts.allow** と **host.deny** のファイルが正しく設定されているか検証します。又、**testparm** プログラムは、テスト後にご使用の **/etc/samba/smb.conf** ファイルの概要やサーバーのロール (スタンドアロン、ドメインなど) も表示します。コメントが除外され、熟練の管理者が読む情報が簡潔に表示されるため、デバッグを行う時に便利です。

例:

```
~]$ testparm
Load smb config files from /etc/samba/smb.conf
Processing section "[homes]"
Processing section "[printers]"
Processing section "[tmp]"
Processing section "[html]"
Loaded services file OK.
Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions
<enter>
# Global parameters
[global]
 workgroup = MYGROUP
 server string = Samba Server
 security = SHARE
 log file = /var/log/samba/%m.log
 max log size = 50
 socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
 dns proxy = No
[homes]
 comment = Home Directories
 read only = No
 browseable = No
[printers]
 comment = All Printers
 path = /var/spool/samba
 printable = Yes
 browseable = No
[tmp]
 comment = Wakko tmp
 path = /tmp
 guest only = Yes
[html]
 comment = Wakko www
 path = /var/www/html
 force user = andriusb
```

```
force group = users
read only = No
guest only = Yes
```

wbinfo

wbinfo *<options>*

wbinfo プログラムは、**winbindd** デーモンからの情報を表示します。**wbinfo** が機能するには、**winbindd** デーモンが実行されている必要があります。

18.1.12. その他のリソース

以下のセクションには、Samba に関する詳しい情報を調べる方法を記載しています。

18.1.12.1. インストールされているドキュメント

-

  注記

  Samba のドキュメントを使用するには、まず最初に root として以下のコマンドを実行し、samba-doc パッケージをご使用のシステムに確実にインストールします:

  ```
  ~]# yum install samba-doc
  ```

  Yum を使用したパッケージのインストールについての詳細は、「パッケージのインストール」 を参照して下さい。

  /usr/share/doc/samba-*<version-number>*/ — Samba ディストリビューションに同梱されているすべての追加ファイル。これには、ヘルパースクリプト、サンプル設定ファイル、ドキュメントなどが含まれています。このディレクトリには、『The Official Samba-3 HOWTO-Collection』 と 『Samba-3 by Example』 のオンラインバージョンも含まれています。それぞれの詳細は、以下の通りです。

  特定の **Samba** 機能の詳細については、次の man ページを参照して下さい:

  - **smb.conf**
  - **samba**
  - **smbd**
  - **nmbd**
  - **winbind**

18.1.12.2. 関連資料

- 『The Official Samba-3 HOWTO-Collection』 (John H. Terpstra、Jelmer R. Vernooij 共著、Prentice Hall 社出版) — Samba 開発チームが刊行した 公式な Samba-3 ドキュメント。ステップバイステップのガイドというよりは、むしろ参照ガイドです。
- 『Samba-3 by Example』 (John H. Terpstra 著、Prentice Hall 社出版) — Samba 開発チームが刊行した、もう一つの公式なリリース。OpenLDAP、DNS、DHCP、印刷設定ファイルについ

ての詳しい例を解説しています。関連情報がステップバイステップで記載されているので、実環境での実装に役立ちます。

- 『Using Samba, 2nd Edition』 (Jay Ts、Robert Eckstein、David Collier-Brown 共著、O'Reilly 社出版) — 包括的な参考資料を満載した、初級から上級ユーザー向けのリソースです。

18.1.12.3. 役立つ Web サイト

- http://www.samba.org/ — Samba 開発チームが作成した Samba ディストリビューション及びすべての正式なドキュメントのホームページ。数々のリソースを HTML と PDF 形式で入手できる他、有償配布のみの書籍も取り扱っています。これらのリンクの多くは、Red Hat Enterprise Linux 固有ではありませんが一部のコンセプトは該当します。

- http://samba.org/samba/archives.html — Samba コミュニティのアクティブな電子メール一覧。リストのアクティビティレベルが高いため、ダイジェストモードを有効にすることを推奨します。

- Samba ニュースグループ — NNTP プロトコルを使用する gmane.org などの Samba のスレッドのニュースグループも利用可能。メーリングリストの電子メールを受信する代わりに利用することができます。

18.2. FTP

File Transfer Protocol (**FTP**) は、今日インターネット上で見られる、最も古く、一般的に使用されているプロトコルです。目的は、ユーザーがリモートホストに直接ログインしたり、リモートシステムの使用法についての知識を持つ必要なく、ネットワーク上のコンピュータホスト間で確実にファイルを転送することです。これによりユーザーは、標準の簡単なコマンドセットを使用してリモートシステム上のファイルにアクセスすることができるようになります。

このセクションでは、**FTP** プロトコルの基礎概要、及び Red Hat Enterprise Linux に同梱されている主要な **FTP** サーバー用の設定オプション ***vsftpd*** を解説しています。

18.2.1. File Transfer Protocol (ファイル転送プロトコル)

しかし、**FTP** がインターネット上で普及しているため、公共とのファイルの共有を要求されることが多くあります。そのためシステム管理者は、**FTP** プロトコル特有の性格について認識しておく必要があります。

インターネット上で使用されているほとんどのプロトコルとの相違点は、**FTP** が正しく機能するために複数のネットワークポートを必要とすることです。**FTP** クライアントアプリケーションが **FTP** サーバーへの接続を開始する際に、*コマンドポート* として知られるポート **21** をサーバー上で開きます。このポートは、すべてのコマンドをサーバーに発行するために使用されます。サーバーから要求されたデータはいずれも *データポート* を介してクライアントに返されます。データ接続が開始されるデータ接続用ポートの番号は、クライアントが *アクティブ* か *パッシブ* のモードでデータを要求するかによって変化します。

これらのモードについての定義は以下の通りです。

アクティブモード

アクティブモードは、**FTP** プロトコルでクライアントへのデータ転送に使用される元来の方法です。**FTP** クライアントがアクティブモードのデータ転送を開始すると、サーバーは、サーバー上のポート **20** から、クライアントの指定する **IP** アドレスとランダムな非特権ポート (**1024** 以上) への接続を開きます。このような方法では、クライアントマシンがポート **1024** 以上での接続を受け入れるように許可されていなければならないことになります。インターネットのような、セキュリ

ティ保護されていないネットワークが増加するにともない、ファイアウォールを使用したクライアントマシンの保護が普及しています。このようなクライアント側のファイアウォールはアクティブモードの **FTP** サーバーから着信する接続を拒否する場合が多いため、パッシブモードが考案されました。

パッシブモード

パッシブモードは、アクティブモードと同様に、**FTP** クライアントアプリケーションによって開始されます。サーバーからのデータを要求する際に、**FTP** クライアントはパッシブモードでデータにアクセスしたいことを知らせると、サーバーはサーバー上の **IP** アドレスとランダムな非特権ポート (**1024** 以上) を提供します。クライアントは、サーバー上のそのポートに接続して要求した情報をダウンロードします。

パッシブモードは、クライアント側のファイアウォールによるデータ接続障害の問題を解決しますが、サーバー側のファイアウォールの管理を複雑化させてしまう場合があります。**FTP** サーバー上の非特権ポートの範囲を制限することにより、サーバー上の開いたポートの数を減らすことができます。また、この方法により、サーバーを対象としたファイアウォールのルールを設定する手順が簡略化されます。パッシブポートの制限に関する更なる情報は、「ネットワークのオプション」 をご参照ください。

18.2.2. vsftpd サーバー

Very Secure FTP Daemon (**vsftpd**) は、高速で安定性があり、かつ何よりもセキュアであるように土台から設計されています。**vsftpd** は、多数の接続を効率的かつセキュアに処理できる能力があることから、Red Hat Enterprise Linux に標準装備されている、唯一の独立型 **FTP** サーバーです。

vsftpd で使用されるセキュリティーモデルには、以下にあげる 3 つの主要な側面があります:

- **権限プロセスと非権限プロセス間に断然たる相違** — 別々のプロセスが異なるタスクを処理して、それらの各プロセスはタスクに必要な最低限の権限で実行します。
- **高い権限を必要とするタスクを、必要最小限の権限を持つプロセスで処理** — **libcap** ライブラリ内にある互換性を利用して、通常は完全な root 権限を必要とするタスクを、権限が低いプロセスでより安全に実行することができます。
- **プロセスの大半が chroot jail で実行** — できる限り、プロセスは共有ディレクトリに対してルートディレクトリを変更します。このディレクトリは、**chroot** jail と見なされます。例えば、ディレクトリ **/var/ftp/** がプライマリ共有ディレクトリの場合、**vsftpd** は、**/var/ftp/** を **/** として知られる、新規 root ディレクトリとして再割り当てします。これにより、新たな root ディレクトリ下に格納されていないディレクトリに対する、潜在的な悪質ハッカー行為を行うことができないようにします。

これらのセキュリティプラクティスを使用すると、**vsftpd** による要求対応方法に以下のような影響を及ぼします:

- **親プロセスは、必要最小限の権限で稼働します** — 親プロセスは、リスクレベルを最低限に抑えるために必要とされる権限のレベルを動的に算出します。子プロセスは、**FTP** クライアントとの直接的なインタラクションを処理し、可能な限り無しに近い権限で稼働します。
- **高い権限を必要とするオペレーションはすべて、小さな親プロセスによって処理します** — Apache **HTTP** Server とほぼ同様に、**vsftpd** は権限のない子プロセスを起動させ、着信接続を処理します。これにより、権限のある親プロセスを最小限に抑えられ、比較的少ないタスクを処理することになります。

- **親プロセスは、権限のない子プロセスからのすべての要求を信頼しません** — 子プロセスとの通信はソケット上で受信され、子プロセスからの情報の妥当性は動作を実施する前にチェックされます。

- **FTP クライアントとのインタラクションの大半は、chroot jail** 内の権限のない子プロセスによって処理されます — これらの子プロセスには権限がなく、共有ディレクトリへのアクセスしかないため、プロセスがクラッシュした際に攻撃者がアクセスできるのは共有ファイルのみです。

18.2.3. vsftpd によるファイルのインストール

vsftpd RPM により、デーモン (**/usr/sbin/vsftpd**)、その設定および関連ファイルと **FTP** ディレクトリがシステム上にインストールされます。**vsftpd** の設定に関連したファイルとディレクトリの一覧は以下の通りです。

- **/etc/rc.d/init.d/vsftpd** — **vsftpd** を起動/停止、あるいはリロードさせるのに **/sbin/service** コマンドによって使用される **初期化スクリプト** (*initscript*)。このスクリプトの使用については、「vsftpd の起動と停止」 をご参照ください。

- **/etc/pam.d/vsftpd** — **vsftpd** 用の Pluggable Authentication Modules (PAM) の設定ファイル。このファイルは、ユーザーが **FTP** サーバーにログインする際に満たす必要のある要件を指定します。PAM についての更なる情報は、Red Hat Enterprise Linux 6 『Managing Single Sign-On and Smart Cards（シングルサインオンとスマートカードの管理）』 ガイドにある 『Using Pluggable Authentication Modules (PAM の使用)』 の章をご覧下さい。

- **/etc/vsftpd/vsftpd.conf** — **vsftpd** の設定ファイル。このファイルに含まれている重要なオプションの一覧は、「vsftpd 設定オプション」 をご参照ください。

- **/etc/vsftpd.ftpusers** — **vsftpd** へのログインを許可されていないユーザーの一覧。このリストにはデフォルトで **root**、**bin**、および **daemon** ユーザー、その他が含まれます。

- **/etc/vsftpd/user_list** — このファイルは、**/etc/vsftpd/vsftpd.conf** 内で **userlist_deny** ディレクティブが **YES** (デフォルト) または **NO** に設定されているかによって、リストされているユーザーへのアクセスを否認または承認するように設定することが可能です。**/etc/vsftpd/user_list** をユーザーのアクセス権付与に使用する場合、そこにリストするユーザー名は **/etc/vsftpd/ftpusers** には **リストしてはいけません**。

- **/var/ftp/** — このディレクトリには、**vsftpd** によって提供されるファイルが格納されます。また、匿名ユーザー用の **/var/ftp/pub/** ディレクトリも格納されます。両方のディレクトリは、全ユーザーが読み取り可能ですが、書き込みは root ユーザーに限定されています。

18.2.4. vsftpd の起動と停止

vsftpd RPM により、**service** コマンドを使用してアクセス可能な **/etc/rc.d/init.d/vsftpd** スクリプトがインストールされます。

サーバーを起動するには、root として以下を入力します:

```
~]# service vsftpd start
```

サーバーを停止するには、root として以下を入力します:

```
~]# service vsftpd stop
```

restart のオプションは、**vsftpd** を一旦停止させてから起動する簡単な方法です。これは、**vsftpd** の設定ファイルを編集した後に変更を有効にするのに最も効率的な方法です。

サーバーを再起動するには、root として以下を入力します:

```
~]# service vsftpd restart
```

condrestart (*conditional restart*) のオプションは、**vsftpd** が稼働中の場合にのみ起動します。このオプション、デーモンが実行されていない場合には起動しないので、スクリプトに有用です。

条件付きでサーバーを再起動させるには、root として以下を入力します:

```
~]# service vsftpd condrestart
```

デフォルトでは、**vsftpd** サービスは、ブート時には自動的に **起動しません**。**vsftpd** サービスがブート時に起動するように設定するには、**/sbin/chkconfig**、**/usr/sbin/ntsysv**、または **サービス設定ツール** プログラムなどの initscript ユーティリティを使用します。これらのツールに関する更に詳しい情報は、10章 *サービスとデーモン* を参照してください。

18.2.4.1. vsftpd の複数コピーの起動

一台のコンピューターを複数の **FTP** ドメインに使用する場合があります。これは、 *マルチホーム化* と呼ばれるテクニックです。**vsftpd** を使用したマルチホーム化の一つは、デーモンの複数のコピーを実行して、それぞれが自身の設定ファイルを持つようにする方法です。

これを行うには、まずすべての関連する **IP** アドレスをシステム上のネットワークデバイスまたはエイリアスネットワークデバイスに割り当てます。ネットワークデバイスおよびデバイスエイリアスの設定に関する更に詳しい情報には、8章*NetworkManager* を参照してください。ネットワーク設定スクリプトに関する更なる情報には 9章 *ネットワークインターフェース* を参照してください。

次に、**FTP** ドメイン用の DNS サーバーは、正しいマシンを参照するように設定する必要があります。BIND およびその設定ファイルに関する情報には、「BIND」 をご参照ください。

/etc/vsftpd ディレクトリに他の設定ファイルが存在する場合は、**service vsftpd start** を呼び出すことにより、**/etc/rc.d/init.d/vsftpd**initscript が設定ファイルの数と同じ数のプロセスを開始するようになります。各設定ファイルは **/etc/vsftpd/** ディレクトリ内で特有の名前を持ち、root のみの読み込み/書き込み可能である必要があります。

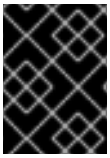
18.2.5. vsftpd 設定オプション

vsftpd は、広く利用可能な他の **FTP** サーバーのようなレベルのカスタマイズは提供しないかもしれませんが、管理者の大半のニーズを満たすのに十分なオプションを提供します。過剰に機能満載されていないことで、設定およびプログラム上のエラーが制限されます。

vsftpd の設定はすべて、その設定ファイル **/etc/vsftpd/vsftpd.conf** によって処理されます。ファイル内では、各指示文が独自の行に記述され、以下のような書式に従います:

```
<directive>=<value>
```

指示文ごとに、*<directive>* の箇所を有効な指示文に、また *<value>* の箇所を有効な値に置き換えます。



重要

一つの指示文内で *<directive>*、イコール記号、および *<value>* の間には空白が入ってはいけません。

コメント行の先頭には、ハッシュマーク (#) を付ける必要があります。これによって、デーモンに無視されます。

利用可能なすべての指示文の完全なリストには、**vsftpd.conf** の man ページをご参照ください。

重要

vsftpd をセキュリティ保護する方法の概要については、 Red Hat Enterprise Linux 6 『セキュリティガイド』 をご参照ください。

以下は、**/etc/vsftpd/vsftpd.conf** 内のより重要な指示文の一部の一覧です。 **vsftpd** の設定ファイル内で明示的に指定されていない、またはコメントアウトされている指示文はすべてデフォルト値に設定されます。

18.2.5.1. デーモンのオプション

以下は、**vsftpd** デーモンの全体的な動作を制御する指示文の一覧です。

- **listen** — 有効にすると、**vsftpd** がスタンドアロンモードで稼働します。Red Hat Enterprise Linux では、この値は **YES** に設定されています。このディレクティブは、**listen_ipv6** ディレクティブとの併用はできません。

  デフォルト値は **NO** です。Red Hat Enterprise Linux 6 では、設定ファイル内で値が **YES** に設定されています。

- **listen_ipv6** — 有効化すると、**vsftpd** がスタンドアロンモードで稼働しますが、**IPv6** ソケットのみに対して待ち受けます。このディレクティブは、**listen** ディレクティブとは併用できません。

  デフォルト値は **NO** です。

- **session_support** — 有効にすると、**vsftpd** は Pluggable Authentication Modules (PAM) を介して、各ユーザーのログインセッションの維持を試みます。詳細情報については、Red Hat Enterprise Linux 6 『Managing Single Sign-On and Smart Cards』 の 『Using Pluggable Authentication Modules (PAMの使用)』 の章と、PAM man ページを参照して下さい。セッションログインが必要でない場合は、このオプションを無効にすると **vsftpd** がより少ないプロセスとより低い権限で稼働するようになります。

  デフォルト値は **YES** です。

18.2.5.2. ログインオプションとアクセス制御

以下は、ログインの動作とアクセス制御のメカニズムをコントロールする指示文のリストです。

- **anonymous_enable** — 有効になると、匿名ユーザーによるログインが可能となります。**anonymous** と **ftp** のユーザー名が受け入れられます。

  デフォルト値は **YES** です。

匿名ユーザーに影響する指示文の一覧には、「匿名ユーザーのオプション」 をご参照ください。

- **banned_email_file** — **deny_email_enable** ディレクティブが **YES** に設定されると、このディレクティブは、サーバーへのアクセスを許可されていない匿名の電子メールパスワードの一覧を格納するファイルを指定します。

  デフォルト値は、**/etc/vsftpd/banned_emails** です。

- **banner_file** — サーバーへの接続が確立した時に表示されるテキストを格納したファイルを指定します。このオプションは、**ftpd_banner** ディレクティブに記載されているあらゆるテキストを上書きします。

  この指示文にはデフォルト値はありません。

- **cmds_allowed** — サーバーが許可する、**FTP** コマンドのコンマ区切りの一覧を指定します。その他のコマンドはすべて拒否されます。

  この指示文にはデフォルト値はありません。

- **deny_email_enable** — 有効になると、**/etc/vsftpd/banned_emails** で指定された、電子メールのパスワードを使用する匿名ユーザーはサーバーへのアクセスを拒否されます。このディレクティブで参照されているファイル名は、**banned_email_file** ディレクティブを使用して指定することができます。

  デフォルト値は **NO** です。

- **ftpd_banner** — 有効になると、サーバーへの接続が確立した時に、このディレクティブで指定されている文字列が表示されます。このオプションは、**banner_file** ディレクティブによって上書きすることができます。

  デフォルトでは、**vsftpd** は標準のバナーを表示します。

- **local_enable** — 有効になると、ローカルユーザーがシステムにログイン可能となります。

  デフォルト値は **NO** です。Red Hat Enterprise Linux 6 では、設定ファイル内で値が **YES** に設定されています。

  ローカルユーザーに影響を及ぼす指示文のリストには、「ローカルユーザーのオプション」 をご参照ください。

- **pam_service_name** — **vsftpd** の PAM サービス名を指定します。

  デフォルト値は、**ftp** です。Red Hat Enterprise Linux 6 では、このオプションは設定ファイル内で **vsftpd** にセットしてあります。

- **tcp_wrappers** — 有効になると、TCP wrappers はサーバーへのアクセスを認可するために使用されます。FTP サーバーが複数の IP アドレス上で設定されている場合は、クライアントが要求している IP アドレスを基にして異なる設定ファイルをロードするように **VSFTPD_LOAD_CONF** 環境変数を使用することができます。

  デフォルト値は **NO** です。Red Hat Enterprise Linux 6 では、設定ファイル内で値が **YES** に設定されています。

- **userlist_deny** — **userlist_enable** 指示文と共に使用して **NO** に設定すると、ユーザー名が **userlist_file** 指示文が特定したファイル内にリストされていない限りは、すべてのロー

カルユーザーのアクセスが否認されます。クライアントにパスワードを要求する前にアクセスが否定されるため、この指示文を **NO** に設定しておくと、ローカルユーザーがネットワーク上で暗号化されていないパスワードを送信するのを防止します。

デフォルト値は **YES** です。

- **userlist_enable** — 有効になると、**userlist_file** 指示文が指定するファイル内にリストされたユーザーは、アクセスを否認されます。クライアントがパスワードを要求する前にアクセスが否認されるため、ユーザーがネットワーク上で暗号化されていないパスワードを送信するのを防止します。

  デフォルト値は **NO** です。Red Hat Enterprise Linux 6 では、設定ファイル内で値が **YES** に設定されています。

- **userlist_file** — **userlist_enable** ディレクティブが有効化された時に、 **vsftpd** が参照するファイルを指定します。

  デフォルト値は **/etc/vsftpd/user_list** で、これはインストール中に作成されます。

18.2.5.3. 匿名ユーザーのオプション

以下は、匿名ユーザーによるサーバーへのアクセスを制御するディレクティブのリストです。これらのオプションを使用するには、**anonymous_enable** ディレクティブは **YES** に設定する必要があります。

- **anon_mkdir_write_enable** — **write_enable** ディレクティブと共に有効にすると、匿名ユーザーは書き込みパーミッションを持つ親ディレクトリ内に新しいディレクトリを作成することができるようになります。

  デフォルト値は **NO** です。

- **anon_root** — 匿名ユーザーのログイン後に **vsftpd** の変更先のディレクトリに指定します。

  この指示文にはデフォルト値はありません。

- **anon_upload_enable** — **write_enable** ディレクティブと共に有効にすると、匿名ユーザーは、書き込みパーミッションのある親ディレクトリ内でファイルをアップロードすることが可能となります。

  デフォルト値は **NO** です。

- **anon_world_readable_only** — 有効にすると、匿名ユーザーは全ユーザーが読み取り可能なファイルのみダウンロードできるようになります。

  デフォルト値は **YES** です。

- **ftp_username** — 匿名の **FTP** ユーザー用に使用される、ローカルユーザーアカウント (**/etc/passwd** にリストされている) を指定します。**/etc/passwd** で指定されているユーザー用のホームディレクトリは、匿名 **FTP** ユーザーの root ディレクトリです。

  デフォルト値は **ftp** です。

- **no_anon_password** — 有効にすると、匿名ユーザーはパスワードを要求されません。

  デフォルト値は **NO** です。

- **secure_email_list_enable** — 有効にすると、指定されたリストの匿名ログイン用 Email パスワードのみが受け入れられます。これは、仮想ユーザーの必要なしに、パブリックコンテンツに対して限定的なセキュリティを提供する便利な方法です。

  提供されたパスワードが **/etc/vsftpd/email_passwords** にリストされていなければ、匿名ログインはできません。ファイル形式は、一行につきパスワード 1 つで、末尾にスペースは入りません。

  デフォルト値は **NO** です。

18.2.5.4. ローカルユーザーのオプション

以下は、ローカルユーザーがサーバーにアクセスする方法を特徴付けるディレクティブの一覧です。これらのオプションを使用するには、**local_enable** ディレクティブを **YES** に設定する必要があります。

- **chmod_enable** — 有効になると、ローカルユーザーに **FTP** コマンド **SITE CHMOD** が許可されます。このコマンドにより、ユーザーはファイル上のパーミッションを変更することができるようになります。

  デフォルト値は **YES** です。

- **chroot_list_enable** — 有効になると、**chroot_list_file** ディレクティブで指定されているファイルにリストされているローカルユーザーは、ログイン時に **chroot** jail に配置されます。

  chroot_local_user ディレクティブと共に有効化されると、**chroot_list_file** ディレクティブで指定されているファイルにリストされているローカルユーザーは、ログイン時に **chroot** jail には **配置されません**。

  デフォルト値は **NO** です。

- **chroot_list_file** — **chroot_list_enable** ディレクティブが **YES** に設定された場合に参照されるローカルユーザーのリストを格納するファイルを指定します。

  デフォルト値は **/etc/vsftpd/chroot_list** です。

- **chroot_local_user** — 有効になると、ログイン後に、ローカルユーザーのホームディレクトリにルートディレクトリが変更されます。

  デフォルト値は **NO** です。

警告

chroot_local_user を有効にすると、特にアップロードの権限を持つユーザーに関して、多くのセキュリティ問題が発生する可能性が出てきます。このため、このオプションは **推奨されません**。

- **guest_enable** — 有効になると、非匿名ユーザーは **guest_username** ディレクティブで指定されているローカルユーザーである ユーザー **guest** としてログインします。

デフォルト値は **NO** です。

- **guest_username** — **guest** ユーザーがマッピングされるユーザー名を指定します。

  デフォルト値は **ftp** です。

- **local_root** — ローカルユーザーのログイン後に **vsftpd** の変更先となるディレクトリを指定します。

  この指示文にはデフォルト値はありません。

- **local_umask** — ファイル作成の umask 値を指定します。デフォルト値は、8 進法の形式 (8 を基数とする数値システム) で、"0" プレフィックスが含まれている点に注意してください。その例外の場合は、10 進法の整数として扱われます。

  デフォルト値は **077** です。Red Hat Enterprise Linux 6 では、このオプションは設定ファイル内で **022** に設定されています。

- **passwd_chroot_enable** — **chroot_local_user** ディレクティブと共に有効になると、**vsftpd** は **/etc/passwd** のホームディレクトリフィールド内の **/./** の発生に基づいてローカルユーザーのルートディレクトリを変更します。

  デフォルト値は **NO** です。

- **user_config_dir** — ローカルシステムユーザーの名前が付いた、そのユーザー固有の設定が格納されている設定ファイルを格納するディレクトリへのパスを指定します。ユーザーの設定ファイル内のディレクティブは、**/etc/vsftpd/vsftpd.conf** の設定ファイルを上書きします。

  この指示文にはデフォルト値はありません。

18.2.5.5. ディレクトリのオプション

以下は、ディレクトリに影響を及ぼすディレクティブの一覧です。

- **dirlist_enable** — 有効になると、ユーザーはディレクトリー覧を閲覧することができるようになります。

  デフォルト値は **YES** です。

- **dirmessage_enable** — 有効になると、メッセージファイルのあるディレクトリにユーザーが入った際には常にメッセージが表示されます。このメッセージは、作業中のディレクトリ内に常駐します。このファイル名は、**message_file** ディレクティブ内で指定されており、デフォルトでは **.message** です。

  デフォルト値は **NO** です。Red Hat Enterprise Linux 6 では、設定ファイル内で値が **YES** に設定されています。

- **force_dot_files** — 有効になると、ドット (**.**) で始まるファイルは、ディレクトリー覧にリストされますが、**.** ファイルおよび **..** ファイルは例外となります。

  デフォルト値は **NO** です。

- **hide_ids** — 有効になると、すべてのディレクトリー覧で **ftp** は各ファイルのユーザーおよびグループとして表示されます。

デフォルト値は **NO** です。

- **message_file** — **dirmessage_enable** ディレクティブを使用する場合にメッセージファイルの名前を指定します。

  デフォルト値は **.message** です。

- **text_userdb_names** — 有効になると、テストユーザー名とグループ名が UID と GID エントリーの代わりに使用されます。このオプションを有効にすると、サーバーのパフォーマンスが低下する可能性があります。

  デフォルト値は **NO** です。

- **use_localtime** — 有効になると、ディレクトリー覧に GMT ではなく現地時間が表示されます。

  デフォルト値は **NO** です。

18.2.5.6. ファイル転送のオプション

以下は、ディレクトリに影響を及ぼすディレクティブの一覧です。

- **download_enable** — 有効になると、ファイルのダウンロードが許可されます。

  デフォルト値は **YES** です。

- **chown_uploads** — 有効になると、匿名ユーザーがアップロードしたファイルはすべて、**chown_username** ディレクティブで指定されたユーザーが所有することになります。

  デフォルト値は **NO** です。

- **chown_username** — **chown_uploads** ディレクティブが有効化されている場合に、匿名でアップロードされたファイルの所有権を指定します。

  デフォルト値は **root** です。

- **write_enable** — 有効になると、ファイルシステムの変更が可能な **DELE**、**RNFR**、および **STOR** などの **FTP** コマンドが使用可能となります。

  デフォルト値は **NO** です。Red Hat Enterprise Linux 6 では、設定ファイル内で値が **YES** に設定されています。

18.2.5.7. ロギングのオプション

以下は、**vsftpd** のロギング動作に影響を及ぼすディレクティブのリストです。

- **dual_log_enable** — **xferlog_enable** と共に有効になると、**vsftpd** は次にあげる 2 つのファイルに同時に書き込みを行います：**xferlog_file** ディレクティブで指定されたファイル (デフォルトでは **/var/log/xferlog**) に対する **wu-ftpd** 互換のログ、ならびに**vsftpd_log_file** ディレクティブ (デフォルトでは **/var/log/vsftpd.log**) で指定されている標準の **vsftpd** ログファイル。

  デフォルト値は **NO** です。

- **log_ftp_protocol** — **xferlog_enable** と共に有効になり、**xferlog_std_format** が **NO** に設定されると、すべての **FTP** コマンドと応答がログ記録されます。このディレクティブはデバッグに有用です。

  デフォルト値は **NO** です。

- **syslog_enable** — **xferlog_enable** と共に有効になると、**vsftpd_log_file** ディレクティブ (デフォルトでは **/var/log/vsftpd.log**) が指定する標準の **vsftpd** ログファイルに通常書き込まれるすべてのログは、代わりに、**FTPD** ファシリティ下のシステムロガーに送信されます。

  デフォルト値は **NO** です。

- **vsftpd_log_file** — **vsftpd** ログファイルを指定します。このファイルを使用するには、**xferlog_enable** を有効化し、かつ **xferlog_std_format** を **NO** に設定するか、またはそれが **YES** に設定されている場合は、**dual_log_enable** を有効化する必要があります。**syslog_enable** が **YES** に設定されている場合、このディレクティブで指定したファイルではなく、システムログが使用されることは、注意すべき重要なポイントです。

  デフォルト値は **/var/log/vsftpd.log** です。

- **xferlog_enable** — 有効になると、**vsftpd** は接続 (**vsftpd** 形式のみ) およびファイル転送情報を、**vsftpd_log_file** ディレクティブ (デフォルトでは **/var/log/vsftpd.log**) で指定されているログファイルにログ記録します。**xferlog_std_format** が **YES** に設定されている場合、ファイル転送情報はログ記録されますが、接続についてはログ記録されず、**xferlog_file** (デフォルトでは **/var/log/xferlog**) で指定されているログファイルが代わりに使用されます。**dual_log_enable** が **YES** 設定されている場合には、ログファイルとログ形式の両方が使用されることは注意すべき重要なポイントです。

  デフォルト値は **NO** です。Red Hat Enterprise Linux 6 では、設定ファイル内で値が **YES** に設定されています。

- **xferlog_file** — **wu-ftpd** 互換のログファイルを指定します。このファイルを使用するには、**xferlog_enable** を有効にし、かつ **xferlog_std_format** を **YES** に設定する必要があります。これは、**dual_log_enable** が **YES** に設定されている場合にも使用されます。

  デフォルト値は **/var/log/xferlog** です。

- **xferlog_std_format** — **xferlog_enable** と合わせて有効になると、**wu-ftpd** 互換のファイル転送ログのみが **xferlog_file** ディレクティブ (デフォルトでは **/var/log/xferlog**) で指定されているファイルに書き込まれます。このファイルにロギングされるのはファイル転送のみで、サーバーへの接続をログしないことは注意すべき重要なポイントです。

  デフォルト値は **NO** です。Red Hat Enterprise Linux 6 では、設定ファイル内で値が **YES** に設定されています。

重要

旧式の **wu-ftpd FTP** サーバーによって書き込まれたログファイルとの互換性を維持するために、Red Hat Enterprise Linux の環境下では、**xferlog_std_format** ディレクティブは **YES** に設定されています。ただし、この設定では、サーバー接続はログ記録されないことになります。

vsftpd 形式で接続をログ記録し、かつ **wu-ftpd** 互換のファイル転送ログを維持するには、**dual_log_enable** を **YES** に設定します。

wu-ftpd 互換のファイル転送ログを維持することが重要でない場合は、**xferlog_std_format** を **NO** に設定する、その行にハッシュマーク (#) 付きのコメントを書き込む、その行を完全に削除する、のいずれかを行います。

18.2.5.8. ネットワークのオプション

以下は、**vsftpd** がネットワークとのインタラクションを行う方法に影響を及ぼすディレクティブの一覧です。

- **accept_timeout** — パッシブモードを使用するクライアントが接続を確立する時間を指定します。

  デフォルト値は **60** です。

- **anon_max_rate** — 匿名ユーザー用の最大データ転送速度をバイト/秒で指定します。

  デフォルト値は **0** で、転送速度は制限されません。

- **connect_from_port_20** 有効になると、アクティブモードでのデータ転送中にサーバー上のポート 20 を開くのに十分な権限で **vsftpd** が実行されます。このオプションを無効にすると、**vsftpd** がより低い権限で実行されますが、一部の **FTP** クライアントとの互換性を欠く可能性があります。

  デフォルト値は **NO** です。Red Hat Enterprise Linux 6 では、設定ファイル内で値が **YES** に設定されています。

- **connect_timeout** — アクティブモードを使用するクライアントがデータ接続に応答する際の最大限の時間を秒数で指定します。

  デフォルト値は **60** です。

- **data_connection_timeout** — データ転送でストール (stall) することができる最大時間を秒数で指定します。トリガーされるとリモートクライアントへの接続は閉鎖されます。

  デフォルト値は **300** です。

- **ftp_data_port** — **connect_from_port_20** が **YES** に設定されている場合に、アクティブなデータ接続に使用するポートを指定します。

  デフォルト値は **20** です。

- **idle_session_timeout** — リモートクライアントからのコマンドの最大時間間隔を指定します。トリガーされると、リモートクライアントへの接続は閉鎖されます。

  デフォルト値は **300** です。

- **listen_address** — **vsftpd** がネットワーク接続をリッスンする **IP** アドレスを指定します。

  この指示文にはデフォルト値はありません。

  注記

  異なる **IP** アドレスを提供する **vsftpd** の複数コピーを実行する場合は、**vsftpd** デーモンの各コピー用の設定ファイルには、この指示文用に異なる値がなければなりません。マルチホームの **FTP** サーバーについての更なる情報には、「vsftpd の複数コピーの起動」 をご参照ください。

- **listen_address6** — **listen_ipv6** が **YES** に設定されている場合に、**vsftpd** がネットワーク接続をリッスンする **IPv6** アドレスを指定します。

  この指示文にはデフォルト値はありません。

  注記

  異なる **IP** アドレスを提供する **vsftpd** の複数コピーを実行する場合は、**vsftpd** デーモンの各コピー用の設定ファイルには、この指示文用に異なる値がなければなりません。マルチホームの **FTP** サーバーについての更なる情報には、「vsftpd の複数コピーの起動」 をご参照ください。

- **listen_port** — **vsftpd** がネットワーク接続をリッスンするポートを指定します。

  デフォルト値は **21** です。

- **local_max_rate** — サーバーにログインしたローカルユーザーを対象とするデータ転送の最大速度をバイト/秒で指定します。

  デフォルト値は **0** で、転送速度は制限されません。

- **max_clients** — サーバーがスタンドアロンモードで稼働している時に、そのサーバーに同時に接続可能なクライアントの最大数を指定します。クライアントの接続がこの数を超えると、エラーメッセージが表示されます。

  デフォルト値は **0** で、接続数は制限されません。

- **max_per_ip** — 同一ソースの **IP** アドレスからの接続できるクライアントの最大数を指定します。

  デフォルト値は **0** で、接続数は制限されません。

- **pasv_address** — Network Address Translation (NAT) ファイアウォールの背後にあるサーバーのパブリック用 **IP** アドレス向けに **IP** アドレスを指定します。これにより、**vsftpd** はパッシブモード接続用の正しい返信アドレスを渡すことができます。

  この指示文にはデフォルト値はありません。

- **pasv_enable** — 有効になると、パッシブモードの接続が可能となります。

  デフォルト値は **YES** です。

- **pasv_max_port** — パッシブモード接続用に **FTP** クライアントに送信される可能な限り高いポートを指定します。この設定は、ポートの範囲を制限することでファイアウォールのルール作成が容易にするために使われます。

  デフォルト値は、**0** で、最高のパッシブポート範囲は制限されません。この値は、**65535** を超えてはなりません。

- **pasv_min_port** — パッシブモード接続用に **FTP** クライアントに送信される可能な限り低いポートを指定します。この設定は、ポートの範囲を制限することでファイアウォールのルール作成が容易にするために使われます。

  デフォルト値は **0** で、最低パッシブポートの範囲は制限されません。この値は、**1024** より低くなってはいけません。

- **pasv_promiscuous** — 有効となると、データ接続が同一の **IP** アドレスを起源とすることを確認するためのチェックは行われません。この設定は、特定タイプのトンネリングのみに有用です。

  警告

  データ転送を開始した制御接続として、同一の **IP** アドレスを起源とするパッシブモード接続を検証する重要なセキュリティ機能が無効になってしまうため、このオプションは、絶対的に必要でない限りは有効化しないでください。

  デフォルト値は **NO** です。

- **port_enable** — 有効になると、アクティブモードの接続が可能となります。

  デフォルト値は **YES** です。

18.2.6. その他のリソース

vsftpd についての更なる情報には、以下にあげるリソースをご参照ください。

18.2.6.1. インストールされているドキュメント

- **/usr/share/doc/vsftpd-*<version-number>*/** ディレクトリ — *<version-number>* の箇所を、インストールした **vsftpd** パッケージのバージョンに置き換えます。このディレクトリには、ソフトウェアの基本的な情報を記載した **README** が格納されています。**TUNING** ファイルには、基本的なパフォーマンス調整についてのヒントが、また **SECURITY/** ディレクトリには **vsftpd** で採用されているセキュリティモデルに関する情報が格納されています。

- **vsftpd** に関連した man ページ — デーモンおよび設定ファイルには数多くの man ページがあります。以下は、重要な man ページのリストです。

  サーバーアプリケーション

  - **man vsftpd** — **vsftpd** で利用可能なコマンドラインオプションについて説明しています。

設定ファイル

- **man vsftpd.conf** — **vsftpd** 用の設定ファイル内で利用可能なオプションの詳細な一覧を格納しています。
- **man 5 hosts_access** — **hosts.allow** および **hosts.deny** の TCP ラッパーの設定ファイル内で利用可能な書式とオプションについて説明しています。

18.2.6.2. 役に立つ Web サイト

- http://vsftpd.beasts.org/ — **vsftpd** プロジェクトページは、最新のドキュメントやソフトウェアの作成者の連絡先を入手することができる便利なサイトです。
- http://slacksite.com/other/ftp.html — この Web サイトは、アクティブモードとパッシブモードの **FTP** の相異点について簡明に説明しています。
- http://www.ietf.org/rfc/rfc0959.txt — IETF からの **FTP** プロトコルのオリジナルの *Request for Comments* (*RFC*)。

18.3. プリンターの設定

プリンター設定 (Printer Configuration) ツールを使うと、プリンター設定、プリンター設定ファイルの管理、印刷スプールディレクトリ、印刷フィルタ、プリンタークラスの管理ができます。

このツールはコモンユニックスプリンティングシステム (Common Unix Printing System : CUPS) を基本にしています。CUPS を使用した Red Hat Enterprise Linux バージョンからシステムをアップグレードする場合は、アップグレードプロセスは設定を行ったプリンターの設定を保持します。

注記

CUPS Web アプリケーションまたはコマンドラインから直接同じ、追加の動作をプリンターに実行できます。Web ブラウザでアプリケーションにアクセスするには http://localhost:631/ を開いてください。CUPS マニュアルについてはその Web サイトの **ホーム** タブのリンクを参照して下さい。

18.3.1. プリンター設定ツールの起動

プリンター設定ツールを使うと、既存のプリンターで様々な操作を実行したり、新規プリンターを設定することができます。CUPS を直接使用することも可能です (CUPS にアクセスするには http://localhost:631/ をクリックします)。

ツールを起動するには、パネルから **システム** → **管理** → **印刷** の順にクリックするか、コマンドラインで **system-config-printer** コマンドを実行します。

図18.3「プリンター設定のウィンドウ」 のような **Printer Configuration** ウィンドウが表示されます。

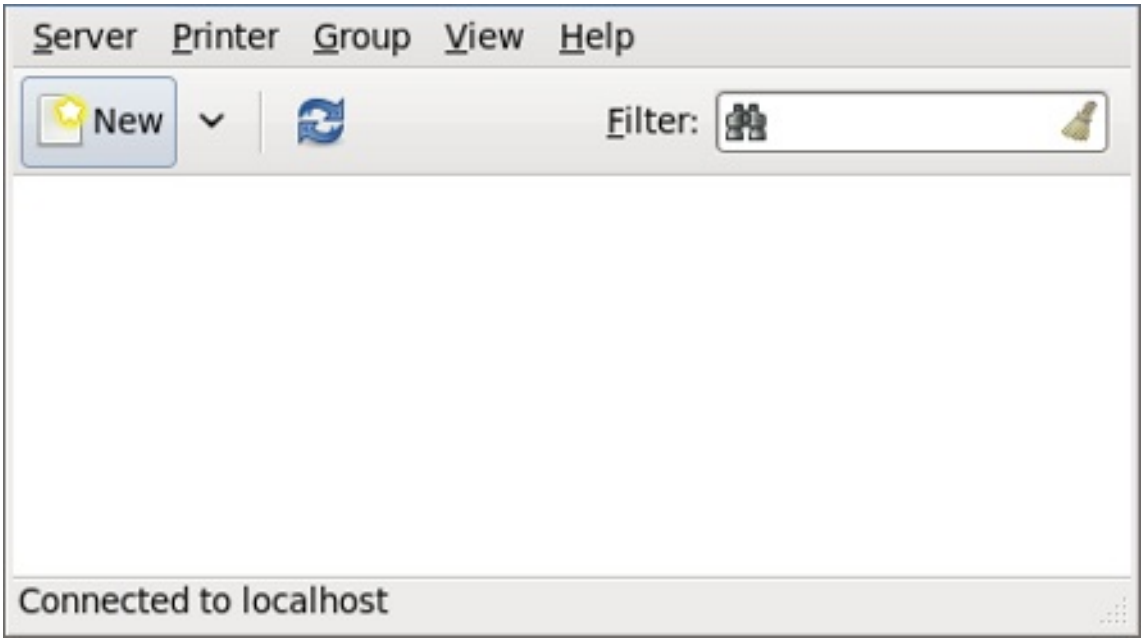


図18.3 プリンター設定のウィンドウ

18.3.2. プリンター設定の開始

プリンターの設定プロセスはプリンターキューのタイプにより異なります。

USB に接続したローカルプリンターを設定する場合は、プリンターは自動的に検出、追加されます。インストールしたいパッケージの確認、ルートパスワードの入力が求められます。他のポートタイプに接続したローカルプリンターやネットワークプリンターの場合は、手動で設定する必要があります。

プリンターを手動で設定するには、以下の手順に従います。

1. プリンター設定ツールを起動します (「プリンター設定ツールの起動」 を参照)。
2. **サーバー** → **新規** → **プリンター** の順にクリックします。
3. **Authenticate** ダイアログボックスで、確認のためルートユーザーパスワードを入力します。
4. プリンターの接続タイプを選択し、右側エリアでその詳細を記入します。

18.3.3. ローカルプリンターの追加

以下の手順に従い、シリアルポート以外に接続されたローカルプリンターを追加します。

1. **新規プリンター** ダイアログを開きます (「プリンター設定の開始」 を参照)。
2. デバイスが自動的に表示されない場合は、左側の一覧でプリンターを接続するポートを選択してください (**Serial Port #1** または **LPT #1** など)。
3. 右側で、接続プロパティを入力します:

   その他 の場合

   URI (例えば file:/dev/lp0)

   Serial Port の場合

   通信速度

パリティ

データビット

フロー制御

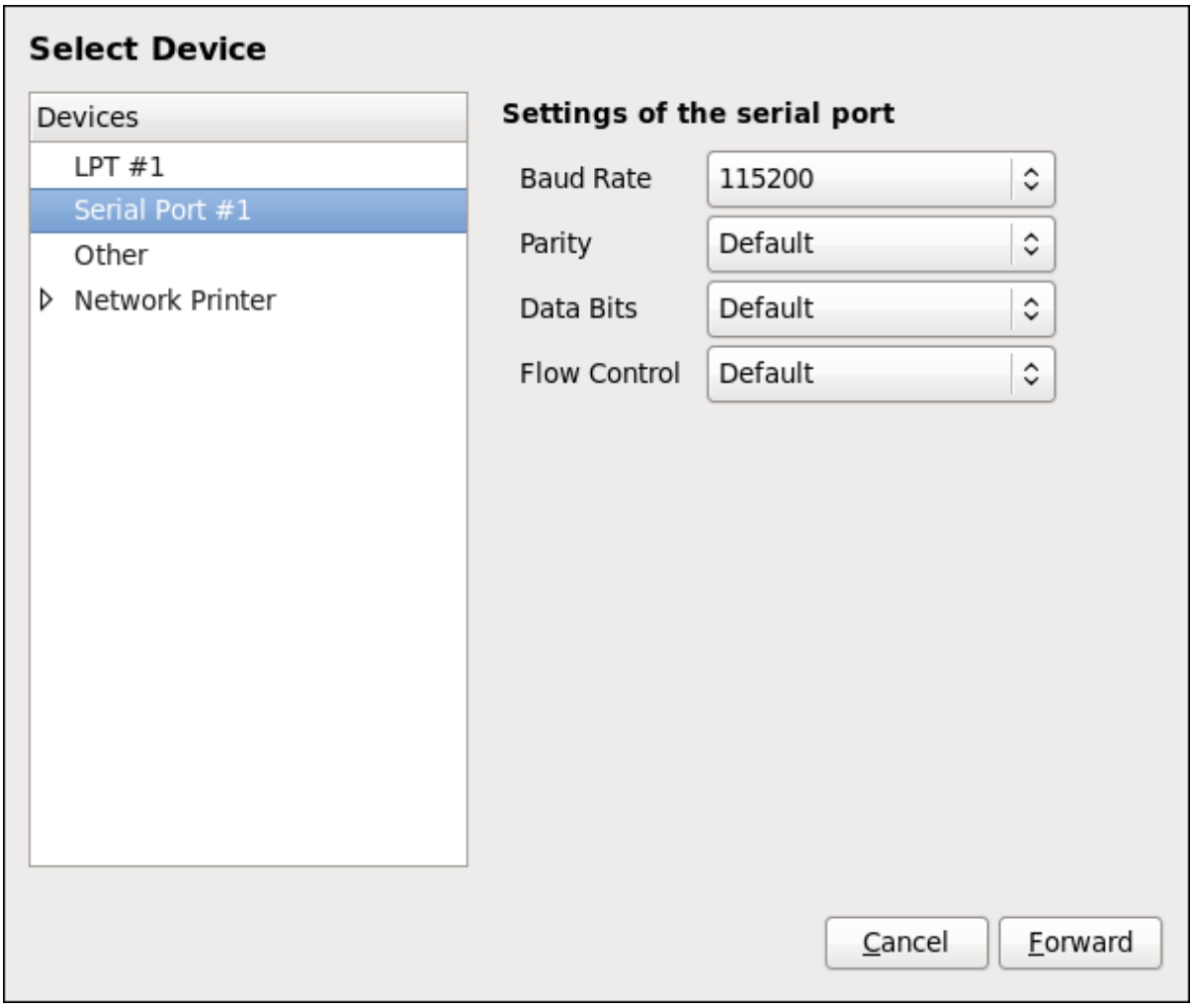


図18.4 ローカルプリンターの追加

4. **進む (Forward)** をクリックします。
5. プリンターのモデルを選択します。詳細は 「プリンターモデルの選択と完了」 を参照して下さい。

18.3.4. AppSocket/HP JetDirect プリンターの追加

以下の手順に従い、AppSocket/HP JetDirect プリンターを追加します。

1. **新規プリンター** ダイアログを開きます (「プリンター設定ツールの起動」 を参照)。
2. 左側の一覧で **ネットワークプリンター** → **AppSocket/HP JetDirect** の順に選択します。
3. 右側で、接続設定を入力します。

ホスト名

プリンターのホスト名または IP アドレス

ポート番号

印刷ジョブをリッスンするプリンターポート (デフォルトは **9100**)

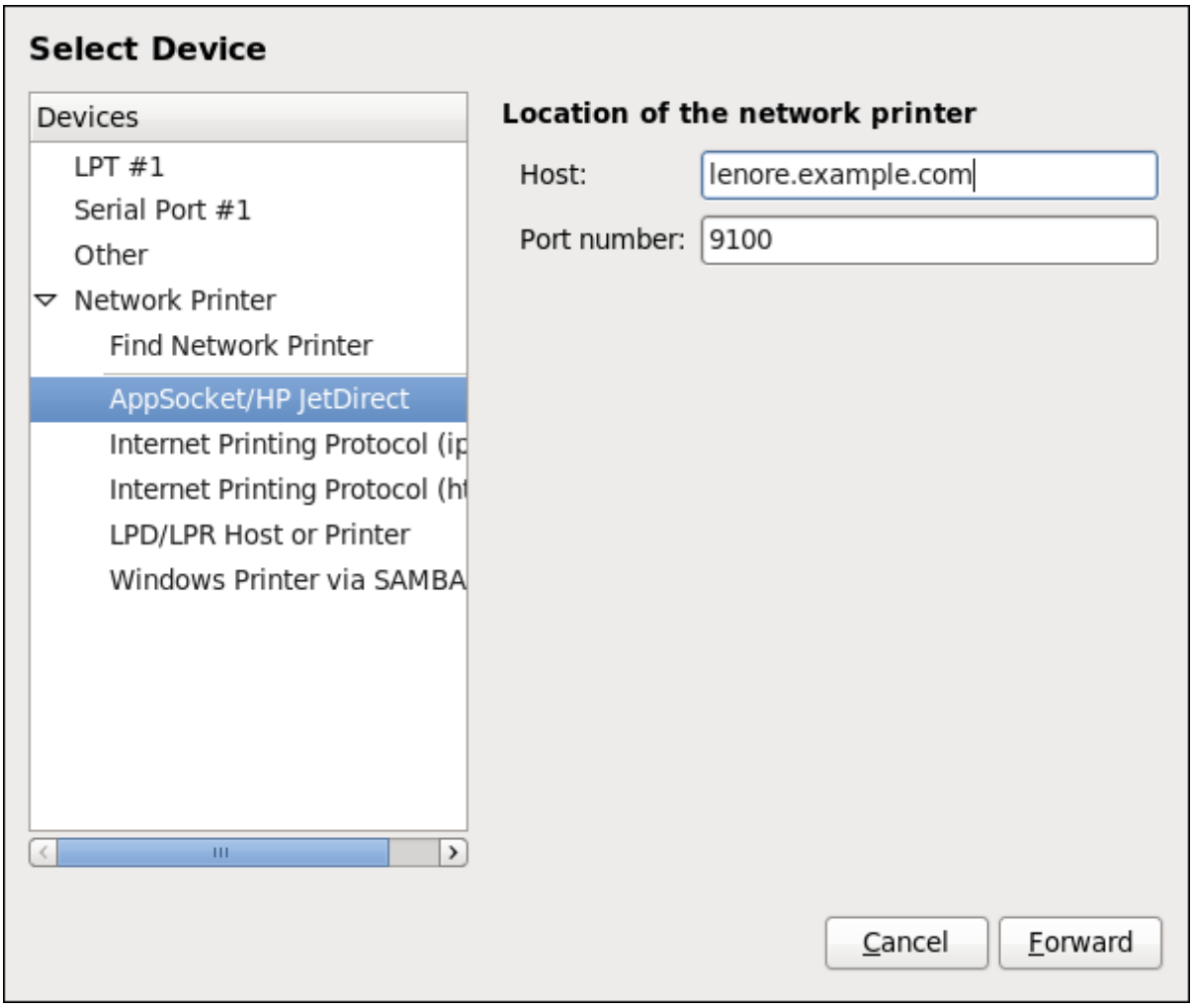


図18.5 JetDirect プリンターの追加

4. **進む (Forward)** をクリックします。

5. プリンターのモデルを選択します。詳細は 「プリンターモデルの選択と完了」 を参照して下さい。

18.3.5. IPP プリンターの追加

IPP プリンターは同じ TCP/IP ネットワーク上の異なるシステムにつながっているプリンターです。このプリンターが取り付けられているシステムは CUPS を実行しているか、単に IPP を使用するよう設定されているだけです。

プリンターサーバーでファイアウォールが有効な場合は、ポート 631 で受信 TCP 接続が可能なようにファイアウォールを設定する必要があります。プロトコルを参照する CUPS により、クライアントマシンは共有 CUPS キューを自動的に検出することが可能です。これを有効にするには、クライアントマシンのファイアウォールをポート 631 で受信 UDP パッケージを許可するよう設定する必要があります。

以下の手順に従い、IPP プリンターを追加します。

1. **新規プリンター** ダイアログを開きます (「プリンター設定の開始」 を参照)。

2. 左側のデバイスの一覧で、**ネットワークプリンター**、**Internet Printing Protocol (ipp)** または **Internet Printing Protocol (https)** の順に選択します。

3. 右側で、接続設定を入力します。

   ホスト

   IPP プリンターのホスト名

   キュー

   新規のキューに与えるキューの名前です(このボックスが空白のままであると、デバイスノードを基にした名前が使用されます)。

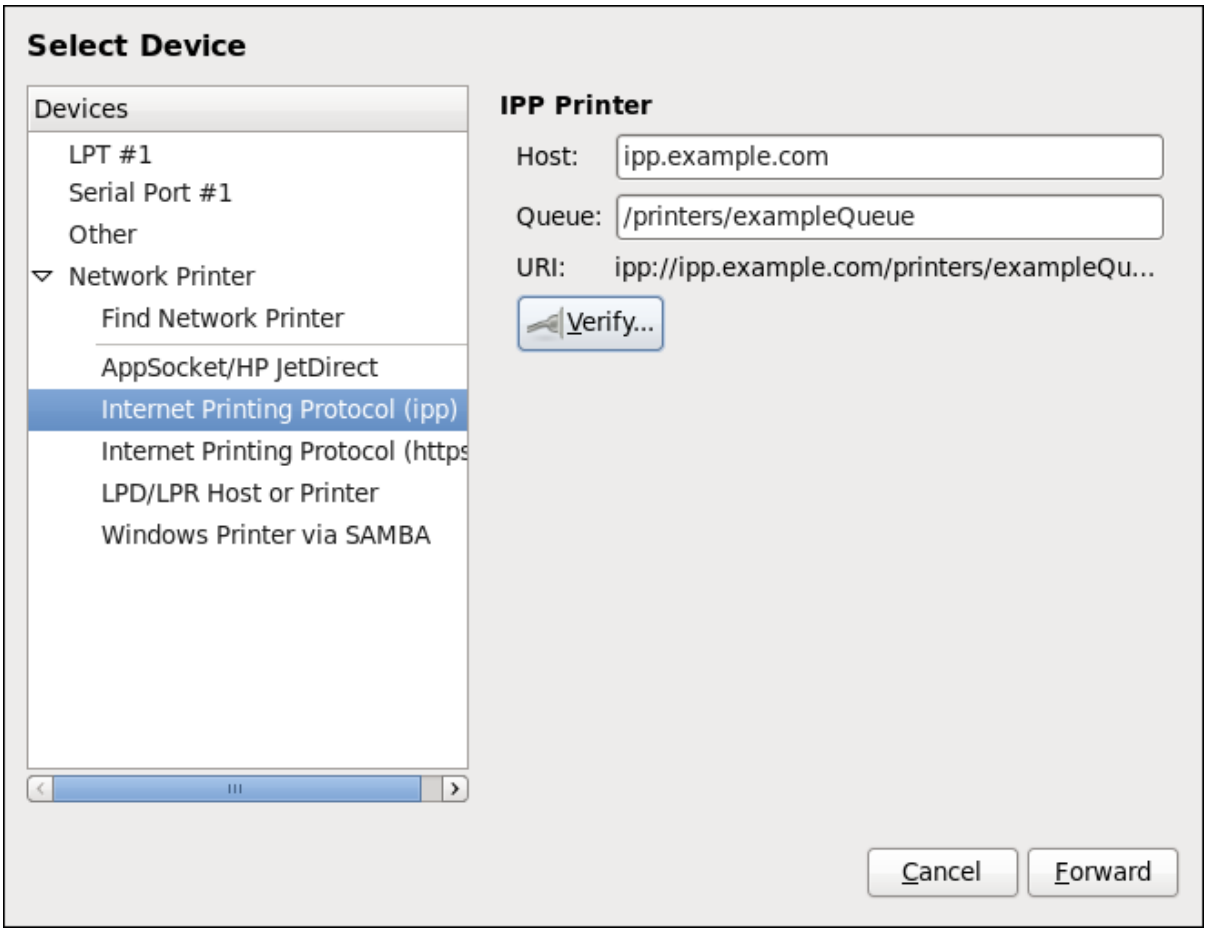


図18.6 IPP プリンターの追加

4. **進む** をクリックし、続けます。

5. プリンターのモデルを選択します。詳細は 「プリンターモデルの選択と完了」 を参照して下さい。

18.3.6. LPD/LPR Host or Printer の追加

以下の手順に従い、LPD/LPR host or printer を追加します。

1. **新規プリンター** ダイアログを開きます (「プリンター設定の開始」 を参照)。

2. 左のデバイス一覧から、**ネットワークプリンター** → **LPD/LPR Host or Printer** の順に選択します。

3. 右側で、接続設定を入力します。

   ホスト

   LPD/LPR printer or host のホスト名

   オプションで **Probe** をクリックすると、LPD ホスト上のキューを検索できます。

   キュー

   新規のキューに与えるキューの名前です(このボックスが空白のままであると、デバイスノードを基にした名前が使用されます)。

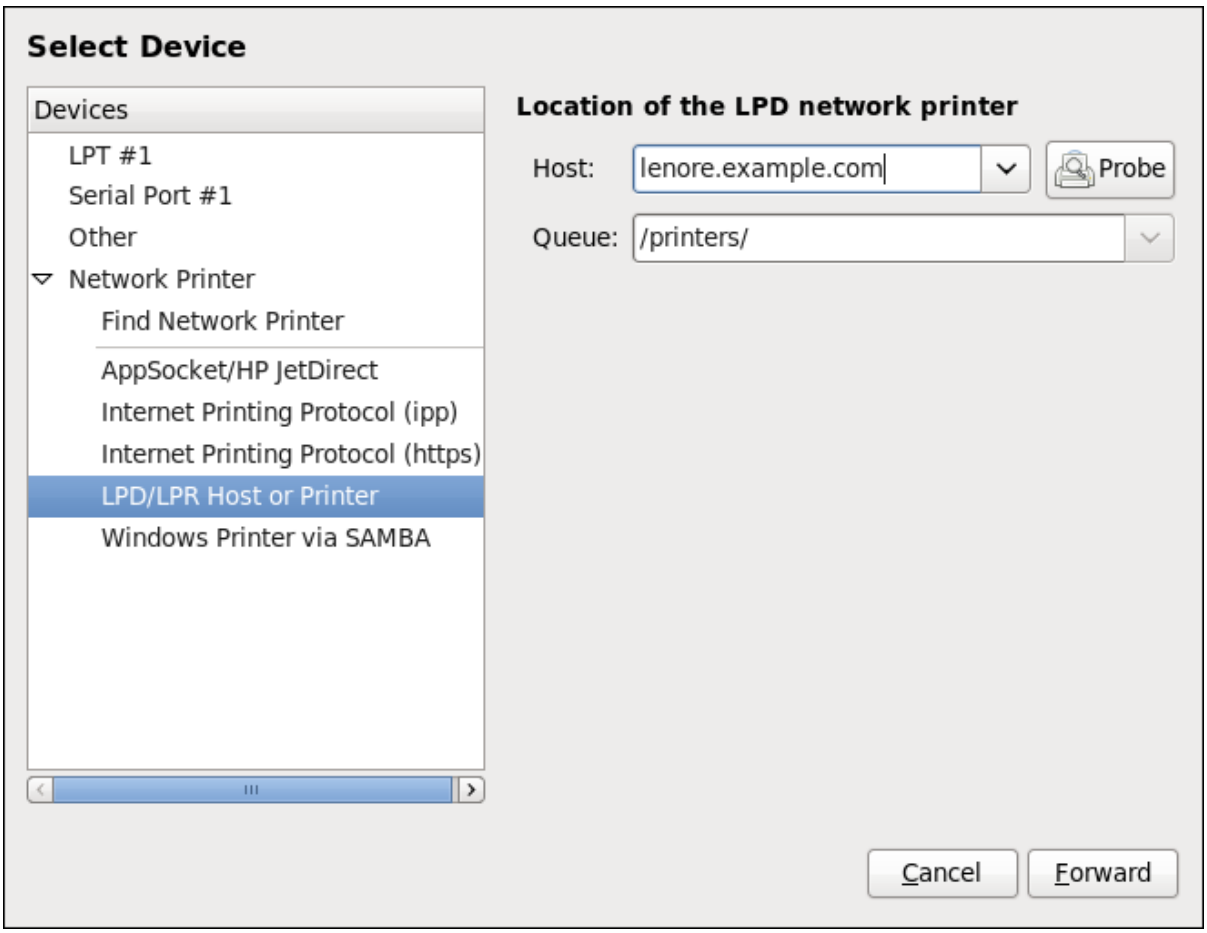


図18.7 LPD/LPR プリンターの追加

4. **進む** をクリックし、続けます。

5. プリンターのモデルを選択します。詳細は 「プリンターモデルの選択と完了」 を参照して下さい。

18.3.7. Samba (SMB) プリンターの追加

Samba プリンターを追加するには、以下の手順を実行します:

1. **新規プリンター** ダイアログを開きます (「プリンター設定の開始」 を参照)。

2. 左側の一覧で、**ネットワークプリンター** → **SAMBA 経由の Windows プリンター** の順に選択します。

3. **smb://** フィールドに SMB アドレスを入力します。入力形式は *computer name/printer share* にします。図18.8「SMB プリンターの追加」 では、*computer name* は **dellbox**、*printer share* は **r2** です。

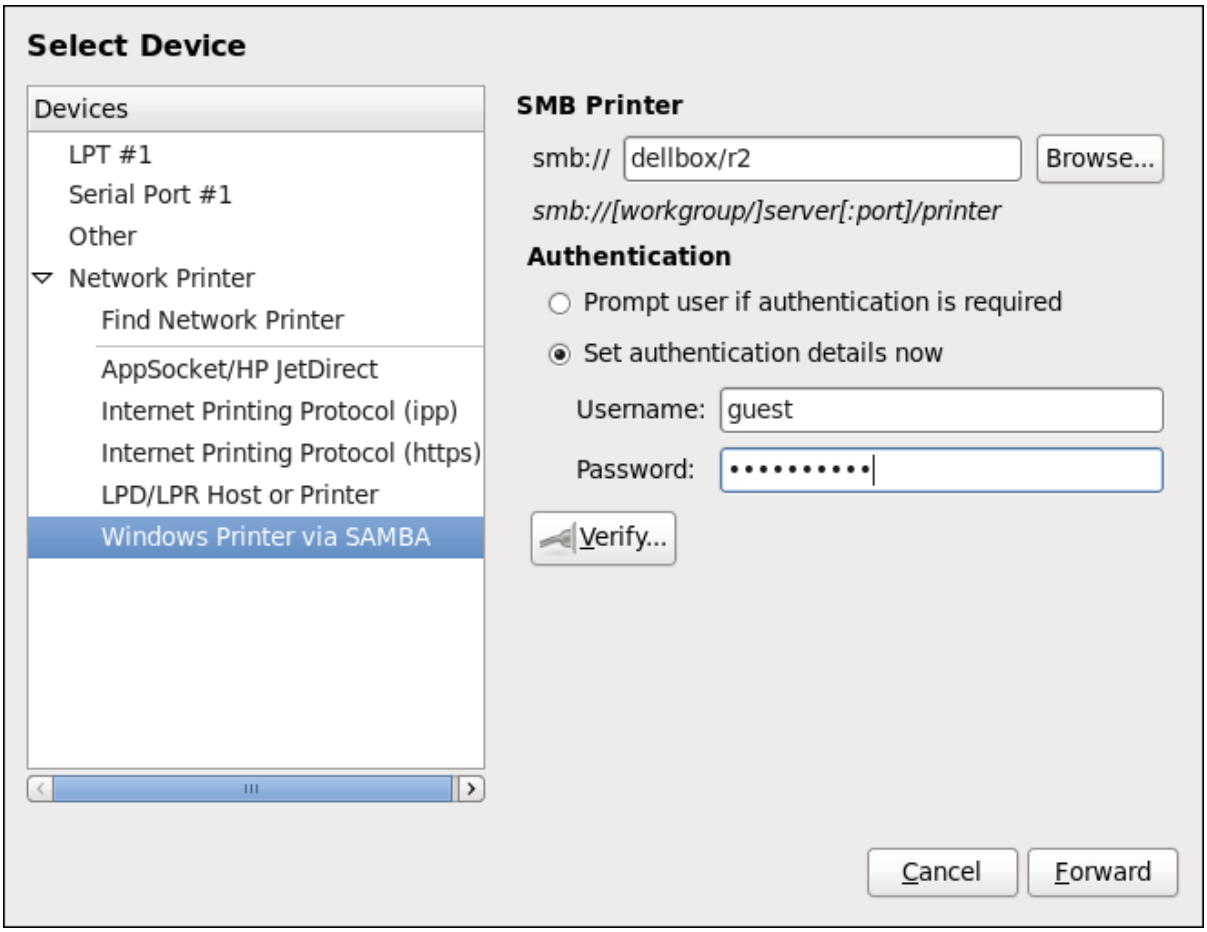


図18.8 SMB プリンターの追加

4. 利用できるワークグループやドメインを確認するには、**閲覧する (Browse)** をクリックします。特定のホストのキューだけを表示させるには、ホスト名 (NetBios 名) を入力して、**閲覧する** をクリックします。

5. 以下のオプションのどちらかを選択します。

   - **認証が必要は場合にはユーザーに催促する (Prompt user if authentication is required)** を選択すると、文書を印刷する時にはユーザーはユーザー名とパスワードを提供することになります。

   - **認証の詳細を今設定します (Set authentication details now)** を選択すると、今認証情報を提供するため後では入力が不必要となります。**ユーザー名** フィールドで、ユーザー名を入力してプリンターにアクセスします。このユーザーは SMB システムで存在している必要があり、ユーザーはプリンターへのアクセス権限を持っている必要があります。デフォルトのユーザー名は通常 Windows サーバーでは **guest** または Samba サーバーでは **nobody** です。

6. **ユーザー名** フィールドに指定したユーザーの **パスワード** (必要な場合は) を入力します。

警告

Samba プリンターのユーザー名とパスワードはプリンターサーバーにルート及び lpd が読み取り可能な非暗号化ファイルとして保存されています。そのため、プリンターサーバーにルートアクセスを持つ他のユーザーは、Samba プリンターへのアクセスに使用するユーザー名とパスワードを閲覧することができます。

Samba プリンターへアクセスするためのユーザー名とパスワードを選択する場合は、ローカルの Red Hat Enterprise Linux システムにアクセスする時に使用するパスワードとは異なるパスワードを選ぶことをお勧めします。

Samba 印刷サーバーで共有するファイルがある場合も、印刷キューで使用されるパスワードとは異なるパスワードを使用することが推奨されます。

7. **確認 (Verify)** をクリックし、接続をテストします。確認が成功すると、ダイアログボックスが表示され、プリンター共有のアクセスを確認します。

8. **進む (Forward)** をクリックします。

9. プリンターのモデルを選択します。詳細は 「プリンターモデルの選択と完了」 を参照して下さい。

18.3.8. プリンターモデルの選択と完了

正しくプリンターの接続タイプを選択すると、システムはドライバーを取得するよう試行します。プロセスが失敗した場合は、ドライバーリソースを手動で検索することができます。

以下の手順に従い、プリンタードライバーを設定してインストールを完了します。

1. 自動的にドライバーが検知され、ウィンドウが表示されます。以下のオプションのうちいずれかを選択します。

   - **データベースからプリンターを選択 (Select a Printer from database)** — システムは **製造元** の一覧からご使用のプリンターの選択した製造元に基づきドライバーを選択します。ご使用のプリンターのモデルが一覧にない場合は、**Generic** を選択してください。

   - **PPD ファイルを提供 (Provide PPD file)** — システムは備わっているポストスクリプトプリンタデスクリプション (PostScript Printer Description : PPD) を使用します。PPD ファイルは通常製造元が提供するプリンターに同梱されています。PPD ファイルが利用可能な場合は、このオプションを選択し、オプション詳細の下にあるブラウザバーを使用して、PPD ファイルを選択できます。

   - **ダウンロードするプリンタードライバーを検出 (Search for a printer driver to download)** — 製造元とご使用のプリンターモデルを **製造元とモデル** フィールドに入力し、OpenPrinting.org で適切なパッケージを検索します。

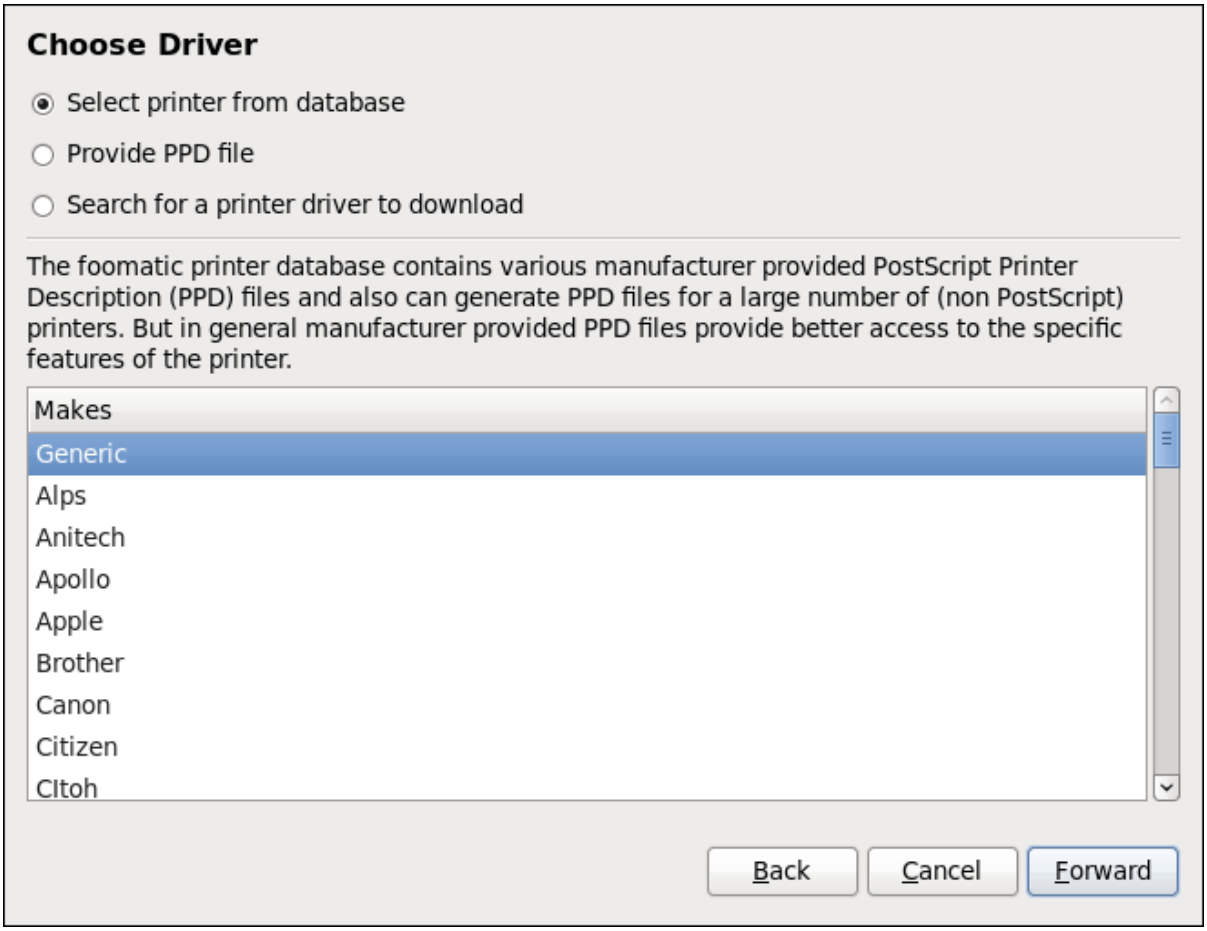


図18.9 プリンターブランドの選択

2. 以前に選択したものにより、以下に表示される詳細は異なります。

   - **データベースからプリンターを選択** オプションの場合は、プリンターブランドを表示
   - **PPD ファイルを提供** オプションの場合は、PPD ファイルの場所を表示
   - **ダウンロードするプリンタードライバーを検出** オプションの場合は、プリンターの製造元とモデルを表示

3. **進む** をクリックし、続けます。

4. 選択したオプションが該当する場合は、図18.10「プリンターモデルの選択」 のようなウィンドウが表示されます。左側の **モデル** の列で該当するモデルを選択します。

注記

右側で、推奨される印刷ドライバーが自動的に選択されています。ただし、別の利用可能なドライバーを選ぶことも可能です。印刷ドライバーは、プリンターが理解できる形式で印刷したいデータを処理します。ローカルプリンターは直接ご使用のコンピュータにつながっているため、プリンターに送られるデータを処理するプリンタードライバーが必要になります。

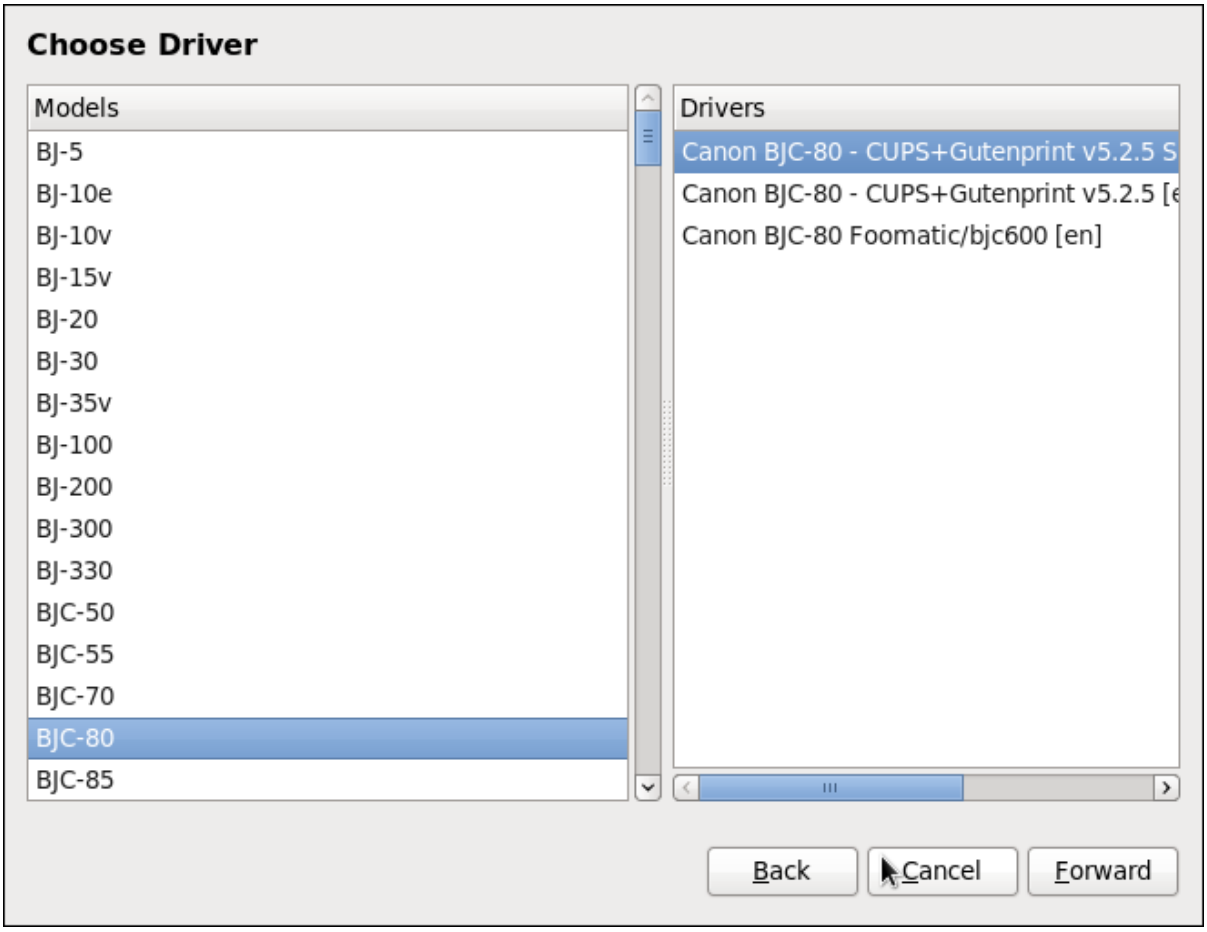


図18.10 プリンターモデルの選択

5. **進む (Forward)** をクリックします。

6. **プリンターの説明 (Describe Printer)** の下の、**プリンター名 (Printer Name)** フィールドに一意のプリンター名を入力します。名前には文字、数字、ダッシュ (-)、アンダースコア (_) を含むことができますが、スペースは含むことが **できません**。また、**説明 (Description)** と **場所 (Location)** フィールドに詳細情報を追加することも可能です。どちらもオプションで、スペースを入れることは可能です。

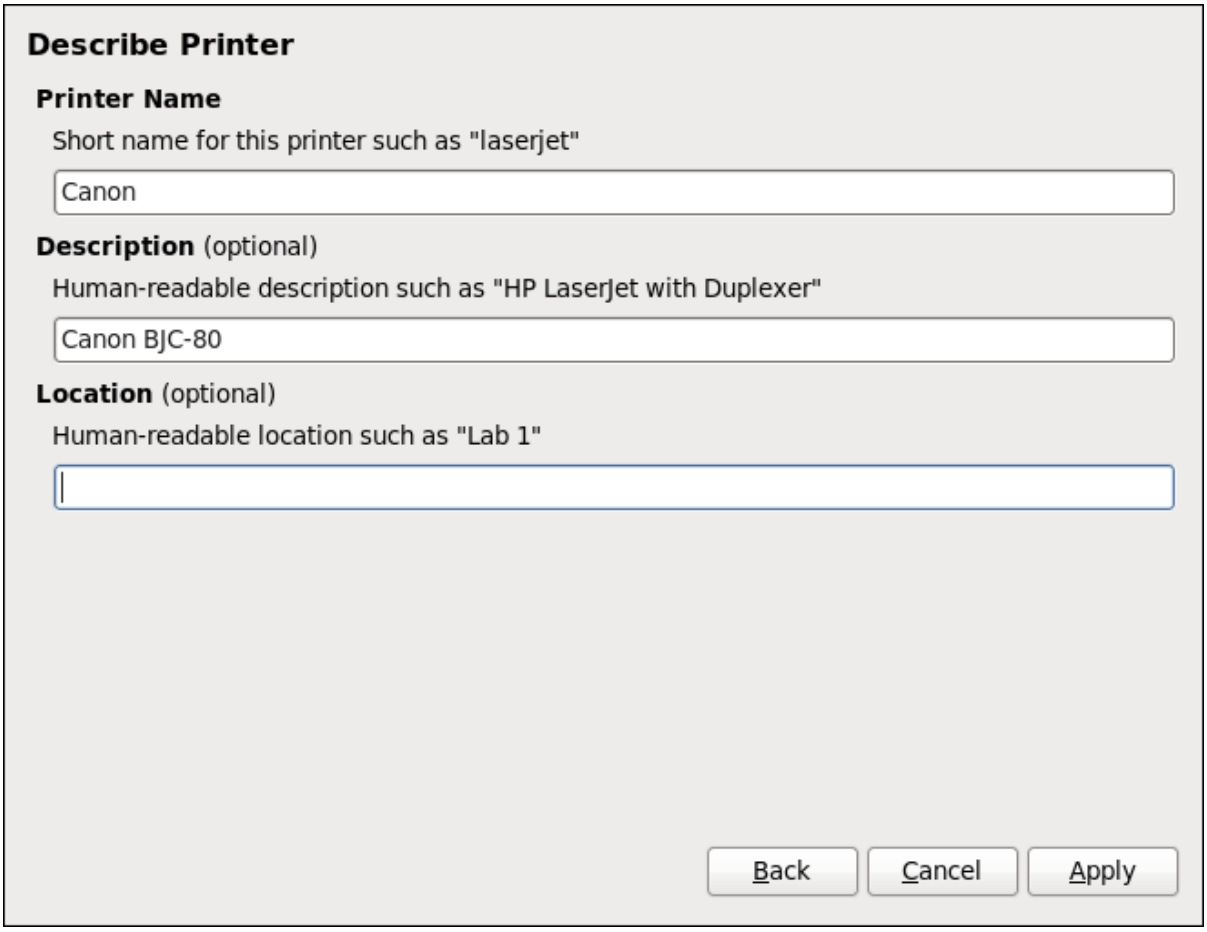


図18.11 プリンターの設定

7. 設定が正しければ、ご使用のプリンター設定を確認して、印刷キューを追加するために **適用 (Apply)** をクリックします。**戻る (Back)** をクリックすると、プリンター設定を変更できます。

8. 変更が適用されると、テストページの印刷を行うダイアログボックスが表示されます。**Yes** をクリックするとテストページが印刷されます。「テストページの印刷」 を参照してテストページを後で印刷することもできます。

18.3.9. テストページの印刷

プリンターを設定、またはプリンターの設定を変更した後は、テストページを印刷して、プリンターが適切に機能していることを確認します。

1. **印刷 (Printing)** ウィンドウでプリンターを右クリックし、**プロパティ (Properties)** をクリックします。

2. プロパティのウィンドウで、左側の **設定 (Settings)** をクリックします。

3. 表示されている **設定** タブで、**テストページの印刷 (Print Test Page)** ボタンをクリックします。

18.3.10. 既存プリンターの修正

既存のプリンターを削除するには、**プリンター設定** ウィンドウで該当するプリンターを選択し、**プリンター → 削除** の順に選択します。プリンターの削除を確認します。別の方法として、**Delete** キーを押しても削除できます。

デフォルトのプリンターを設定するには、プリンターの一覧で該当するプリンターを右クリックし、コンテキストメニューの **デフォルトに設定** ボタンをクリックします。

18.3.10.1. 設定のページ

プリンターのドライバー設定を変更するには、**プリンター** 一覧で該当する名前でダブルクリックします。そして、左側の **設定** ラベルをクリックし、**設定** ページを表示させます。

製造元やモデルなどのプリンター設定の変更、テストページの印刷、デバイスの場所 (URI) の変更など行うことができます。

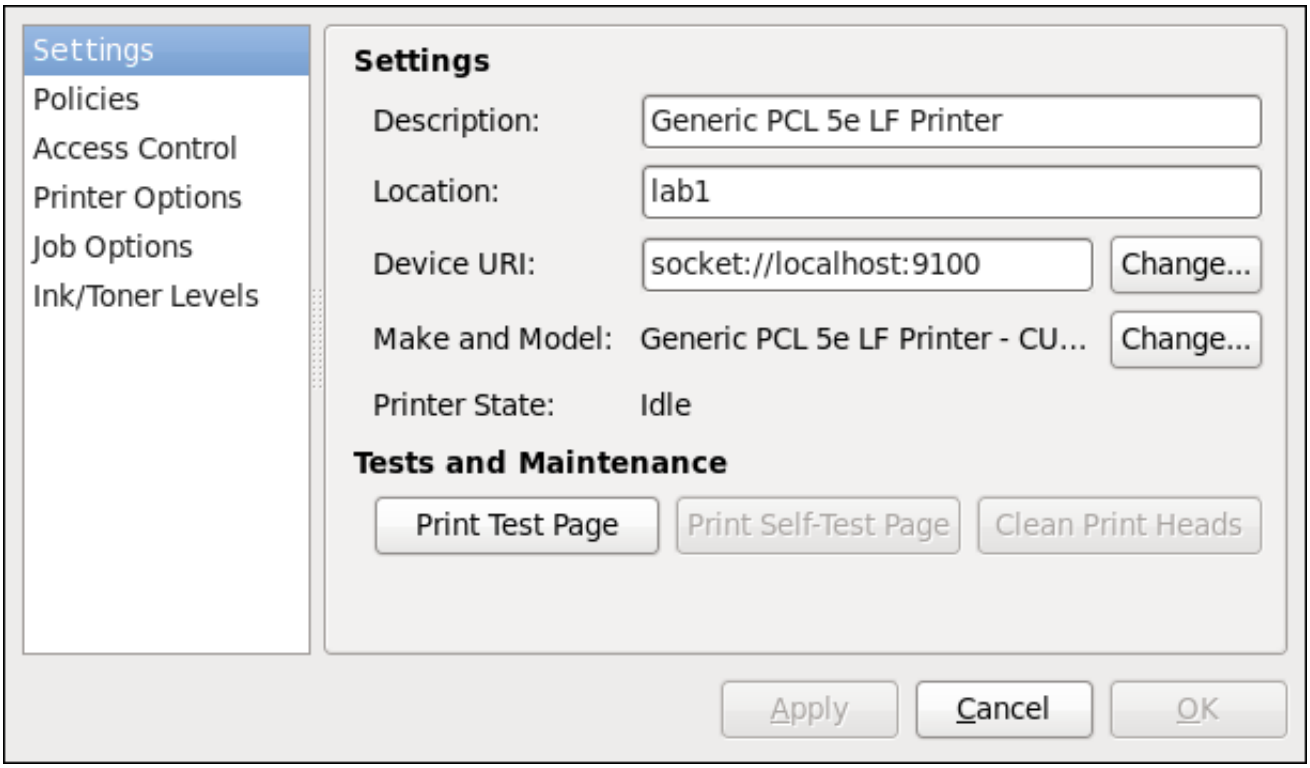


図18.12 設定のページ

18.3.10.2. ポリシーのページ

プリンターの状態や出力を変更するには、左側の **ポリシー (Policies)** ボタンをクリックします。

プリンターの状態を選択したり、プリンターの **エラーポリシー (Error Policy)** を設定することができます (エラーが発生した場合は、印刷ジョブを中止、再試行、停止できます)。

バナーページ (banner page) (送信元プリンター、ジョブを開始したユーザー名、印刷中の文書のセキュリティ状態など、印刷ジョブの特徴を説明するページ) の作成も可能です。**開始バナー (Starting Banner)** または **終了バナー (Ending Banner)** のドロップメニューをクリックし、印刷ジョブの性質に最適なオプションを選択します (**最上位秘密 (topsecret)**、**クラス分類 (classified)**、**秘密 (confidential)** など)。

18.3.10.2.1. プリンターの共有

ポリシー ページでは、プリンターを共有すると印を付けることができます。プリンターを共有にすると、ネットワーク上で公開されているユーザーは使用できます。プリンターの共有機能を有効にするには、**サーバー → 設定** の順にクリックし、**このシステムに接続されている共有プリンターを公開 (Publish shared printers connected to this system)** を選択します。

最後に、ファイアウォールがポート 631 への受信 TCP 接続 (system-config-firewall のネットワーク印刷サーバー) を許可していることを確認します。

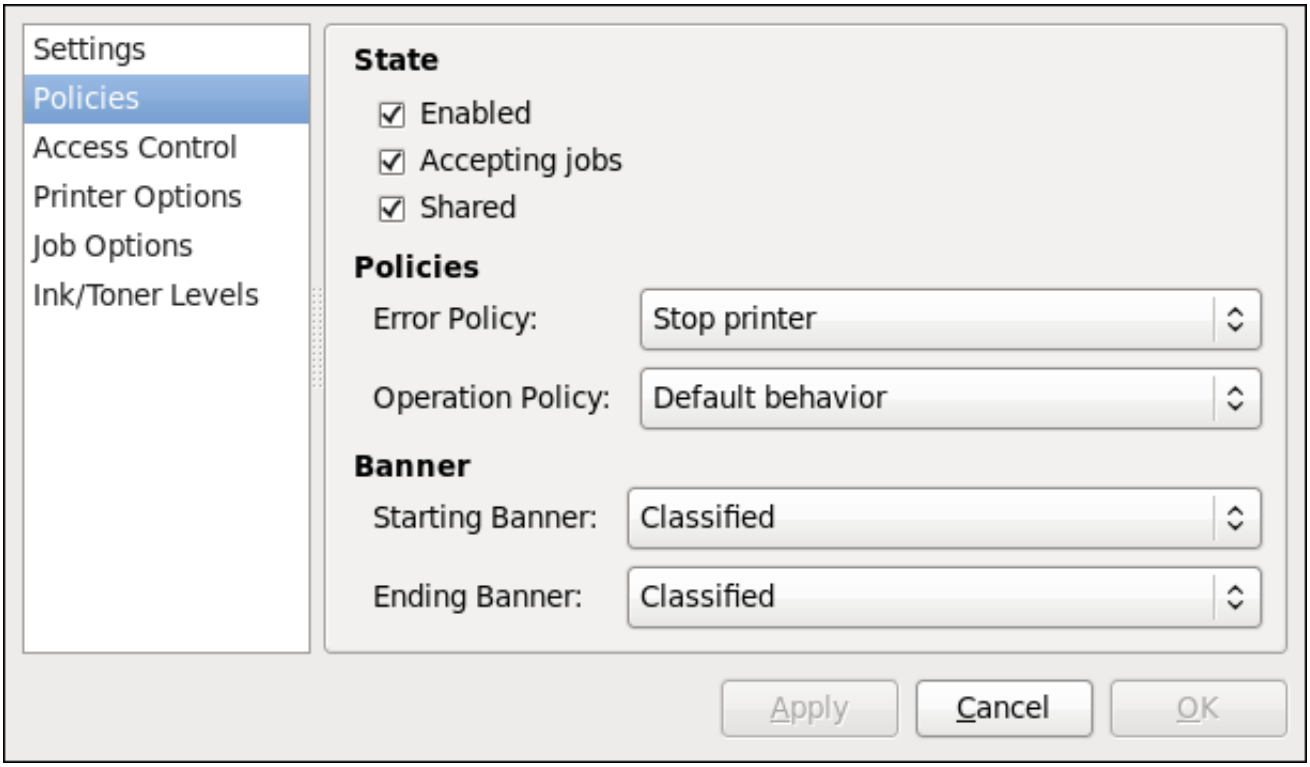


図18.13 ポリシーのページ

18.3.10.2.2. アクセス制御のページ

アクセス制御 ページで設定したプリンターへのユーザーレベルのアクセスを変更できます。左側の **アクセス制御 (Access Control)** のラベルをクリックするとページが表示されます。**次のユーザーを除いて全員に印刷を許可する (Allow printing for everyone except these users)** または **次のユーザー以外の印刷を拒否する (Deny printing for everyone except these users)** のどちらかを選択し、以下のようにユーザーセットを定義します。テキストボックスにユーザー名を入力し、**追加 (Add)** ボタンをクリックしてユーザーセットにユーザーを追加します。

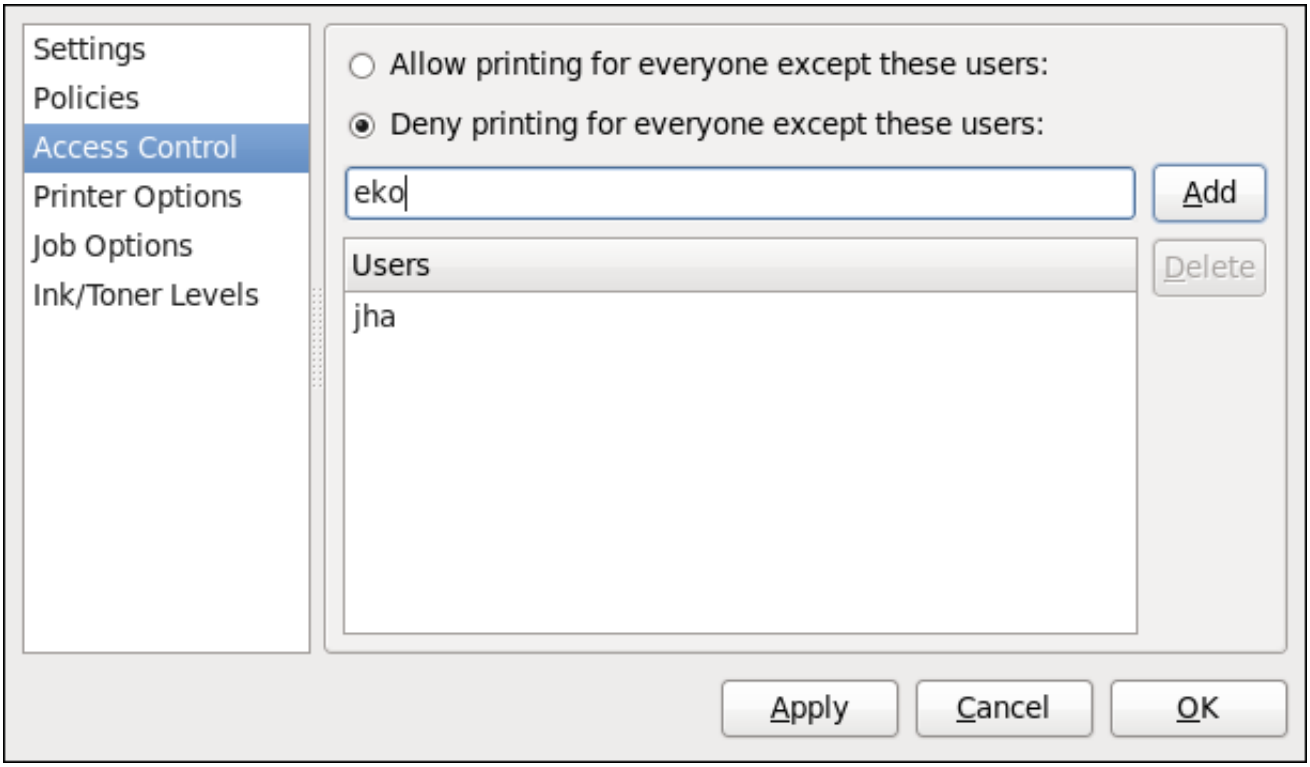


図18.14 アクセス制御のページ

18.3.10.2.3. プリンターオプションのページ

プリンターオプション (Printer Options) ページにはプリンターのメディアや出力の様々な設定オプションがあります。内容はプリンターごとに異なる場合があります。一般的な印刷の用紙、品質、サイズ設定が含まれます。

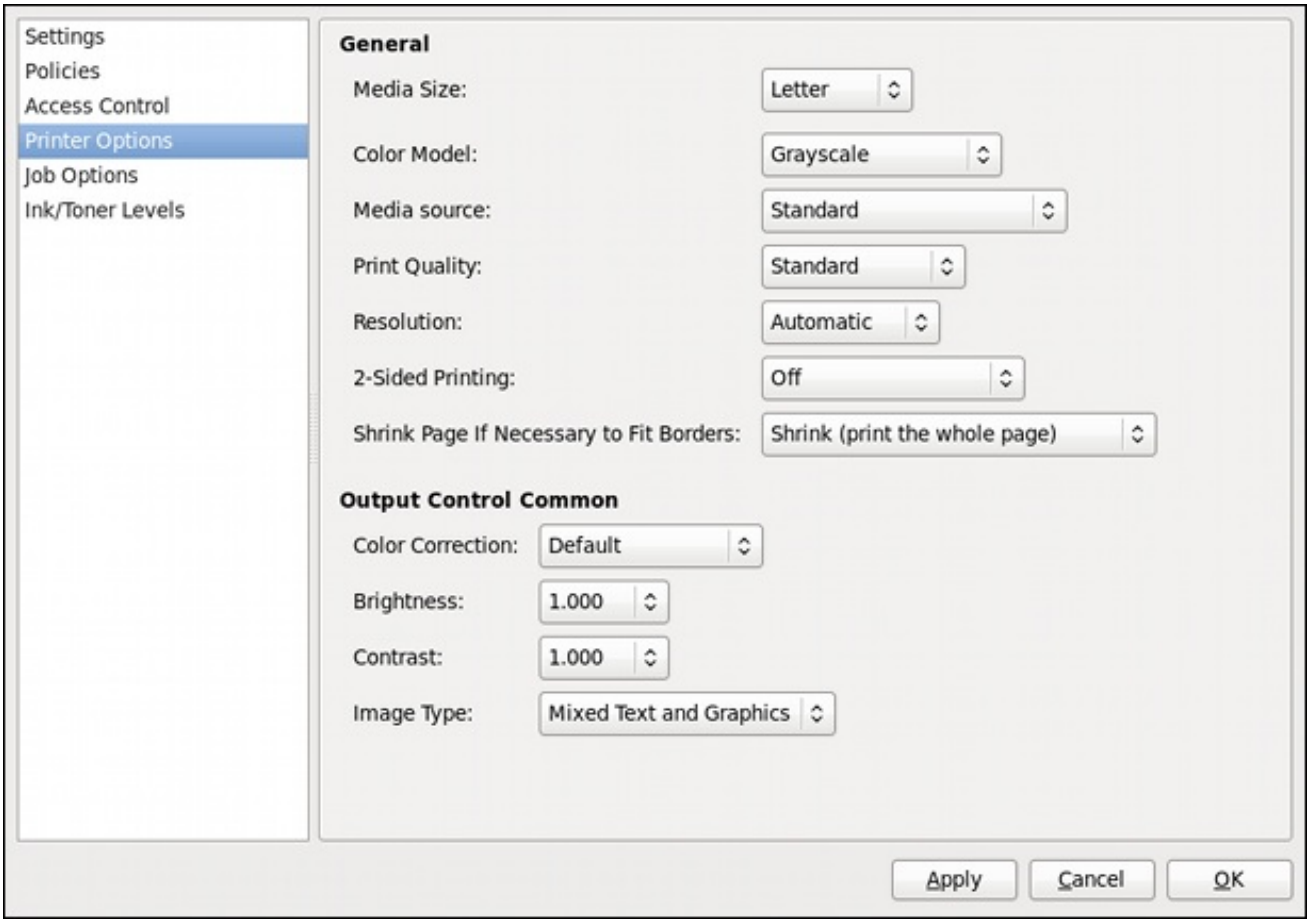


図18.15 プリンターオプションのページ

18.3.10.2.4. 依頼のオプションのページ

依頼のオプション (`Job Options`) ページでは、プリンターのジョブのオプションを詳細設定できます。左側の **依頼のオプション** をクリックすると、ページが表示されます。デフォルト設定を編集し、部数、印刷の向き、スライドごとのページ、拡大縮小 (印刷可能領域のサイズを拡大または縮小する。この機能によりサイズが印刷領域を超えるものを印刷媒体である用紙に合うようにします)、テキストオプションなど、カスタムの依頼のオプションを適用します。

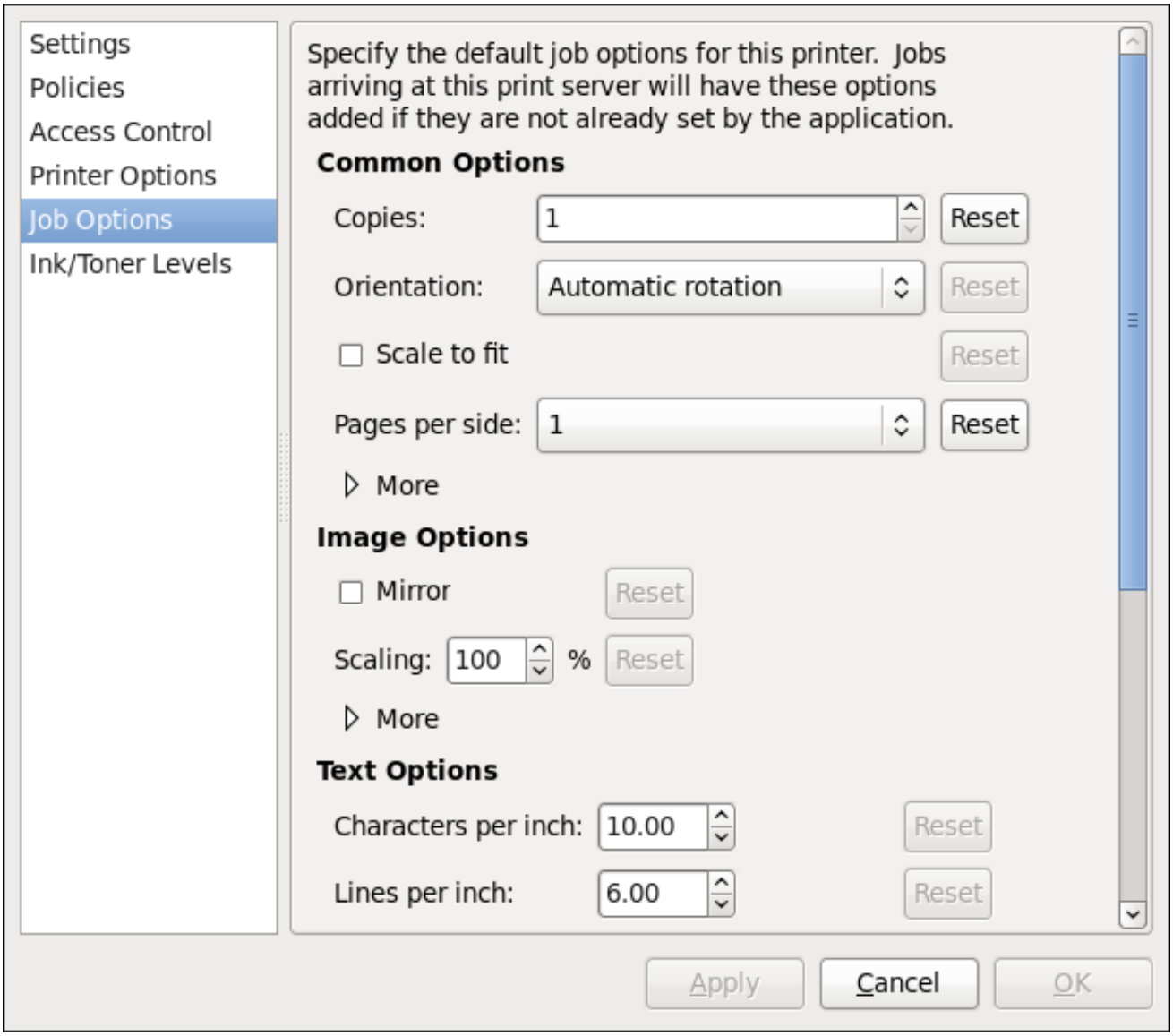


図18.16 依頼のオプションのページ

18.3.10.2.5. インク/トナーのレベルのページ

インク/トナーのレベル (Ink/Toner Levels) ページには該当する場合はトナーの状態の詳細、及びプリンターの状態のメッセージが表示されます。左側の **インク/トナーレベル** のラベルをクリックすると、ページが表示されます。

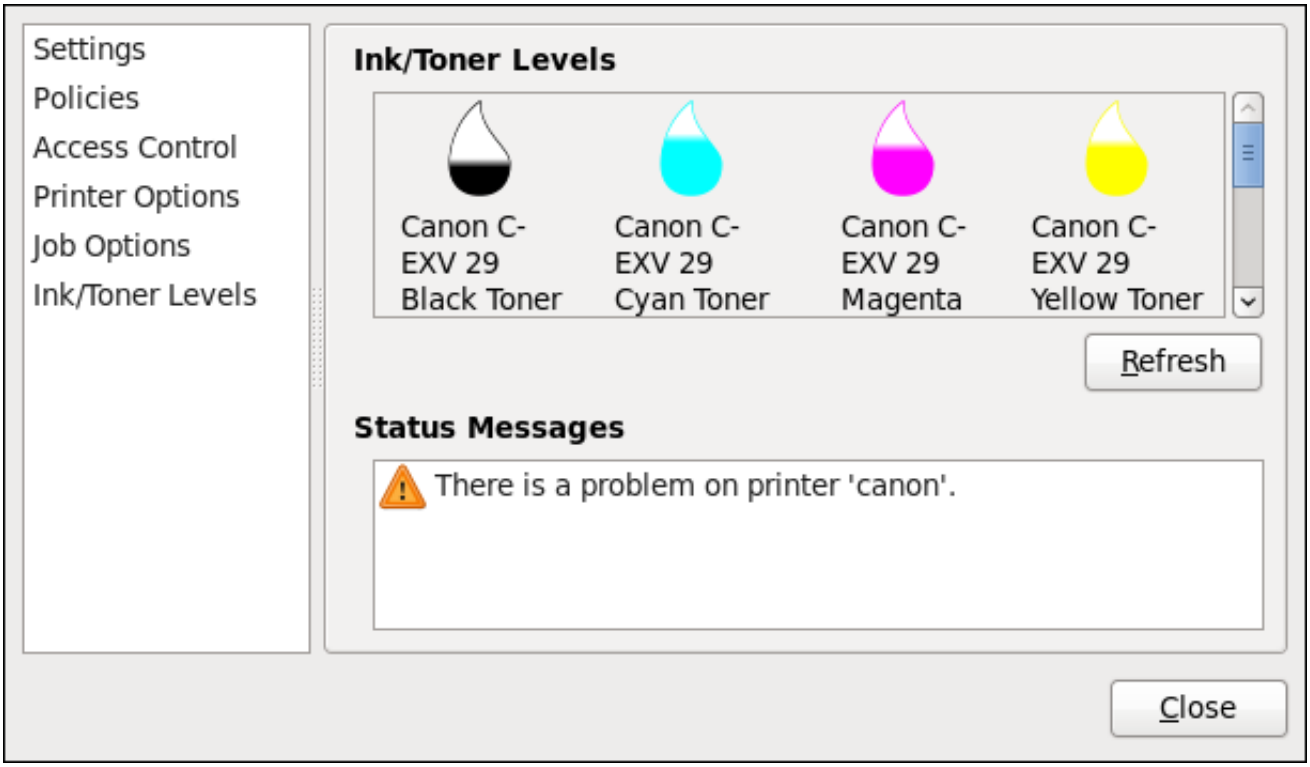


図18.17 インク/トナーのレベルのページ

18.3.10.3. 印刷ジョブの管理

Emacs からのテキストファイルの印刷または **GIMP** からの画像の印刷などプリンターデーモンに印刷ジョブを送ると、印刷ジョブは印刷のスプールキューに追加されます。印刷のスプールキューはプリンターに送られた印刷ジョブの一覧で、各印刷要求に関する情報、例えば印刷要求の状態、ジョブ番号などを表示します。

印刷が処理されている間、**プリンター状態**のアイコンがパネルの **通知スペース** に表示されます。その**プリンター状態**をクリックすると、図18.18「GNOME 印刷の状態」 に似たウィンドウが表示され、印刷ジョブの状態を確認できます。

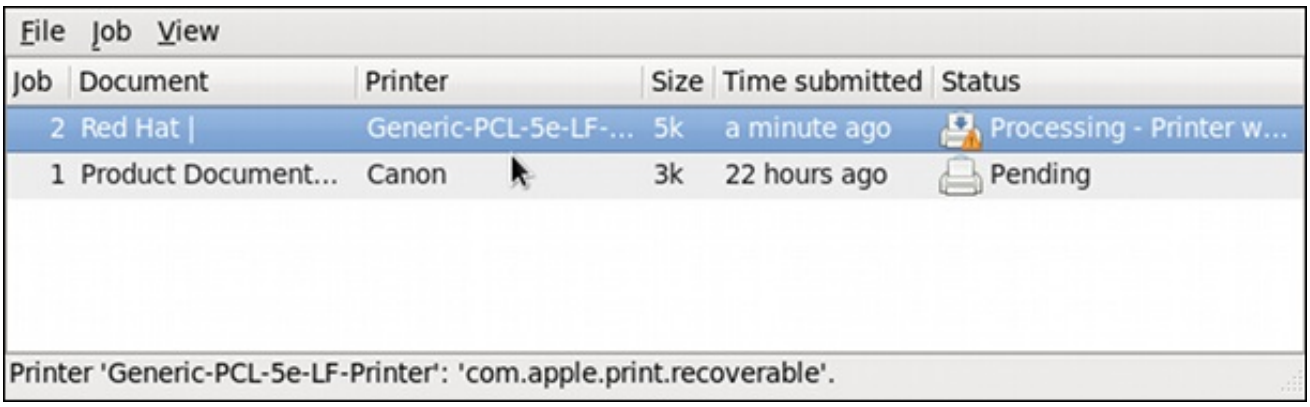


図18.18 GNOME 印刷の状態

印刷ジョブをキャンセル、保留、解除、再印刷、認証するためには、**GNOME 印刷の状態 (GNOME Print Status)** でジョブを選択し、**ジョブ (Job)** メニューでそれぞれコマンドをクリックします。

シェルプロンプトから印刷スプールの印刷ジョブの一覧を表示させるには、`lpstat -o` コマンドを入力します。最後の数行は以下のようになります。

例18.1 `lpstat -o` 出力の例

```
$ lpstat -o
Charlie-60              twaugh             1024   Tue 08 Feb 2011
16:42:11 GMT
Aaron-61                twaugh             1024   Tue 08 Feb 2011
16:42:44 GMT
Ben-62                  root               1024   Tue 08 Feb 2011
16:45:42 GMT
```

印刷ジョブをキャンセルしたい場合は、**`lpstat -o`** コマンドを使って依頼したジョブ番号を見つけます。その後 **`cancel`** ***ジョブ番号*** コマンドを使用します。例えば、**`cancel 60`** だと 例18.1「**`lpstat -o` 出力の例**」 の印刷ジョブを取り消します。他のユーザーによって開始された印刷ジョブを **`cancel`** コマンドでキャンセルすることはできません。ただし、**`cancel -U root`** ***`job_number`*** コマンドを使用すると、強制的にそうしたジョブを削除することは可能です。そうしたキャンセルを防ぐには、プリンターの操作ポリシーを **認証済** に変更することでルート認証を強要することができます。

シェルプロンプトから直接ファイルを印刷することもできます。例えば **`lp sample.txt`** コマンドはテキストファイル **`sample.txt`** を印刷します。印刷フィルターはファイルのタイプを決定し、プリンターが理解できる形式に変換します。

18.3.11. その他のリソース

Red Hat Enterprise Linux の印刷に関する詳細は、以下のリソースを参照して下さい。

18.3.11.1. インストールされているドキュメント

`man lp`

`lpr` コマンドのマニュアルページです。コマンドラインからファイルを印刷することができます。

`man cancel`

印刷キューから印刷ジョブを削除するためのコマンドラインユーティリティのマニュアルページです。

`man mpage`

1 枚の用紙に複数ページを印刷するためのコマンドラインユーザーのマニュアルページです。

`man cupsd`

CUPS プリンターデーモンのマニュアルページです。

`man cupsd.conf`

CUPS プリンターデーモンの設定ファイルのマニュアルページです。

`man classes.conf`

CUPS のクラス設定ファイルのマニュアルページです。

`man lpstat`

`lpstat` のマニュアルページで、クラス、ジョブ、プリンターなどの状態情報を表示します。

18.3.11.2. 役立つ Web サイト

http://www.linuxprinting.org/

『GNU/Linux Printing』 には Linux の印刷に関する豊富な情報が記載されています。

http://www.cups.org/

CUPS のドキュメンテーション、FAQ、ニュースグループについて記載しています。

パート VI. 監視と自動化

ここでは、システム管理者がシステムパフォーマンスのモニター、システムタスクの自動化、バグの報告を行うための各種ツールについて解説します。

第19章 システムモニタリングツール

システムを設定するには、多くの場合システム管理者は空きメモリの容量、空きディスク領域、ハードディスクのパーティション設定状況、実行中のプロセスを決定する必要があります。

19.1. システムプロセスの表示

19.1.1. ps コマンドの使用

ps コマンドは、実行中のプロセスについての情報を表示します。静的な一覧、すなわちコマンドを実行するときに実行しているプロセスのスナップショットです。実行中のプロセスを定期的に更新した一覧を表示させるには、**top** コマンドまたは **システムモニター** アプリケーションを代わりに使用します。

他のユーザーが所有しているプロセスを含め、現在システム上で実行中の全プロセスを一覧表示するには、シェルプロンプトで以下を入力します。

```
ps ax
```

一覧表示された各プロセスについて **ps ax** コマンドが表示するのは次のとおりです。プロセス ID (**PID**)、それに関連付けされたターミナル (**TTY**)、現在のステータス (**STAT**)、累積 CPU 時間 (**TIME**)、実行ファイル名 (**COMMAND**) です。例えば以下のとおりです。

```
~]$ ps ax
  PID TTY      STAT   TIME COMMAND
    1 ?        Ss     0:01 /sbin/init
    2 ?        S      0:00 [kthreadd]
    3 ?        S      0:00 [migration/0]
    4 ?        S      0:00 [ksoftirqd/0]
    5 ?        S      0:00 [migration/0]
    6 ?        S      0:00 [watchdog/0]
[出力は省略されています]
```

各プロセスと同時に所有者も表示するには、以下のコマンドを使用します。

```
ps aux
```

ps ax コマンドで提供される情報以外に、**ps aux** はプロセス所有者の有効なユーザー名 (**USER**)、CPU のパーセンテージ (**%CPU**) およびメモリ使用率 (**%MEM**)、キロバイト単位での仮想メモリサイズ (**VSZ**)、キロバイト単位での非スワップの物理メモリサイズ (**RSS**)、プロセスの開始日時を表示します。例えば以下のとおりです。

```
~]$ ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.1  19404   832 ?        Ss   Mar02   0:01
/sbin/init
root         2  0.0  0.0      0     0 ?        S    Mar02   0:00
[kthreadd]
root         3  0.0  0.0      0     0 ?        S    Mar02   0:00
[migration/0]
root         4  0.0  0.0      0     0 ?        S    Mar02   0:00
[ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S    Mar02   0:00
```

```
[migration/0]
root         6  0.0  0.0      0     0 ?        R    Mar02   0:00
[watchdog/0]
[出力は省略されています]
```

ps コマンドを **grep** と組み合わせて使用しても、特定のプロセスが実行中かどうかを確認することができます。例えば、**Emacs** が実行中かどうかを知るには、以下を入力します。

```
~]$ ps ax | grep emacs
12056 pts/3    S+     0:00 emacs
12060 pts/2    S+     0:00 grep --color=auto emacs
```

利用可能なコマンドラインオプションの全一覧は、**ps**(1) の man ページを参照してください。

19.1.2. top コマンドの使用

top コマンドは、システム上で実行中のプロセスの一覧をリアルタイムで表示します。また、システムのアップタイム、現在の CPU およびメモリ使用率、実行中のプロセスの合計数についての追加情報も表示します。さらには、一覧の並び替えやプロセスの kill などの操作も実行できます。

top コマンドを実行するには、シェルプロンプトで以下を入力します。

```
top
```

一覧表示された各プロセスについて **top** コマンドが表示するのは次のとおりです。プロセス ID (**PID**)、プロセス所有者の有効なユーザー名、(**USER**)、優先度 (**PR**)、nice 値 (**NI**)、プロセスが使用する仮想メモリ容量 (**VIRT**)、プロセスが使用する非スワップの物理メモリ容量 (**RES**)、プロセスが使用する共有メモリ容量 (**SHR**)、CPU のパーセンテージ (**%CPU**) およびメモリ使用率 (**%MEM**)、累積 CPU 時間 (**TIME+**)、実行ファイル名 (**COMMAND**) です。例えば以下のとおりです。

```
~]$ top
top - 02:19:11 up 4 days, 10:37,  5 users,  load average: 0.07, 0.13, 0.09
Tasks: 160 total,   1 running, 159 sleeping,   0 stopped,   0 zombie
Cpu(s): 10.7%us,  1.0%sy,  0.0%ni, 88.3%id,  0.0%wa,  0.0%hi,  0.0%si,
0.0%st
Mem:    760752k total,   644360k used,   116392k free,     3988k buffers
Swap:  1540088k total,    76648k used,  1463440k free,   196832k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
14401 jhradile  20   0  313m  10m 5732 S  5.6  1.4   6:27.29 gnome-system-
mo
 1764 root      20   0  133m  23m 4756 S  5.3  3.2   6:32.66 Xorg
13865 jhradile  20   0 1625m 177m 6628 S  0.7 23.8   0:57.26 java
   20 root      20   0     0    0    0 S  0.3  0.0   4:44.39 ata/0
 2085 root      20   0 40396  348  276 S  0.3  0.0   1:57.13 udisks-daemon
    1 root      20   0 19404  832  604 S  0.0  0.1   0:01.21 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.01 kthreadd
    3 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    4 root      20   0     0    0    0 S  0.0  0.0   0:00.02 ksoftirqd/0
    5 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    6 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 watchdog/0
    7 root      20   0     0    0    0 S  0.0  0.0   0:01.00 events/0
    8 root      20   0     0    0    0 S  0.0  0.0   0:00.00 cpuset
    9 root      20   0     0    0    0 S  0.0  0.0   0:00.00 khelper
```

```
   10 root      20   0     0    0    0 S  0.0  0.0   0:00.00 netns
   11 root      20   0     0    0    0 S  0.0  0.0   0:00.00 async/mgr
   12 root      20   0     0    0    0 S  0.0  0.0   0:00.00 pm
[出力は省略されています]
```

表19.1「インタラクティブな top コマンド」 には、**top** と使用できるインタラクティブな役立つコマンドが記載されています。詳細については、**top**(1) の man ページを参照してください。

表19.1 インタラクティブな top コマンド

コマンド	詳細
Enter、**Space**	表示を最新の情報に直ちに更新します。
h、**?**	ヘルプ画面を表示します。
k	プロセスを kill します。プロセス ID およびプロセスに送信するシグナルがプロンプトされます。
n	表示されるプロセス番号を変更します。番号を入力するようプロンプトされます。
u	一覧をユーザー別に並べ替えます。
M	一覧をメモリ使用率で並べ替えます。
P	一覧を CPU 使用率で並べ替えます。
q	ユーティリティを終了して、シェルプロンプトに戻ります。

19.1.3. システムモニターツールの使用

システムモニター ツールの **プロセス** タブを使用することで、グラフィカルユーザーインターフェースからプロセスの表示、検索、優先度の変更、kill を行うことができます。

システムモニター ツールを起動するには、パネルから **アプリケーション** → **システムツール** → **システムモニター** の順に選択するか、シェルプロンプトで **gnome-system-monitor** と入力します。その後、**プロセス** タブをクリックすると実行中のプロセスの一覧が表示されます。

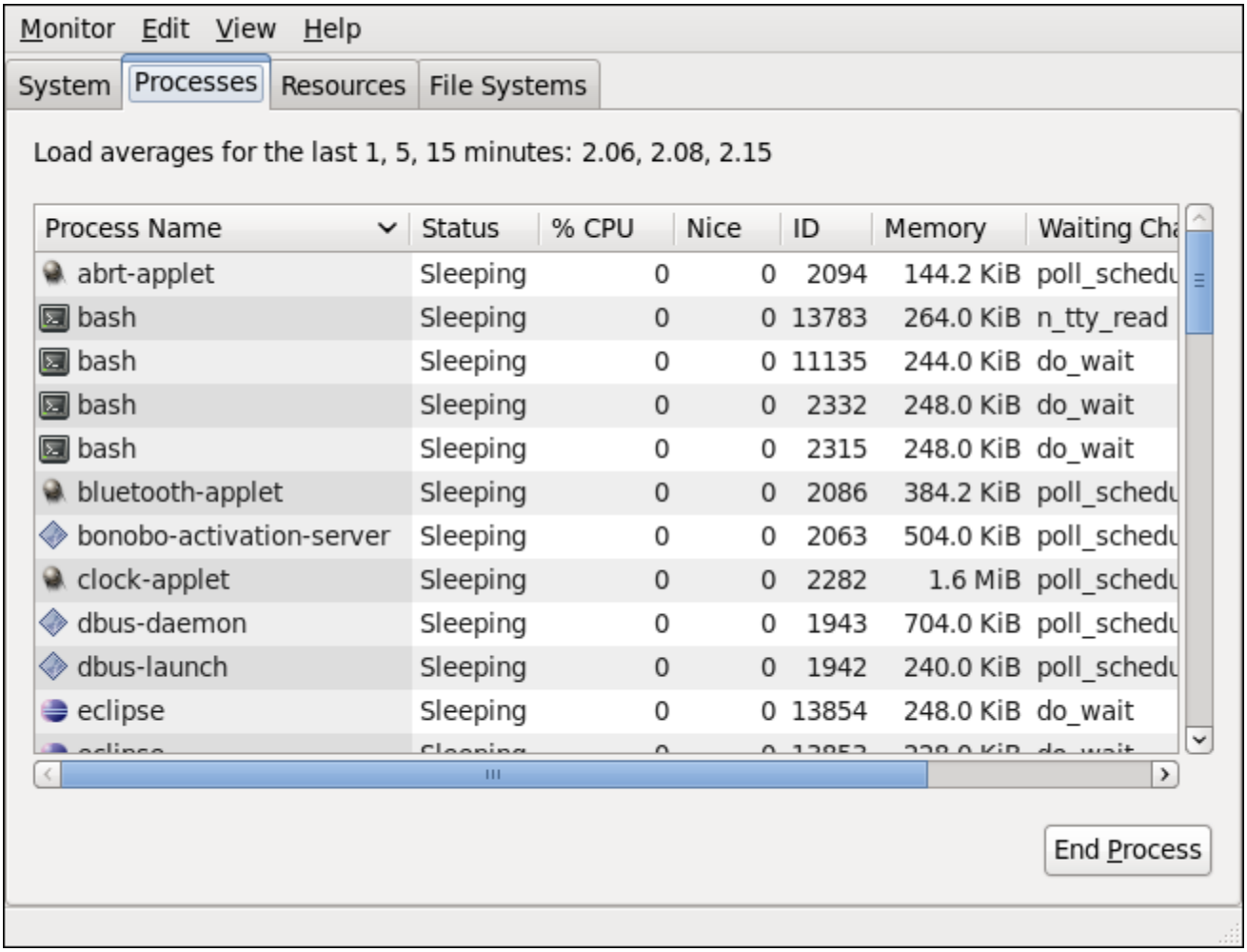


図19.1 システムモニター — プロセス

一覧表示された各プロセスについて **システムモニター** ツールが表示するのは次のとおりです。プロセス名 (**Process Name**)、現在の状態 (**Status**)、CPU 使用量のパーセンテージ (**% CPU**)、nice 値 (**Nice**)、プロセス ID (**ID**)、メモリ使用量 (**Memory**)、プロセスが待機しているチャンネル (**Waiting Channel**)、セッションについての補足情報 (**Session**) です。特定のカラム別に情報を昇順で並び替えるには、コラム名をクリックします。再度コラム名をクリックすると、昇順と降順が切り替わります。

デフォルトでは、**システムモニター** ツールは現在のユーザーが所有しているプロセスの一覧を表示します。**表示** メニューから各種オプションを選択すると、以下を実行できます。

- 実行中のプロセスのみの表示
- すべてのプロセスの表示
- ユーザーのプロセスの表示
- プロセスの依存関係の表示
- 選択したプロセスのメモリマップの表示
- 選択したプロセスで開いたファイルの表示
- プロセス一覧を最新の情報に更新

また、**編集** メニューの各種オプションを使用すると以下を実行できます。

- プロセスの停止

- プロセスの再開
- プロセスの終了
- プロセスの kill (強制終了)
- 選択したプロセスの優先度の変更
- **システムモニター** 設定 (プロセスの一覧を更新する間隔や表示する情報) の編集

プロセスを終了するには、一覧からプロセスを選択して、**プロセスの終了** ボタンをクリックしても可能です。

19.2. メモリ使用量の表示

19.2.1. free コマンドの使用

free コマンドは、システムのメモリの空き容量と使用量を表示します。シェルプロンプトで以下を入力してください。

```
free
```

free コマンドは、物理メモリ (**Mem**) およびスワップ領域 (**Swap**) に関する情報を提供します。メモリの合計量 (**total**) だけでなく、使用メモリ容量 (**used**)、空きメモリ容量 (**free**)、共有メモリ容量 (**shared**)、カーネルバッファに使用されるメモリ容量 (**buffers**)、キャッシュに使用されるメモリ容量 (**cached**) を表示します。例えば以下のとおりです。

```
~]$ free
             total       used       free     shared    buffers
cached
Mem:        760752     661332      99420          0       6476
317200
-/+ buffers/cache:     337656     423096
Swap:      1540088     283652    1256436
```

デフォルトでは、**free** は値をキロバイトで表示します。値をメガバイトで表示するには、**-m** コマンドラインオプションを指定してください。

```
free -m
```

例えば以下のようにします:

```
~]$ free -m
             total       used       free     shared    buffers
cached
Mem:           742        646         96          0          6
309
-/+ buffers/cache:        330        412
Swap:         1503        276       1227
```

利用可能なコマンドラインオプションの全一覧は、**free**(1) の man ページを参照してください。

19.2.2. システムモニターツールの使用

システムモニター ツールの **リソース** タブは、システムのメモリの空き容量と使用量を表示します。

システムモニター ツールを起動するには、パネルから **アプリケーション** → **システムツール** → **システムモニター** の順に選択するか、シェルプロンプトで **gnome-system-monitor** を入力します。その後、**リソース** タブをクリックするとシステムのメモリ使用率が表示されます。

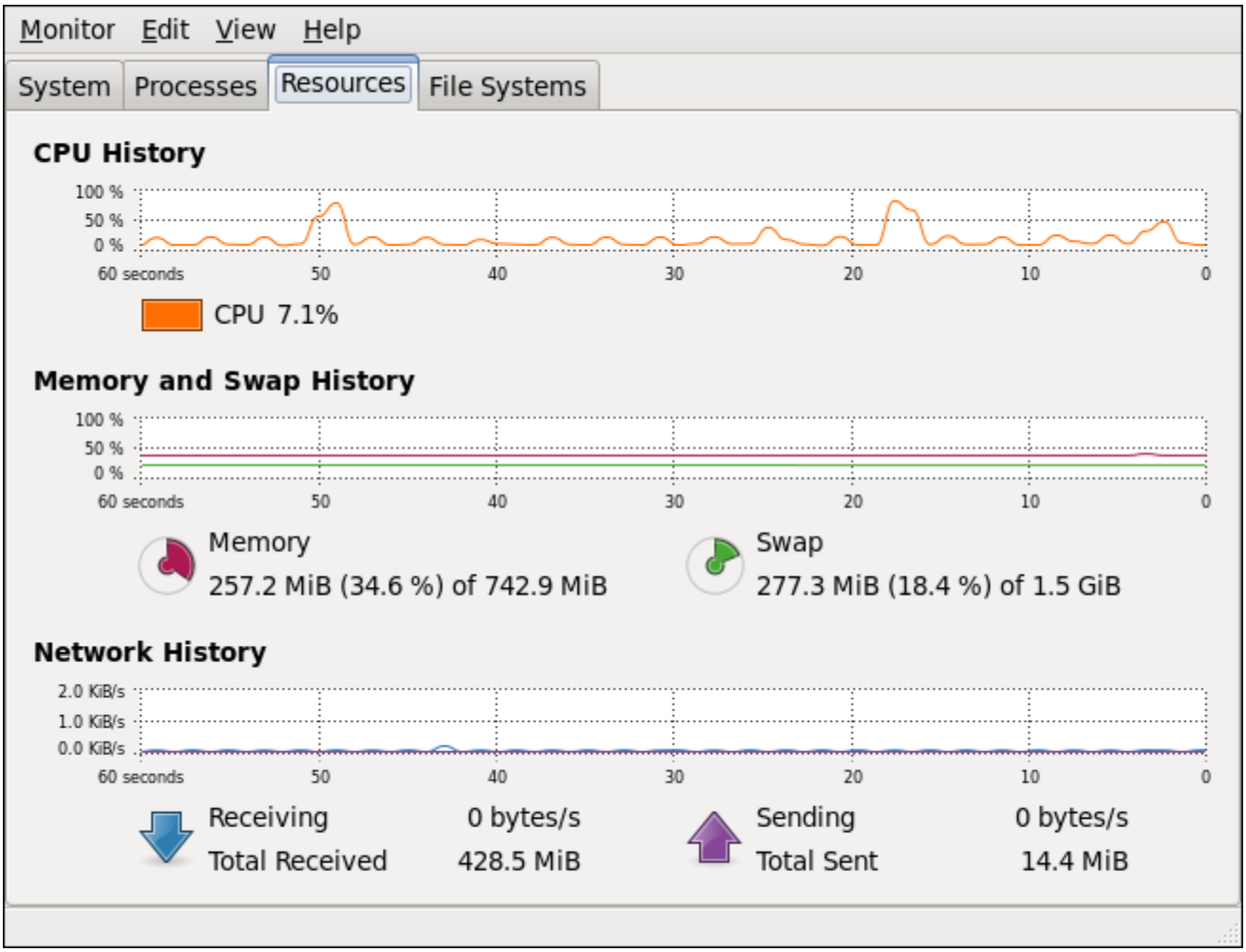


図19.2 システムモニター — リソース

システムモニター ツールは、**メモリとスワップの履歴** のセクションでメモリおよびスワップの使用履歴、さらには物理メモリの総容量 (**メモリ**)、スワップ領域 (**スワップ**)、使用量をグラフィカルに表示します。

19.3. CPU 使用率の表示

19.3.1. システムモニターツールの使用

システムモニター ツールの **リソース** タブは、システムの現在の CPU 使用率を表示します。

システムモニター ツールを起動するには、パネルから **アプリケーション** → **システムツール** → **システムモニター** の順に選択するか、シェルプロンプトで **gnome-system-monitor** を入力します。その後、**リソース** タブをクリックするとシステムの CPU 使用率が表示されます。

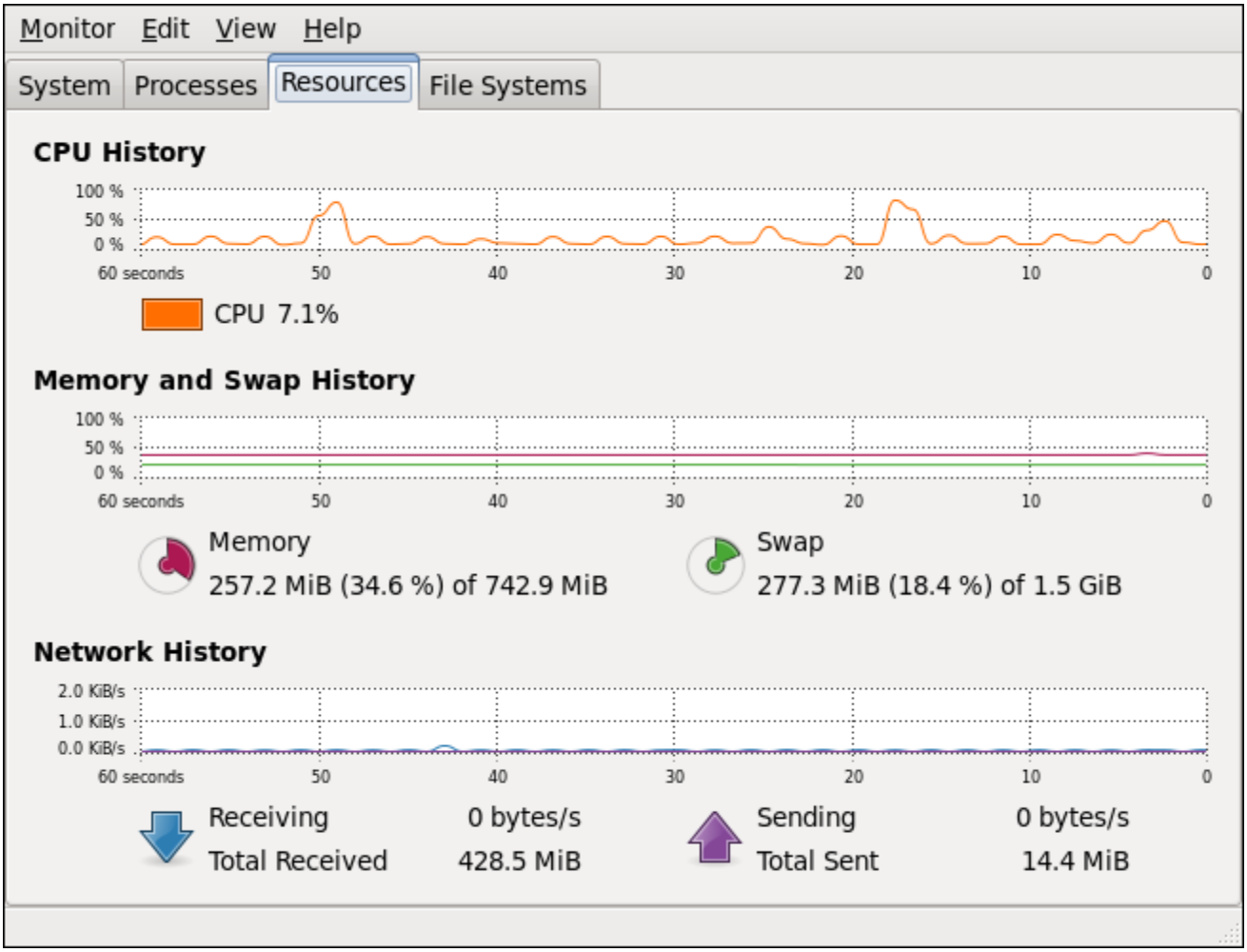


図19.3 システムモニター — リソース

システムモニター ツールの **CPU 使用率の履歴** セクションでは、CPU 使用率の履歴をグラフィカルに表示するとともに現在使用中の CPU をパーセンテージで表示します。

19.4. ブロックデバイスとファイルシステムの表示

19.4.1. lsblk コマンドの使用

lsblk コマンドは、利用可能なブロックデバイスの一覧を表示します。シェルプロンプトで以下を入力してください。

```
lsblk
```

一覧表示された各ブロックデバイスについて **lsblk** コマンドが表示するのは次のとおりです。デバイス名 (**NAME**)、メジャーおよびマイナーデバイス番号 (**MAJ:MIN**)、リムーバブルデバイスかどうか (**RM**)、そのサイズ (**SIZE**)、読み取り専用デバイスかどうか (**RO**)、そのタイプ (**TYPE**)、デバイスのマウント先 (**MOUNTPOINT**) です。例えば以下のとおりです。

```
~]$ lsblk
NAME                    MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sr0                      11:0    1  1024M  0 rom
vda                     252:0    0    20G  0 rom
|-vda1                  252:1    0   500M  0 part /boot
```

```
`-vda2                    252:2    0  19.5G  0 part
  |-vg_kvm-lv_root (dm-0) 253:0    0    18G  0 lvm  /
  `-vg_kvm-lv_swap (dm-1) 253:1    0   1.5G  0 lvm  [SWAP]
```

デフォルトでは、**lsblk** はツリーのような形式でブロックデバイスを一覧表示します。通常の一覧として表示するには、**-l** コマンドラインオプションを追加します。

```
lsblk -l
```

例えば以下のようにします:

```
~]$ lsblk -l
NAME                  MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sr0                    11:0    1  1024M  0 rom
vda                   252:0    0    20G  0 rom
vda1                  252:1    0   500M  0 part /boot
vda2                  252:2    0  19.5G  0 part
vg_kvm-lv_root (dm-0) 253:0    0    18G  0 lvm  /
vg_kvm-lv_swap (dm-1) 253:1    0   1.5G  0 lvm  [SWAP]
```

利用可能なコマンドラインオプションの全一覧は、**lsblk**(8) の man ページを参照してください。

19.4.2. blkid コマンドの使用

blkid コマンドは利用可能なブロックデバイスに関する情報を表示します。**root** としてシェルプロンプトで以下を入力してください。

```
blkid
```

一覧表示された各ブロックデバイスについて **blkid** コマンドは、*汎用一意識別子* (**UUID**)、ファイルシステムのタイプ (**TYPE**)、ボリュームラベル (**LABEL**) などの属性を表示します。例えば以下のとおりです。

```
~]# blkid
/dev/vda1: UUID="7fa9c421-0054-4555-b0ca-b470a97a3d84" TYPE="ext4"
/dev/vda2: UUID="7IvYzk-TnnK-oPjf-ipdD-cofz-DXaJ-gPdgBW"
TYPE="LVM2_member"
/dev/mapper/vg_kvm-lv_root: UUID="a07b967c-71a0-4925-ab02-aebcad2ae824"
TYPE="ext4"
/dev/mapper/vg_kvm-lv_swap: UUID="d7ef54ca-9c41-4de4-ac1b-4193b0c1ddb6"
TYPE="swap"
```

デフォルトでは、**lsblk** コマンドはすべての利用可能なブロックデバイスを一覧表示します。特定のデバイスの情報のみを表示するには、コマンドラインでデバイス名を指定します。

```
blkid device_name
```

例えば、**/dev/vda1** に関する情報を表示するには、以下を入力してください。

```
~]# blkid /dev/vda1
/dev/vda1: UUID="7fa9c421-0054-4555-b0ca-b470a97a3d84" TYPE="ext4"
```

上記のコマンドに **-p** および **-o udev** のコマンドラインオプションを使用して、さらに詳しい情報を取得することも可能です。このコマンドを実行するには、**root** 権限が必要な点に注意してください。

```
blkid -po udev device_name
```

例:

```
~]# blkid -po udev /dev/vda1
ID_FS_UUID=7fa9c421-0054-4555-b0ca-b470a97a3d84
ID_FS_UUID_ENC=7fa9c421-0054-4555-b0ca-b470a97a3d84
ID_FS_VERSION=1.0
ID_FS_TYPE=ext4
ID_FS_USAGE=filesystem
```

利用可能なコマンドラインオプションの全一覧は、**blkid**(8) の man ページを参照してください。

19.4.3. findmnt コマンドの使用

findmnt コマンドは、現在マウントされているファイルシステムの一覧を表示します。シェルプロンプトで以下を入力します。

```
findmnt
```

一覧表示された各ファイルシステムについて **findmnt** コマンドが表示するのは、次のとおりです。マウントポイント先 (**TARGET**)、ソースデバイス (**SOURCE**)、ファイルシステムのタイプ (**FSTYPE**)、関連するマウントオプション (**OPTIONS**) です。例えば以下のとおりです。

```
~]$ findmnt
TARGET                          SOURCE                      FSTYPE   OPTIONS
/                               /dev/mapper/vg_kvm-lv_root ext4
rw,relatime,sec
|-/proc                         /proc                       proc
rw,relatime
| |-/proc/bus/usb               /proc/bus/usb               usbfs
rw,relatime
| `-/proc/sys/fs/binfmt_misc                               binfmt_m
rw,relatime
|-/sys                          /sys                        sysfs
rw,relatime,sec
|-/selinux                                                  selinuxf
rw,relatime
|-/dev                          udev                        devtmpfs
rw,relatime,sec
| `-/dev                        udev                        devtmpfs
rw,relatime,sec
|   |-/dev/pts                  devpts                      devpts
rw,relatime,sec
|   `-/dev/shm                  tmpfs                       tmpfs
rw,relatime,sec
|-/boot                         /dev/vda1                   ext4
rw,relatime,sec
|-/var/lib/nfs/rpc_pipefs       sunrpc                      rpc_pipe
rw,relatime
|-/misc                         /etc/auto.misc              autofs
```

```
rw,relatime,fd=
`-/net                          -hosts                          autofs
rw,relatime,fd=
[出力は省略されています]
```

デフォルトでは、**findmnt** はツリーのような形式でファイルシステムを一覧表示します。通常の一覧として表示するには、**-l** コマンドラインオプションを追加します。

```
findmnt -l
```

例えば以下のようにします:

```
~]$ findmnt -l
TARGET                   SOURCE                      FSTYPE   OPTIONS
/proc                    /proc                       proc     rw,relatime
/sys                     /sys                        sysfs
rw,relatime,seclabe
/dev                     udev                        devtmpfs
rw,relatime,seclabe
/dev/pts                 devpts                      devpts
rw,relatime,seclabe
/dev/shm                 tmpfs                       tmpfs
rw,relatime,seclabe
/                        /dev/mapper/vg_kvm-lv_root ext4
rw,relatime,seclabe
/selinux                                             selinuxf rw,relatime
/dev                     udev                        devtmpfs
rw,relatime,seclabe
/proc/bus/usb            /proc/bus/usb               usbfs    rw,relatime
/boot                    /dev/vda1                   ext4
rw,relatime,seclabe
/proc/sys/fs/binfmt_misc                             binfmt_m rw,relatime
/var/lib/nfs/rpc_pipefs  sunrpc                      rpc_pipe rw,relatime
/misc                    /etc/auto.misc              autofs
rw,relatime,fd=7,pg
/net                     -hosts                      autofs
rw,relatime,fd=13,p
[出力は省略されています]
```

特定のタイプのファイルシステムのみを一覧表示するよう選択することも可能です。そのためには、**-t** コマンドラインオプション、その次にファイルシステムのタイプを追加します。

```
findmnt -t type
```

例えば、**ext4** ファイルシステムをすべて表示するには以下を入力します。

```
~]$ findmnt -t ext4
TARGET SOURCE                      FSTYPE OPTIONS
/      /dev/mapper/vg_kvm-lv_root ext4
rw,relatime,seclabel,barrier=1,data=ord
/boot  /dev/vda1                   ext4
rw,relatime,seclabel,barrier=1,data=ord
```

利用可能なコマンドラインオプションの全一覧は、**findmnt**(8) の man ページを参照してください。

19.4.4. df コマンドの使用

df コマンドは、システムのディスク領域の使用量についての詳しいレポートを表示します。シェルプロンプトで以下を入力してください。

```
df
```

一覧表示された各ファイルシステムについて **df** コマンドが表示するのは次のとおりです。ファイルシステム名 (**Filesystem**)、サイズ (**1K-blocks** または **Size**)、使用領域 (**Used**)、利用可能な領域 (**Available**)、使用領域のパーセンテージ (**Use%**)、ファイルシステムのマウント先 (**Mounted on**) です。例えば以下のとおりです。

```
~]$ df
Filesystem                   1K-blocks      Used Available Use% Mounted on
/dev/mapper/vg_kvm-lv_root  18618236   4357360  13315112  25% /
tmpfs                         380376       288    380088   1% /dev/shm
/dev/vda1                     495844     77029    393215  17% /boot
```

デフォルトでは、**df** コマンドは 1 キロバイトブロック単位でパーティションのサイズを、キロバイト単位で使用中および利用可能なディスク領域の容量を表示します。この情報をメガバイトとギガバイトで表示するには、**-h** コマンドラインオプションを指定することで、**df** が人間が読み取れる形式で値を表示します。

```
df -h
```

例えば以下のようにします:

```
~]$ df -h
Filesystem                   Size  Used Avail Use% Mounted on
/dev/mapper/vg_kvm-lv_root    18G  4.2G   13G  25% /
tmpfs                        372M  288K  372M   1% /dev/shm
/dev/vda1                    485M   76M  384M  17% /boot
```

利用可能なコマンドラインオプションの全一覧は、**df**(1) の man ページを参照してください。

19.4.5. du コマンドの使用

du コマンドはディレクトリ内のファイルが使用している領域を表示します。現在の作業ディレクトリ内のサブディレクトリ用のディスク使用量を表示するには、コマンドラインオプションなしで以下のコマンドを実行します。

```
du
```

例:

```
~]$ du
14972   ./Downloads
4       ./.gnome2
4       ./.mozilla/extensions
4       ./.mozilla/plugins
12      ./.mozilla
15004   .
```

デフォルトでは、**du** コマンドはキロバイト単位でディスク使用量を表示します。メガバイトおよびギガバイト単位で表示するには、**-h** コマンドラインオプションを指定することで、人間が読み取れる形式で値を表示することができます。

```
du -h
```

例えば以下のようにします:

```
~]$ du -h
15M     ./Downloads
4.0K    ./.gnome2
4.0K    ./.mozilla/extensions
4.0K    ./.mozilla/plugins
12K     ./.mozilla
15M     .
```

du コマンドは、一覧の最後で常に現在のディレクトリの合計量を表示します。この情報のみを表示するには、**-s** コマンドラインオプションを使用します。

```
du -sh
```

例:

```
~]$ du -sh
15M     .
```

利用可能なコマンドラインオプションの全一覧は、**du**(1) の man ページを参照してください。

19.4.6. システムモニターツールの使用

システムモニター ツールの **ファイルシステム** タブは、グラフィカルユーザーインターフェースでファイルシステムおよびディスク領域の使用量を表示します。

システムモニター ツールを起動するには、パネルから **アプリケーション** → **システムツール** → **システムモニター** の順に選択するか、シェルプロンプトで **gnome-system-monitor** と入力します。その後、**ファイルシステム** タブをクリックするとファイルシステムの一覧が表示されます。

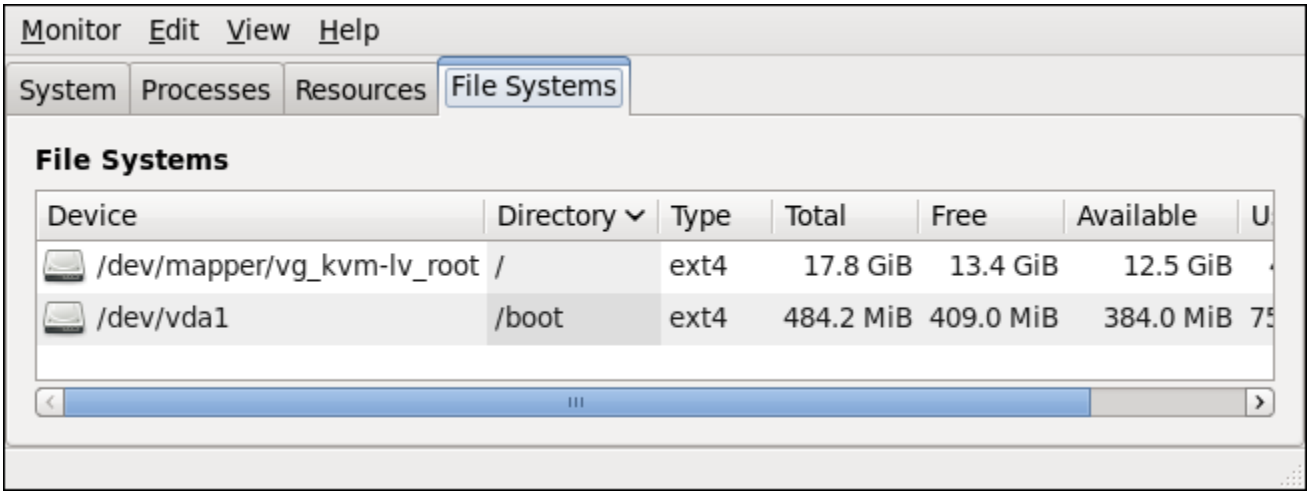


図19.4 システムモニター — ファイルシステム

一覧表示された各ファイルシステムについて **システムモニター** ツールが表示するのは次のとおりです。ソースデバイス (**Device**)、マウントポイント先 (**Directory**)、ファイルシステムのタイプ (**Type**)、さらにはそのサイズ (**Total**)、空き領域 (**Free**)、利用可能な領域 (**Available**)、使用領域 (**Used**) です。

19.5. ハードウェア情報の表示

19.5.1. lspci コマンドの使用

lspci コマンドは、PCI バスおよびそれらに接続されているデバイスの情報を表示します。システム内にあるすべての PCI デバイスを一覧表示するには、シェルプロンプトで以下を入力してください。

```
lspci
```

これは、以下のようにデバイスのシンプルな一覧を表示します。

```
~]$ lspci
00:00.0 Host bridge: Intel Corporation 82X38/X48 Express DRAM Controller
00:01.0 PCI bridge: Intel Corporation 82X38/X48 Express Host-Primary PCI
Express Bridge
00:1a.0 USB Controller: Intel Corporation 82801I (ICH9 Family) USB UHCI
Controller #4 (rev 02)
00:1a.1 USB Controller: Intel Corporation 82801I (ICH9 Family) USB UHCI
Controller #5 (rev 02)
00:1a.2 USB Controller: Intel Corporation 82801I (ICH9 Family) USB UHCI
Controller #6 (rev 02)
[出力は省略されています]
```

-v コマンドラインオプションを使用して、詳しい出力を表示することもできます。さらに詳細な出力を表示したい場合は、**-vv** を使用してください。

```
lspci -v|-vv
```

例えば、システムのビデオカードの製造元やモデル、メモリサイズを確認するには、以下を入力します。

```
~]$ lspci -v
[出力は省略されています]

01:00.0 VGA compatible controller: nVidia Corporation G84 [Quadro FX 370]
(rev a1) (prog-if 00 [VGA controller])
        Subsystem: nVidia Corporation Device 0491
        Physical Slot: 2
        Flags: bus master, fast devsel, latency 0, IRQ 16
        Memory at f2000000 (32-bit, non-prefetchable) [size=16M]
        Memory at e0000000 (64-bit, prefetchable) [size=256M]
        Memory at f0000000 (64-bit, non-prefetchable) [size=32M]
        I/O ports at 1100 [size=128]
        Expansion ROM at <unassigned> [disabled]
        Capabilities: <access denied>
        Kernel driver in use: nouveau
```

```
        Kernel modules: nouveau, nvidiafb

[出力は省略されています]
```

利用可能なコマンドラインオプションの全一覧は、**lspci**(8) の man ページを参照して下さい。

19.5.2. lsusb コマンドの使用

lsusb コマンドは、USB バスおよびそれらに接続されているデバイスの情報を表示します。システム内にあるすべての USB デバイスを一覧表示するには、シェルプロンプトで以下を入力してください。

```
lsusb
```

これは、以下のようにデバイスのシンプルな一覧を表示します。

```
~]$ lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
[出力は省略されています]
Bus 001 Device 002: ID 0bda:0151 Realtek Semiconductor Corp. Mass Storage
Device (Multicard Reader)
Bus 008 Device 002: ID 03f0:2c24 Hewlett-Packard Logitech M-UAL-96 Mouse
Bus 008 Device 003: ID 04b3:3025 IBM Corp.
```

-v コマンドラインオプションを使用すると、さらに詳細な出力を表示することもできます。

```
lsusb -v
```

例えば以下のようにします:

```
~]$ lsusb -v
[出力は省略されています]

Bus 008 Device 002: ID 03f0:2c24 Hewlett-Packard Logitech M-UAL-96 Mouse
Device Descriptor:
  bLength                18
  bDescriptorType         1
  bcdUSB               2.00
  bDeviceClass            0 (Defined at Interface level)
  bDeviceSubClass         0
  bDeviceProtocol         0
  bMaxPacketSize0         8
  idVendor           0x03f0 Hewlett-Packard
  idProduct          0x2c24 Logitech M-UAL-96 Mouse
  bcdDevice           31.00
  iManufacturer           1
  iProduct                2
  iSerial                 0
  bNumConfigurations      1
  Configuration Descriptor:
    bLength                 9
    bDescriptorType         2
[出力は省略されています]
```

利用可能なコマンドラインオプションの全一覧は、**lsusb**(8) の man ページを参照してください。

19.5.3. lspcmcia コマンドの使用

lspcmcia コマンドは、システム内にあるすべての PCMCIA デバイスを一覧表示します。シェルプロンプトで以下を入力してください。

```
lspcmcia
```

例:

```
~]$ lspcmcia
Socket 0 Bridge:        [yenta_cardbus]         (bus ID: 0000:15:00.0)
```

-v コマンドラインオプションを使用して、詳しい出力を表示することもできます。さらに詳細な出力を表示したい場合は、**-vv** を使用してください。

```
lspcmcia -v|-vv
```

例えば以下のようにします:

```
~]$ lspcmcia -v
Socket 0 Bridge:        [yenta_cardbus]         (bus ID: 0000:15:00.0)
        Configuration:  state: on       ready: unknown
```

利用可能なコマンドラインオプションの全一覧は、**pccardctl**(8) の man ページを参照してください。

19.5.4. lscpu コマンドの使用

lscpu コマンドは、システム内にある CPU に関する情報を一覧表示します。例えば、CPU 数、アーキテクチャー、ベンダー、ファミリ、モデル、CPU キャッシュなどです。シェルプロンプトで以下を入力してください。

```
lscpu
```

例:

```
~]$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                4
On-line CPU(s) list:   0-3
Thread(s) per core:    1
Core(s) per socket:    4
Socket(s):             1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 23
Stepping:              7
CPU MHz:               1998.000
BogoMIPS:              4999.98
```

```
Virtualization:        VT-x
L1d cache:             32K
L1i cache:             32K
L2 cache:              3072K
NUMA node0 CPU(s):     0-3
```

利用可能なコマンドラインオプションの全一覧は、**lscpu**(1) の man ページを参照してください。

19.6. NET-SNMP を使用したパフォーマンスのモニタリング

Red Hat Enterprise Linux 6 には、柔軟かつ拡張可能な SNMP (*Simple Network Management Protocol: シンプルネットワークマネージメントプロトコル*) エージェントを含む **Net-SNMP** ソフトウェアスイートが含まれています。このエージェントと関連ユーティリティを使用すると、多くのシステムからのパフォーマンスデータを SNMP プロトコルによるポーリングに対応する各種ツールに提供することができます。

本項では、ネットワーク上でパフォーマンスデータを安全に提供するための Net-SNMP エージェントの設定方法、SNMP プロトコルを使用したデータの取得方法、カスタムのパフォーマンスメトリックを提供するための SNMP エージェントの拡張方法について説明します。

19.6.1. Net-SNMP のインストール

Net-SNMP ソフトウェアスイートは、Red Hat Enterprise Linux ソフトウェアディストリビューションの RPM パッケージのセットとして利用可能です。表19.2「利用可能な Net-SNMP パッケージ」 は、各パッケージと内容を要約したものです。

表19.2 利用可能な Net-SNMP パッケージ

パッケージ	提供する項目
net-snmp	SNMP Agent Daemon とドキュメント。このパッケージは、パフォーマンスデータをエクスポートするために必要です。
net-snmp-libs	**netsnmp** ライブラリと同梱の管理情報ベース (MIB)。このパッケージは、パフォーマンスデータをエクスポートするために必要です。
net-snmp-utils	**snmpget** や **snmpwalk** などの SNMP クライアント。このパッケージは、SNMP によりシステムのパフォーマンスデータをクエリするために必要です。
net-snmp-perl	**mib2c** ユーティリティと **NetSNMP** Perl モジュール。
net-snmp-python	Python 用の SNMP クライアントライブラリ。

これらのパッケージをインストールするには、以下の形式で **yum** コマンドを使用します:

```
yum install package…
```

例えば、本項の残りで使用される SNMP Agent Daemon 及び SNMP クライアントをインストールするには、シェルプロンプトで以下を入力します:

```
~]# yum install net-snmp net-snmp-libs net-snmp-utils
```

このコマンドを実行するには、スーパーユーザー権限 (つまり **root** でログインしている) が必要である点に注意して下さい。Red Hat Enterprise Linux に新しいパッケージをインストールする方法の詳細については、「パッケージのインストール」 を参照して下さい。

19.6.2. Net-SNMP Daemon の実行

net-snmp パッケージには、SNMP Agent Daemon である **snmpd** が含まれています。本項では、**snmpd** サービスを起動/停止/再起動する方法、それを特定のランレベルで有効にする方法について説明します。ランレベルの概念や Red Hat Enterprise Linux における一般的なシステムサービスの管理方法の詳細については、10章 *サービスとデーモン* を参照して下さい。

19.6.2.1. サービスの開始

現行のセッションで **snmpd** サービスを実行するには、シェルプロンプトで **root** として以下を入力します:

```
service snmpd start
```

ブート時にサービスが自動的に起動するよう設定するには、以下のコマンドを使用します:

```
chkconfig snmpd on
```

これにより、ランレベル 2、3、4、5 でサービスが有効になります。別の方法として、「サービスの有効化と無効化」 で説明のとおり **サービスの設定** ユーティリティを使用することもできます。

19.6.2.2. サービスの停止

実行中の **snmpd** サービスを停止するには、シェルプロンプトで **root** として以下を入力します:

```
service snmpd stop
```

ブート時のサービスの起動を無効にするには、以下のコマンドを使用します:

```
chkconfig snmpd off
```

これにより、すべてのランレベルでサービスが無効になります。別の方法として、「サービスの有効化と無効化」 で説明のとおり **サービスの設定** ユーティリティを使用することもできます。

19.6.2.3. サービスの再開始

実行中の **snmpd** サービスを再起動するには、シェルプロンプトで以下を入力します:

```
service snmpd restart
```

これにより、サービスの停止と再起動が連続して行われます。サービスを停止せずに設定の再ロードだけを行いたい場合は、代わりに以下のコマンドを実行します:

```
service snmpd reload
```

これにより、実行中の **snmpd** サービスが設定を再ロードするようになります。

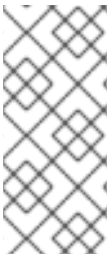
別の方法として、「サービスの開始、再起動と停止」 で説明のとおり **サービスの設定** ユーティリティを使用することもできます。

19.6.3. Net-SNMP の設定

Net-SNMP Agent Daemon の設定を変更するには、**/etc/snmp/snmpd.conf** 設定ファイルを編集します。Red Hat Enterprise Linux 6 に備わっているデフォルトの **snmpd.conf** ファイルには、多くのコメントが含まれているため、エージェントを設定するのに適切な開始ポイントとなります。

本項では、システム情報及び認証の設定という 2 つの一般的なタスクについて重点的に説明します。利用可能な設定指示文の詳細については、**snmpd.conf**(5) の man ページを参照して下さい。さらに、net-snmp パッケージには **snmpconf** と呼ばれるユーティリティがあり、有効なエージェント設定をインタラクティブに作成するために使用できます。

本項で説明されている **snmpwalk** ユーティリティを使用するには、net-snmp-utils パッケージがインストールされている必要がある点に注意して下さい。



注記

設定ファイルへの変更を反映させるには、**root** として以下のコマンドを実行することで、**snmpd** サービスが設定を再度読み取るように強制します:

```
service snmpd reload
```

19.6.3.1. システム情報の設定

Net-SNMP は、**system** ツリーにより基本的なシステム情報を提供します。例えば、次の **snmpwalk** コマンドはデフォルトのエージェント設定を持つ **system** ツリーを示しています:

```
~]# snmpwalk -v2c -c public localhost system
SNMPv2-MIB::sysDescr.0 = STRING: Linux localhost.localdomain 2.6.32-
122.el6.x86_64 #1 SMP Wed Mar 9 23:54:34 EST 2011 x86_64
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (99554) 0:16:35.54
SNMPv2-MIB::sysContact.0 = STRING: Root <root@localhost> (configure
/etc/snmp/snmp.local.conf)
SNMPv2-MIB::sysName.0 = STRING: localhost.localdomain
SNMPv2-MIB::sysLocation.0 = STRING: Unknown (edit /etc/snmp/snmpd.conf)
```

デフォルトでは、**sysName** オブジェクトはホスト名に設定されています。**sysLocation** 及び **sysContact** オブジェクトは、**syslocation** 及び **syscontact** 指示文の値を変更することで **/etc/snmp/snmpd.conf** ファイルで設定可能です。例えば以下のとおりです:

```
syslocation Datacenter, Row 3, Rack 2
syscontact UNIX Admin <admin@example.com>
```

設定ファイルに変更を加えた後は、再度 **snmpwalk** コマンドを実行することにより設定を再ロードしてテストします:

```
~]# service snmpd reload
Reloading snmpd: [  OK  ]
~]# snmpwalk -v2c -c public localhost system
SNMPv2-MIB::sysDescr.0 = STRING: Linux localhost.localdomain 2.6.32-
```

```
122.el6.x86_64 #1 SMP Wed Mar 9 23:54:34 EST 2011 x86_64
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (158357) 0:26:23.57
SNMPv2-MIB::sysContact.0 = STRING: UNIX Admin <admin@example.com>
SNMPv2-MIB::sysName.0 = STRING: localhost.localdomain
SNMPv2-MIB::sysLocation.0 = STRING: Datacenter, Row 3, Rack 2
```

19.6.3.2. 認証の設定

Net-SNMP Agent Daemon は SNMP プロトコルの 3 つの全バージョンに対応します。最初の 2 つのバージョン (1 と 2c) は、*コミュニティ文字列* を使用した簡易認証を提供します。この文字列は、エージェントとクライアントユーティリティ間で共有される秘密です。この文字列は、ネットワーク上でクリアテキストで渡されますが、安全性があるとは見なされません。SNMP プロトコルのバージョン 3 は、各種プロトコルを使用したユーザー認証とメッセージの暗号化に対応しています。また Net-SNMP エージェントは、SSH によるトンネリング、X.509 証明書を用いた TLS 認証、Kerberos 認証にも対応しています。

SNMP Version 2c Community の設定
SNMP version 2c community を設定するには、**/etc/snmp/snmpd.conf** 設定ファイルにある **rocommunity** または **rwcommunity** 指示文を使用します。指示文の形式は、以下のとおりです:

```
directive community [source [OID]]
```

community は使用するコミュニティ文字列、*source* は IP アドレスまたはサブネット、*OID* はアクセスを提供する SNMP ツリーです。例えば、次の指示文は、ローカルマシン上でコミュニティ文字列「redhat」 を使用するクライアントに **system** ツリーへの読み取り専用のアクセスを与えます:

```
rocommunity redhat 127.0.0.1 .1.3.6.1.2.1.1
```

設定をテストするには、**-v** と **-c** オプションを付けた **snmpwalk** コマンドを使用します:

```
~]# snmpwalk -v2c -c redhat localhost system
SNMPv2-MIB::sysDescr.0 = STRING: Linux localhost.localdomain 2.6.32-
122.el6.x86_64 #1 SMP Wed Mar 9 23:54:34 EST 2011 x86_64
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (158357) 0:26:23.57
SNMPv2-MIB::sysContact.0 = STRING: UNIX Admin <admin@example.com>
SNMPv2-MIB::sysName.0 = STRING: localhost.localdomain
SNMPv2-MIB::sysLocation.0 = STRING: Datacenter, Row 3, Rack 2
```

SNMP Version 3 User の設定
SNMP version 3 user を設定するには、**net-snmp-create-v3-user** コマンドを使用します。このコマンドは、**/var/lib/net-snmp/snmpd.conf** および **/etc/snmp/snmpd.conf** ファイルへエントリを追加し、ユーザーを作成してそのユーザーにアクセスを付与します。**net-snmp-create-v3-user** コマンドは、エージェントが実行中でない時にのみ実行可能な点に注意してください。以下の例では 「redhatsnmp」 というパスワードを持つ 「admin」 ユーザーを作成します。

```
~]# service snmpd stop
Stopping snmpd:                                            [  OK  ]
~]# net-snmp-create-v3-user
Enter a SNMPv3 user name to create:
admin
Enter authentication pass-phrase:
redhatsnmp
```

```
Enter encryption pass-phrase:
  [press return to reuse the authentication pass-phrase]

adding the following line to /var/lib/net-snmp/snmpd.conf:
   createUser admin MD5 "redhatsnmp" DES
adding the following line to /etc/snmp/snmpd.conf:
   rwuser admin
~]# service snmpd start
Starting snmpd:                                            [  OK  ]
```

net-snmp-create-v3-user が **/etc/snmp/snmpd.conf** に追加する **rwuser** 指示文 (または **-ro** のコマンドラインオプションが提供されている場合は **rouser**) の形式は、**rwcommunity** 及び **rocommunity** 指示文と似ています。

```
directive user [noauth|auth|priv] [OID]
```

user はユーザー名で、*OID* はアクセスを提供する SNMP ツリーです。デフォルトでは、Net-SNMP Agent Daemon は認証済み要求のみを許可します (**auth** オプション)。**noauth** オプションを使うと、認証されていない要求を許可することができ、**priv** オプションは暗号化の使用を強制します。**authpriv** オプションを使用すると、要求の認証、応答の暗号化が必要であると指定します。

例えば、以下の行では、ユーザー 「admin」 にツリー全体への読み取りと書き込みのアクセスを付与しています:

```
rwuser admin authpriv .1
```

設定をテストするには、使用しているユーザーのホームディレクトリ内に **.snmp** ディレクトリを作成して、そのディレクトリに次の行を含む **snmp.conf** と呼ばれる設定ファイルを作成します (**~/.snmp/snmp.conf**):

```
defVersion 3
defSecurityLevel authPriv
defSecurityName admin
defPassphrase redhatsnmp
```

これで、エージェントをクエリする場合、**snmpwalk** コマンドはこうした認証設定を使用するようになります:

```
~]$ snmpwalk -v3 localhost system
SNMPv2-MIB::sysDescr.0 = STRING: Linux localhost.localdomain 2.6.32-
122.el6.x86_64 #1 SMP Wed Mar 9 23:54:34 EST 2011 x86_64
[出力は省略されています]
```

19.6.4. SNMP によるパフォーマンスデータの取得

Red Hat Enterprise Linux の Net-SNMP Agent は、SNMP プロトコルによりパフォーマンスの各種情報を提供します。さらにエージェントは、システム上のインストールされた RPM パッケージの一覧、システム上で現在実行中のプロセス一覧、またはシステムのネットワーク設定をクエリすることもできます。

本項では、SNMP による利用可能なパフォーマンスチューニングに関連する OID の概要について説明します。本項では、net-snmp-utils パッケージがインストールされ、「認証の設定」 で説明のとおりユーザーは SNMP ツリーへのアクセスが許可されていることを前提としています。

19.6.4.1. ハードウェアの設定

Net-SNMP に含まれている **Host Resources MIB** は、ホストのハードウェア及びソフトウェアの現行設定に関する情報をクライアントユーティリティに表示します。表19.3「利用可能な OID」 には、その MIB で利用可能な様々な OID を要約した内容が記載されています。

表19.3 利用可能な OID

OID	詳細
HOST-RESOURCES-MIB::hrSystem	アップタイム、ユーザー数、実行中のプロセス数などのシステム情報全般が含まれています。
HOST-RESOURCES-MIB::hrStorage	メモリ及びファイルシステムの使用に関するデータが含まれています。
HOST-RESOURCES-MIB::hrDevices	すべてのプロセッサー、ネットワークデバイス、ファイルシステムの一覧が含まれています。
HOST-RESOURCES-MIB::hrSWRun	実行中の全プロセス一覧が含まれています。
HOST-RESOURCES-MIB::hrSWRunPerf	HOST-RESOURCES-MIB::hrSWRun からのプロセステーブル上のメモリと CPU 統計が含まれています。
HOST-RESOURCES-MIB::hrSWInstalled	RPM データベースの一覧が含まれています。

入手可能な情報のサマリを取得するために使用できる Host Resources MIB には、多くの SNMP テーブルがあります。次の例は、**HOST-RESOURCES-MIB::hrFSTable** を表示しています:

```
~]$ snmptable -Cb localhost HOST-RESOURCES-MIB::hrFSTable
SNMP table: HOST-RESOURCES-MIB::hrFSTable

 Index MountPoint RemoteMountPoint                                Type
    Access Bootable StorageIndex LastFullBackupDate LastPartialBackupDate
     1        "/"                "" HOST-RESOURCES-TYPES::hrFSLinuxExt2
 readWrite     true           31      0-1-1,0:0:0.0         0-1-1,0:0:0.0
     5 "/dev/shm"                ""     HOST-RESOURCES-TYPES::hrFSOther
 readWrite    false           35      0-1-1,0:0:0.0         0-1-1,0:0:0.0
     6    "/boot"                "" HOST-RESOURCES-TYPES::hrFSLinuxExt2
 readWrite    false           36      0-1-1,0:0:0.0         0-1-1,0:0:0.0
```

HOST-RESOURCES-MIB の詳細については、**/usr/share/snmp/mibs/HOST-RESOURCES-MIB.txt** ファイルを参照して下さい。

19.6.4.2. CPU とメモリ情報

大半のシステムのパフォーマンスデータは **UCD SNMP MIB** にあります。**systemStats** OID は、プロセッサー使用率に関する多くのカウンターを提供します:

```
~]$ snmpwalk localhost UCD-SNMP-MIB::systemStats
UCD-SNMP-MIB::ssIndex.0 = INTEGER: 1
```

```
UCD-SNMP-MIB::ssErrorName.0 = STRING: systemStats
UCD-SNMP-MIB::ssSwapIn.0 = INTEGER: 0 kB
UCD-SNMP-MIB::ssSwapOut.0 = INTEGER: 0 kB
UCD-SNMP-MIB::ssIOSent.0 = INTEGER: 0 blocks/s
UCD-SNMP-MIB::ssIOReceive.0 = INTEGER: 0 blocks/s
UCD-SNMP-MIB::ssSysInterrupts.0 = INTEGER: 29 interrupts/s
UCD-SNMP-MIB::ssSysContext.0 = INTEGER: 18 switches/s
UCD-SNMP-MIB::ssCpuUser.0 = INTEGER: 0
UCD-SNMP-MIB::ssCpuSystem.0 = INTEGER: 0
UCD-SNMP-MIB::ssCpuIdle.0 = INTEGER: 99
UCD-SNMP-MIB::ssCpuRawUser.0 = Counter32: 2278
UCD-SNMP-MIB::ssCpuRawNice.0 = Counter32: 1395
UCD-SNMP-MIB::ssCpuRawSystem.0 = Counter32: 6826
UCD-SNMP-MIB::ssCpuRawIdle.0 = Counter32: 3383736
UCD-SNMP-MIB::ssCpuRawWait.0 = Counter32: 7629
UCD-SNMP-MIB::ssCpuRawKernel.0 = Counter32: 0
UCD-SNMP-MIB::ssCpuRawInterrupt.0 = Counter32: 434
UCD-SNMP-MIB::ssIORawSent.0 = Counter32: 266770
UCD-SNMP-MIB::ssIORawReceived.0 = Counter32: 427302
UCD-SNMP-MIB::ssRawInterrupts.0 = Counter32: 743442
UCD-SNMP-MIB::ssRawContexts.0 = Counter32: 718557
UCD-SNMP-MIB::ssCpuRawSoftIRQ.0 = Counter32: 128
UCD-SNMP-MIB::ssRawSwapIn.0 = Counter32: 0
UCD-SNMP-MIB::ssRawSwapOut.0 = Counter32: 0
```

特に、**ssCpuRawUser**、**ssCpuRawSystem**、**ssCpuRawWait** 及び **ssCpuRawIdle** の OID は、システムがカーネル領域、ユーザー領域または I/O でプロセッサー時間の大半を費やしているかどうかを知る時に役立つカウンターを提供します。**ssRawSwapIn** と **ssRawSwapOut** は、システムがメモリ消費不足かどうかを知る際にも有用な場合もあります。

さらなるメモリ情報は、**free** コマンドと類似するデータを提供する **UCD-SNMP-MIB::memory** OID で入手できます:

```
~]$ snmpwalk localhost UCD-SNMP-MIB::memory
UCD-SNMP-MIB::memIndex.0 = INTEGER: 0
UCD-SNMP-MIB::memErrorName.0 = STRING: swap
UCD-SNMP-MIB::memTotalSwap.0 = INTEGER: 1023992 kB
UCD-SNMP-MIB::memAvailSwap.0 = INTEGER: 1023992 kB
UCD-SNMP-MIB::memTotalReal.0 = INTEGER: 1021588 kB
UCD-SNMP-MIB::memAvailReal.0 = INTEGER: 634260 kB
UCD-SNMP-MIB::memTotalFree.0 = INTEGER: 1658252 kB
UCD-SNMP-MIB::memMinimumSwap.0 = INTEGER: 16000 kB
UCD-SNMP-MIB::memBuffer.0 = INTEGER: 30760 kB
UCD-SNMP-MIB::memCached.0 = INTEGER: 216200 kB
UCD-SNMP-MIB::memSwapError.0 = INTEGER: noError(0)
UCD-SNMP-MIB::memSwapErrorMsg.0 = STRING:
```

負荷平均は、**UCD SNMP MIB** でも入手可能です。SNMP テーブル **UCD-SNMP-MIB::laTable** には、1 分、5 分、15 分の負荷平均を示した一覧があります:

```
~]$ snmptable localhost UCD-SNMP-MIB::laTable
SNMP table: UCD-SNMP-MIB::laTable

 laIndex laNames laLoad laConfig laLoadInt laLoadFloat laErrorFlag
laErrMessage
```

```
        1  Load-1   0.00    12.00          0    0.000000     noError
        2  Load-5   0.00    12.00          0    0.000000     noError
        3 Load-15   0.00    12.00          0    0.000000     noError
```

19.6.4.3. ファイルシステムとディスク情報

Host Resources MIB は、ファイルシステムのサイズと使用量についての情報を表示します。**HOST-RESOURCES-MIB::hrStorageTable** テーブルには、各ファイルシステム (及び各メモリプール) のエントリがあります:

```
~]$ snmptable -Cb localhost HOST-RESOURCES-MIB::hrStorageTable
SNMP table: HOST-RESOURCES-MIB::hrStorageTable

 Index                                          Type            Descr
AllocationUnits    Size   Used AllocationFailures
     1           HOST-RESOURCES-TYPES::hrStorageRam Physical memory
1024 Bytes 1021588 388064                   ?
     3 HOST-RESOURCES-TYPES::hrStorageVirtualMemory  Virtual memory
1024 Bytes 2045580 388064                   ?
     6         HOST-RESOURCES-TYPES::hrStorageOther  Memory buffers
1024 Bytes 1021588  31048                   ?
     7         HOST-RESOURCES-TYPES::hrStorageOther   Cached memory
1024 Bytes  216604 216604                   ?
    10 HOST-RESOURCES-TYPES::hrStorageVirtualMemory      Swap space
1024 Bytes 1023992      0                   ?
    31     HOST-RESOURCES-TYPES::hrStorageFixedDisk               /
4096 Bytes 2277614 250391                   ?
    35     HOST-RESOURCES-TYPES::hrStorageFixedDisk        /dev/shm
4096 Bytes  127698      0                   ?
    36     HOST-RESOURCES-TYPES::hrStorageFixedDisk           /boot
1024 Bytes  198337  26694                   ?
```

HOST-RESOURCES-MIB::hrStorageSize 及び **HOST-RESOURCES-MIB::hrStorageUsed** の OID を使用すると、それぞれマウントされたファイルシステムの残容量を算出することができます。

I/O データは **UCD-SNMP-MIB::systemStats** (**ssIORawSent.0** と **ssIORawRecieved.0**) 及び **UCD-DISKIO-MIB::diskIOTable** にあります。後者は、前者と比べてより詳細なデータを提供します。このテーブルには、**diskIONReadX** 及び **diskIONWrittenX** の OID があり、システムブートから問題のブロックデバイスに対し読み取り及び書き込みを実行したバイト数のカウンターを提供します:

```
~]$ snmptable -Cb localhost UCD-DISKIO-MIB::diskIOTable
SNMP table: UCD-DISKIO-MIB::diskIOTable

 Index Device     NRead  NWritten Reads Writes LA1 LA5 LA15    NReadX
NWrittenX
...
    25    sda 216886272 139109376 16409   4894   ?   ?    ? 216886272
139109376
    26   sda1   2455552      5120   613      2   ?   ?    ?   2455552
5120
    27   sda2   1486848         0   332      0   ?   ?    ?   1486848
0
    28   sda3 212321280 139104256 15312   4871   ?   ?    ? 212321280
139104256
```

19.6.4.4. ネットワーク情報

Interfaces MIB はネットワークデバイスの情報を提供します。**IF-MIB::ifTable** は、SNMP テーブルにシステム上の各インターフェースのエントリ、インターフェースの設定、インターフェース用の各種パケットカウンターを提供します。以下の例は、2 つの物理ネットワークインターフェースを持つシステム上の **ifTable** の最初の数カラムを示しています。

```
~]$ snmptable -Cb localhost IF-MIB::ifTable
SNMP table: IF-MIB::ifTable

 Index Descr              Type    Mtu    Speed      PhysAddress AdminStatus
     1    lo softwareLoopback 16436 10000000
up
     2  eth0   ethernetCsmacd  1500        0 52:54:0:c7:69:58          up
     3  eth1   ethernetCsmacd  1500        0 52:54:0:a7:a3:24        down
```

ネットワークトラフィックは、**IF-MIB::ifOutOctets** 及び **IF-MIB::ifInOctets** の OID にあります。以下の SNMP クエリは、このシステム上の各インターフェースに対するネットワークトラフィックを取得します:

```
~]$ snmpwalk localhost IF-MIB::ifDescr
IF-MIB::ifDescr.1 = STRING: lo
IF-MIB::ifDescr.2 = STRING: eth0
IF-MIB::ifDescr.3 = STRING: eth1
~]$ snmpwalk localhost IF-MIB::ifOutOctets
IF-MIB::ifOutOctets.1 = Counter32: 10060699
IF-MIB::ifOutOctets.2 = Counter32: 650
IF-MIB::ifOutOctets.3 = Counter32: 0
~]$ snmpwalk localhost IF-MIB::ifInOctets
IF-MIB::ifInOctets.1 = Counter32: 10060699
IF-MIB::ifInOctets.2 = Counter32: 78650
IF-MIB::ifInOctets.3 = Counter32: 0
```

19.6.5. Net-SNMP の拡張

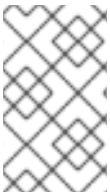
Net-SNMP Agent は、raw システムメトリックに加えてアプリケーションメトリックを提供するために拡張することができます。これにより、パフォーマンス問題のトラブルシューティングだけでなく容量計画も行うことができます。例えば、試験中に電子メールシステムの 5 分の負荷平均が 15 であったことを把握しておくことは役に立つかもしれませんが、毎秒 80,000 メッセージの処理中に電子メールシステムの負荷平均が 15 であることを知っておく方がはるかに役立ちます。アプリケーションメトリックがシステムメトリックと同じインターフェースで使用可能な場合、システムパフォーマンスの様々な負荷状況の影響も視覚化することができます (例えば 10,000 メッセージが追加されると、負荷平均は 100,000 まで直線的に増加します)。

Red Hat Enterprise Linux に同梱されているアプリケーションの多くは、Net-SNMP Agent を拡張して、SNMP によるアプリケーションメトリックを提供します。カスタムアプリケーション用にエージェントを拡張する方法もいくつかあります。本項では、シェルスクリプトと Perl プラグインを使ったエージェントの拡張方法について説明します。本項では、net-snmp-utils 及び net-snmp-perl パッケージがインストールされ、「認証の設定」 で説明のとおりユーザーは SNMP ツリーへのアクセスが許可されていることを前提としています。

19.6.5.1. シェルスクリプトによる Net-SNMP の拡張

Net-SNMP Agent は、任意のシェルスクリプトをクエリするために使用可能な拡張 MIB (**NET-SNMP-**

EXTEND-MIB) を提供します。実行するシェルスクリプトを指定するには、**/etc/snmp/snmpd.conf** ファイルの **extend** 指示文を使用します。定義されると、Agent は SNMP により終了コードとコマンドの出力を提供します。以下の例は、プロセステーブルの **httpd** プロセスの数を決定するスクリプトを使ってこの仕組みを説明しています。



注記

Net-SNMP Agent には、**proc** 指示文によりプロセステーブルを確認する組み込みメカニズムも備わっています。詳細については **snmpd.conf**(5) の man ページを参照して下さい。

以下のシェルスクリプトの終了コードは、任意の時点におけるシステム上で実行中の **httpd** プロセスの数です:

```
#!/bin/sh

NUMPIDS=`pgrep httpd | wc -l`

exit $NUMPIDS
```

SNMP でこのスクリプトを利用可能にするには、システムパス上にある場所にスクリプトをコピー後、実行ビットを設定して、**extend** 指示文を **/etc/snmp/snmpd.conf** ファイルに追加します。**extend** 指示文の形式は、以下のとおりです:

```
extend name prog args
```

name は拡張するための識別文字列、*prog* は実行するプログラム、*args* はプログラムに渡す引数です。例として、上記のシェルスクリプトが **/usr/local/bin/check_apache.sh** にコピーされた場合、以下の指示文は SNMP ツリーにスクリプトを追加します:

```
extend httpd_pids /bin/sh /usr/local/bin/check_apache.sh
```

スクリプトは **NET-SNMP-EXTEND-MIB::nsExtendObjects** でクエリできます:

```
~]$ snmpwalk localhost NET-SNMP-EXTEND-MIB::nsExtendObjects
NET-SNMP-EXTEND-MIB::nsExtendNumEntries.0 = INTEGER: 1
NET-SNMP-EXTEND-MIB::nsExtendCommand."httpd_pids" = STRING: /bin/sh
NET-SNMP-EXTEND-MIB::nsExtendArgs."httpd_pids" = STRING:
/usr/local/bin/check_apache.sh
NET-SNMP-EXTEND-MIB::nsExtendInput."httpd_pids" = STRING:
NET-SNMP-EXTEND-MIB::nsExtendCacheTime."httpd_pids" = INTEGER: 5
NET-SNMP-EXTEND-MIB::nsExtendExecType."httpd_pids" = INTEGER: exec(1)
NET-SNMP-EXTEND-MIB::nsExtendRunType."httpd_pids" = INTEGER: run-on-
read(1)
NET-SNMP-EXTEND-MIB::nsExtendStorage."httpd_pids" = INTEGER: permanent(4)
NET-SNMP-EXTEND-MIB::nsExtendStatus."httpd_pids" = INTEGER: active(1)
NET-SNMP-EXTEND-MIB::nsExtendOutput1Line."httpd_pids" = STRING:
NET-SNMP-EXTEND-MIB::nsExtendOutputFull."httpd_pids" = STRING:
NET-SNMP-EXTEND-MIB::nsExtendOutNumLines."httpd_pids" = INTEGER: 1
NET-SNMP-EXTEND-MIB::nsExtendResult."httpd_pids" = INTEGER: 8
NET-SNMP-EXTEND-MIB::nsExtendOutLine."httpd_pids".1 = STRING:
```

注意する点は、終了コード (この例では 「8」) は INTEGER (整数) タイプとして、出力は STRING (文

字列) タイプとして、示されていることです。複数のメトリックを整数として表示するには、**extend** 指示文を使用してスクリプトに異なる引数を指定します。例えば、以下のシェルスクリプトを使用すると、任意の文字列に一致するプロセスの数を見つけ出すことができ、プロセスの数を示すテキスト文字列も出力します:

```
#!/bin/sh

PATTERN=$1
NUMPIDS=`pgrep $PATTERN | wc -l`

echo "There are $NUMPIDS $PATTERN processes."
exit $NUMPIDS
```

次の **/etc/snmp/snmpd.conf** 指示文は、上記のスクリプトが **/usr/local/bin/check_proc.sh** にコピーされる時に **httpd** PID の数と併せて **snmpd** PID の数を与えます:

```
extend httpd_pids /bin/sh /usr/local/bin/check_proc.sh httpd
extend snmpd_pids /bin/sh /usr/local/bin/check_proc.sh snmpd
```

以下の例は、**nsExtendObjects** OID の **snmpwalk** の出力を示しています:

```
~]$ snmpwalk localhost NET-SNMP-EXTEND-MIB::nsExtendObjects
NET-SNMP-EXTEND-MIB::nsExtendNumEntries.0 = INTEGER: 2
NET-SNMP-EXTEND-MIB::nsExtendCommand."httpd_pids" = STRING: /bin/sh
NET-SNMP-EXTEND-MIB::nsExtendCommand."snmpd_pids" = STRING: /bin/sh
NET-SNMP-EXTEND-MIB::nsExtendArgs."httpd_pids" = STRING:
/usr/local/bin/check_proc.sh httpd
NET-SNMP-EXTEND-MIB::nsExtendArgs."snmpd_pids" = STRING:
/usr/local/bin/check_proc.sh snmpd
NET-SNMP-EXTEND-MIB::nsExtendInput."httpd_pids" = STRING:
NET-SNMP-EXTEND-MIB::nsExtendInput."snmpd_pids" = STRING:
...
NET-SNMP-EXTEND-MIB::nsExtendResult."httpd_pids" = INTEGER: 8
NET-SNMP-EXTEND-MIB::nsExtendResult."snmpd_pids" = INTEGER: 1
NET-SNMP-EXTEND-MIB::nsExtendOutLine."httpd_pids".1 = STRING: There are 8
httpd processes.
NET-SNMP-EXTEND-MIB::nsExtendOutLine."snmpd_pids".1 = STRING: There are 1
snmpd processes.
```

警告

整数の終了コードは 0 から 255 の範囲に制限されています。256 を超える可能性がある値については、スクリプトの標準出力 (文字列として入力されるもの) を使用するか、エージェントを拡張するという別の方法を実行して下さい。

この最後の例では、システムの空きメモリと **httpd** プロセスの数のクエリを示しています。このクエリは、パフォーマンステスト中にメモリ負担に与えるプロセス数の影響を知るために使用することができます:

```
~]$ snmpget localhost \
    'NET-SNMP-EXTEND-MIB::nsExtendResult."httpd_pids"' \
    UCD-SNMP-MIB::memAvailReal.0
NET-SNMP-EXTEND-MIB::nsExtendResult."httpd_pids" = INTEGER: 8
UCD-SNMP-MIB::memAvailReal.0 = INTEGER: 799664 kB
```

19.6.5.2. Perl による Net-SNMP の拡張

extend 指示文を使用してシェルスクリプトを実行することは、SNMP によるカスタムのアプリケーションメトリックを表示するにはかなり限られた方法です。Net-SNMP Agent は、カスタムオブジェクトを表示するための埋め込み Perl インターフェースも提供します。net-snmp-perl パッケージには、Red Hat Enterprise Linux に埋め込み Perl プラグインを書き込むために使用される **NetSNMP::agent** Perl モジュールがあります。

NetSNMP::agent Perl モジュールは、エージェントの OID ツリーの一部に対する要求を処理するために使用される **agent** オブジェクトを提供します。**agent** オブジェクトのコンストラクターには、エージェントを **snmpd** のサブエージェントまたはスタンドアロンエージェントとして実行するためのオプションがあります。埋め込みエージェントを作成するために必要な引数はありません:

```
use NetSNMP::agent (':all');

my $agent = new NetSNMP::agent();
```

agent オブジェクトには、コールバック関数を特定の OID に登録するために使用される **register** メソッドがあります。**register** 関数は、名前、OID、コールバック関数へのポインターを取ります。以下の例は、**hello_handler** と呼ばれるコールバック関数を OID **.1.3.6.1.4.1.8072.9999.9999** で要求を処理する SNMP Agent に登録しています:

```
$agent->register("hello_world", ".1.3.6.1.4.1.8072.9999.9999",
                 \&hello_handler);
```

注記

通常、OID **.1.3.6.1.4.1.8072.9999.9999** (**NET-SNMP-MIB::netSnmpPlaypen**) は表示目的のみに使用されます。お客様の組織に root OID がない場合は、ISO Name Registration Authority (米国では ANSI) にご連絡いただくと取得できます。

ハンドラー関数は、***HANDLER***、***REGISTRATION_INFO***、***REQUEST_INFO***、***REQUESTS*** の 4 つのパラメーターで呼び出されます。***REQUESTS*** パラメーターには、現在の呼び出しの要求一覧が含まれており、反復されデータが追加されるはずです。一覧の **request** オブジェクトには get 及び set メソッドがあるため、要求の OID と value を操作することができます。例として、以下の呼び出しは 「hello world」 という文字列に要求オブジェクトの値を設定します:

```
$request->setValue(ASN_OCTET_STR, "hello world");
```

ハンドラー関数は、GET 要求と GETNEXT 要求という 2 種類の SNMP 要求に応答できます。要求のタイプは、ハンドラー関数に第 3 のパラメーターとして渡される **request_info** オブジェクトの **getMode** メソッドを呼び出すことによって、決定されます。要求が GET 要求である場合、呼び出し元は、ハンドラーに要求の OID に応じて **request** オブジェクトの value を設定するよう求めます。要求が GETNEXT 要求である場合、呼び出し元は、ハンドラーに要求の OID をツリー内で次に利用可能な OID に設定するよう求めます。以下のコードは、この例を示しています:

```
my $request;
my $string_value = "hello world";
my $integer_value = "8675309";

for($request = $requests; $request; $request = $request->next()) {
  my $oid = $request->getOID();
  if ($request_info->getMode() == MODE_GET) {
    if ($oid == new NetSNMP::OID(".1.3.6.1.4.1.8072.9999.9999.1.0")) {
      $request->setValue(ASN_OCTET_STR, $string_value);
    }
    elsif ($oid == new NetSNMP::OID(".1.3.6.1.4.1.8072.9999.9999.1.1")) {
      $request->setValue(ASN_INTEGER, $integer_value);
    }
  } elsif ($request_info->getMode() == MODE_GETNEXT) {
    if ($oid == new NetSNMP::OID(".1.3.6.1.4.1.8072.9999.9999.1.0")) {
      $request->setOID(".1.3.6.1.4.1.8072.9999.9999.1.1");
      $request->setValue(ASN_INTEGER, $integer_value);
    }
    elsif ($oid < new NetSNMP::OID(".1.3.6.1.4.1.8072.9999.9999.1.0")) {
      $request->setOID(".1.3.6.1.4.1.8072.9999.9999.1.0");
      $request->setValue(ASN_OCTET_STR, $string_value);
    }
  }
}
```

getMode が **MODE_GET** を返す場合、ハンドラーは **request** オブジェクトの **getOID** 呼び出しの値を分析します。**request** の value は、OID が 「.1.0」 で終わる場合は **string_value** に、OID が 「.1.1」 で終わる場合は **integer_value** に設定されます。**getMode** が **MODE_GETNEXT** を返す場合、ハンドラーは要求の OID が 「.1.0」 かどうかを見つけ出し、その後 「.1.1」 に対する OID と値を設定します。ツリーで要求が 「.1.0」 より高い場合は、「.1.0」 に対する OID と値が設定されます。実際、これはツリーの 「次の」 値を返すため、**snmpwalk** のようなプログラムは構造に関する事前知識なくツリーをトラバースできます。

変数のタイプは **NetSNMP::ASN** からの定数を使用して設定されます。利用可能な定数の全一覧については、**NetSNMP::ASN** の **perldoc** を参照して下さい。

この例の Perl プラグインのコード全一覧は、以下のとおりです:

```
#!/usr/bin/perl

use NetSNMP::agent (':all');
use NetSNMP::ASN qw(ASN_OCTET_STR ASN_INTEGER);

sub hello_handler {
  my ($handler, $registration_info, $request_info, $requests) = @_;
  my $request;
  my $string_value = "hello world";
  my $integer_value = "8675309";

  for($request = $requests; $request; $request = $request->next()) {
    my $oid = $request->getOID();
    if ($request_info->getMode() == MODE_GET) {
      if ($oid == new NetSNMP::OID(".1.3.6.1.4.1.8072.9999.9999.1.0")) {
        $request->setValue(ASN_OCTET_STR, $string_value);
      }
```

```
      elsif ($oid == new NetSNMP::OID(".1.3.6.1.4.1.8072.9999.9999.1.1"))
{
        $request->setValue(ASN_INTEGER, $integer_value);
      }
    } elsif ($request_info->getMode() == MODE_GETNEXT) {
      if ($oid == new NetSNMP::OID(".1.3.6.1.4.1.8072.9999.9999.1.0")) {
        $request->setOID(".1.3.6.1.4.1.8072.9999.9999.1.1");
        $request->setValue(ASN_INTEGER, $integer_value);
      }
      elsif ($oid < new NetSNMP::OID(".1.3.6.1.4.1.8072.9999.9999.1.0")) {
        $request->setOID(".1.3.6.1.4.1.8072.9999.9999.1.0");
        $request->setValue(ASN_OCTET_STR, $string_value);
      }
    }
  }
}

my $agent = new NetSNMP::agent();
$agent->register("hello_world", ".1.3.6.1.4.1.8072.9999.9999",
                 \&hello_handler);
```

プラグインをテストするには、上記のプログラムを **/usr/share/snmp/hello_world.pl** にコピーして、以下の行を **/etc/snmp/snmpd.conf** の設定ファイルに追加して下さい:

```
perl do "/usr/share/snmp/hello_world.pl"
```

新しい Perl プラグインをロードするためには、SNMP Agent Daemon を再起動する必要があります。再起動されると、**snmpwalk** が新しいデータを返すはずです:

```
~]$ snmpwalk localhost NET-SNMP-MIB::netSnmpPlaypen
NET-SNMP-MIB::netSnmpPlaypen.1.0 = STRING: "hello world"
NET-SNMP-MIB::netSnmpPlaypen.1.1 = INTEGER: 8675309
```

snmpget を使用すると、ハンドラーの他のモードを使用することも可能です:

```
~]$ snmpget localhost \
    NET-SNMP-MIB::netSnmpPlaypen.1.0 \
    NET-SNMP-MIB::netSnmpPlaypen.1.1
NET-SNMP-MIB::netSnmpPlaypen.1.0 = STRING: "hello world"
NET-SNMP-MIB::netSnmpPlaypen.1.1 = INTEGER: 8675309
```

19.7. その他のリソース

システム情報の収集方法についての更なる情報は、次のリソースを参照して下さい。

19.7.1. インストールされているドキュメント

- **ps**(1) — **ps** コマンドの man ページです。
- **top**(1) — **top** コマンドの man ページです。
- **free**(1) — **free** コマンドの man ページです。

- **df**(1) — **`df`** コマンドの man ページです。
- **du**(1) — **`du`** コマンドの man ページです。
- **lspci**(8) — **`lspci`** コマンドの man ページです。
- **snmpd**(8) — **`snmpd`** サービスの man ページです。
- **snmpd.conf**(5) — 利用可能な設定指示文についての全ドキュメントを含む **`/etc/snmp/snmpd.conf`** ファイルの man ページです。

第20章 ログファイルの表示と管理

ログファイル とは、システムで実行しているカーネル、サービス、アプリケーションなどのシステムに関するメッセージを格納しているファイルです。各情報にはそれぞれ異なるログファイルがあります。例えば、デフォルトのシステムログファイル、セキュリティメッセージ専用のログファイル、cron タスク用のログファイルなどです。

ログファイルは、カーネルドライバーのロードを試行するなどシステムの問題を解決する場合やシステムへの無許可のログイン試行を探す場合に役立ちます。本章では、ログファイルの場所、ログファイルの閲覧方法、ログファイルの注意する項目を説明します。

一部のログファイルは、**rsyslogd** と呼ばれるデーモンで制御されています。**rsyslogd** で管理されているログファイルの一覧は **/etc/rsyslog.conf** 設定ファイル内にあります。

rsyslog は、**sysklogd** デーモンと置き換わる強化されたマルチスレッドの syslog デーモンです。**rsyslog** は **sysklogd** と同じ機能を持ちながら、強化されたフィルタリング、暗号化で保護されたメッセージのリレー、各種設定オプション、**TCP** または **UDP** プロトコルによる転送のサポートにより拡張します。**rsyslog** と **sysklogd** は互換性がある点に注意して下さい。

20.1. RSYSLOG の設定

rsyslog の主要な設定ファイルは **/etc/rsyslog.conf** です。これは、*グローバル指示文*、*ルール* またはコメント (ハッシュ記号 (#) の後に続く空行かテキスト) で構成されます。グローバル指示文とルールについては、後項で詳しく説明します。

20.1.1. グローバル指示文

グローバル指示文は、**rsyslogd** デーモンに適用する設定オプションを指定します。グローバル指示文は、通常 **rsyslogd** デーモンの動作や従うルールに影響する特定の事前定義された変数の値を指定します。すべてのグローバル指示文は、ドル記号 (**$**) で始まらなければなりません。1 行に指定できるのは、 1 つの指示文のみです。以下は、syslog メッセージキューの最大サイズを指定するグローバル指示文の例です:

```
$MainMsgQueueSize 50000
```

この指示文用に定義されたデフォルトのサイズ (**10,000** メッセージ) は、別の値を指定することで上書きされます (上記の例を参照)。

/etc/rsyslog.conf 設定ファイル内では、複数の指示文を定義することができます。指示文は、同じ指示文が再度検出されるまですべての設定オプションの動作に影響します。

すべての利用可能な設定指示文とそれらの詳しい説明を含む総合一覧は、**/usr/share/doc/rsyslog-<*version-number*>/rsyslog_conf_global.html** を参照して下さい。

20.1.2. モジュール

rsyslog は、そのモジュール式設計により、動的な機能を持つ各種の *モジュール* を提供します。モジュールは、サードパーティによる書き込みが可能な点に注意して下さい。大半のモジュールは、追加の入力 (以下の **入力モジュール** を参照) または出力 (以下の **出力モジュール** を参照) を提供します。各モジュール毎に特有の機能を提供するモジュールもあります。また、モジュールのロード後に利用可能になる追加の設定指示文を提供するモジュールもあります。モジュールをロードするには、以下の構文を使用します:

```
$ModLoad <MODULE>
```

$ModLoad は指定されたモジュールをロードするグローバル指示文であり、*<MODULE>* は目的のモジュールです。例えば、**Text File Input Module** (**imfile** — **rsyslog** を有効にして標準テキストファイルを syslog メッセージに変換) をロードしたい場合、**/etc/rsyslog.conf** 設定ファイルで以下の行を指定します:

```
$ModLoad imfile
```

rsyslog は、以下のような主要カテゴリに分割される多くのモジュールを提供します:

- 入力モジュール — 入力モジュールは、様々なソースからメッセージを収集します。入力モジュールの名前は、常に **imfile** や **imrelp** のように接頭辞 **im** で始まります。
- 出力モジュール — 出力モジュールは、様々な対象にメッセージを格納する機能を提供します。例えば、ネットワーク上でのメッセージの送信、データベースへのメッセージの格納やその暗号化などです。出力モジュールの名前は、**omsnmp** や **omrelp** のように常に接頭辞 **om** で始まります。
- フィルターモジュール — フィルターモジュールは、指定されたルールに従ってメッセージをフィルターできます。フィルターモジュールの名前は、常に接頭辞 **fm** で始まります。
- パーサモジュール — パーサモジュールは、メッセージパーサを使用して受信メッセージの内容を構文解析します。パーサモジュールの名前は、**pmrfc5424** や **pmrfc3164** のように常に接頭辞 **pm** で始まります。
- メッセージ修正モジュール — メッセージ修正モジュールは、syslog メッセージの内容を変更します。メッセージ修正モジュールは、その実装に関してのみ出力モジュールやフィルターモジュールとは異なりますが、同じインターフェースを共有します。
- 文字列ジェネレーターモジュール — 文字列ジェネレーターモジュールは、メッセージの内容を基にして文字列を生成し、**rsyslog** により提供されるテンプレート機能とよく動作します。テンプレートの詳細については、「テンプレート」 を参照して下さい。文字列ジェネレーターモジュールの名前は、**smfile** や **smtradfile** のように常に接頭辞 **sm** で始まります。
- ライブラリモジュール — ライブラリモジュールは、通常他のロード可能なモジュールに機能性を提供します。これらのモジュールは、必要な場合に **rsyslog** により自動的にロードされ、ユーザーは設定できません。

すべての利用可能なモジュールとそれらの詳しい説明を含む総合一覧は、http://www.rsyslog.com/doc/rsyslog_conf_modules.html を参照して下さい。

警告

rsyslog がモジュールをロードする場合、モジュールに対して一部の機能とデータへのアクセスを提供します。これはセキュリティ上の脅威となる可能性があります。セキュリティリスクを最小限にするために、信頼できるモジュールのみを使用するようにして下さい。

20.1.3. ルール

ルールは、syslog メッセージのサブセットを選択する *フィルター* 部分と、選択したメッセージで実行する内容を指定する *動作* 部分により指定されます。**/etc/rsyslog.conf** 設定ファイル内でルールを定義するには、フィルターとアクションの両方を 1 行で 1 つ以上の空白かタブで区切って定義します。フィルターの詳細については 「フィルターの条件」 を、動作の詳細については 「動作」 を参照して下さい。

20.1.3.1. フィルターの条件

rsyslog には、各種プロパティに従って syslog メッセージをフィルターする様々な方法があります。本項では、最も使用されるフィルターの条件を要約しています。

Facility (ファシリティ) /Priority (優先度) ベースのフィルター

syslog メッセージのフィルター動作で最も使用され良く知られているのは、ファシリティ/優先度をベースにしたフィルターです。これは、*facility (ファシリティ)* と *priority (優先度)* の 2 つの条件を基にして syslog メッセージをフィルターします。セレクターを作成するには、以下の構文を使用します:

```
<FACILITY>.<PRIORITY>
```

ここで:

- *<FACILITY>* は、特定の syslog メッセージを生成するサブシステムを指定します。例えば、**mail** サブシステムは、メール関連の全 syslog メッセージを処理します。*<FACILITY>* は、次のいずれかのキーワードで表すことができます:
  auth、**authpriv**、**cron**、**daemon**、**kern**、**lpr**、**mail**、**news**、**syslog**、**user**、**uucp** 及び **local0** から **local7**。

- *<PRIORITY>* は、syslog メッセージの優先度を指定します。*<PRIORITY>* は、次のいずれかのキーワードで表すことができます (昇順で表示):
  debug、**info**、**notice**、**warning**、**err**、**crit**、**alert** 及び **emerg**。

  優先度の前に等号 (**=**) を付けることによって、その優先度を持つ syslog メッセージのみが選択されるよう指定します。他のすべての優先度は無視されます。逆に、優先度の前に感嘆符 (**!**) を付けると、定義された優先度を持つメッセージ以外の全 syslog メッセージが選択されます。これらの 2 つの拡張子のどちらも使用しないと、定義済みあるいは優先度が高い syslog メッセージが選択されます。

上記で指定したキーワードに加えて、アスタリスク (*****) を使用して (アスタリスクを配置する場所がドットの前か後ろかにより)、すべてのファシリティ/優先度を定義することができます。**none** のキーワードを指定すると、優先度のないファシリティを指定することになります。

複数のファシリティと優先度を定義するには、単にカンマ (**,**) で区切るだけです。複数のフィルターを 1 行で定義するには、セミコロン (**;**) で区切ります。

以下に、簡単なファシリティ/優先度をベースにしたフィルターの例をいくつか示します:

```
kern.*    # Selects all kernel syslog messages with any priority
```

```
mail.crit    # Selects all mail syslog messages with priority crit and
higher.
```

```
cron.!info,!debug    # Selects all cron syslog messages except those
with the info or debug priority.
```

プロパティベースのフィルター

プロパティベースのフィルターを使用すると、***timegenerated*** や ***syslogtag*** などのプロパティによって syslog メッセージをフィルターできます。プロパティの詳細については、「プロパティ」を参照して下さい。フィルターで指定された各プロパティを使うと、表20.1「プロパティベースの比較処理　」 に一覧表示されているいずれかの比較処理を使用してプロパティと特定の値を比較することができます。

表20.1 プロパティベースの比較処理

比較処理	詳細
contains	指定された文字列が、プロパティにより提供されたテキストの一部にマッチするかどうかを確認します。
isequal	指定された文字列を、プロパティにより提供されたすべてのテキストと比較します。
startswith	指定された文字列が、プロパティにより提供されたテキストの接頭辞にマッチするかどうかを確認します。
regex	指定された POSIX BRE (Basic Regular Expression) 正規表現をプロパティにより提供されたテキストと比較します。
ereregex	指定された POSIX ERE (Extended Regular Expression) 正規表現をプロパティにより提供されたテキストと比較します。

プロパティベースのフィルターを定義するには、以下の構文を使用します:

```
:<PROPERTY>, [!]<COMPARE_OPERATION>, "<STRING>"
```

ここで:

- *<PROPERTY>* 属性は、希望するプロパティ (例えば ***timegenerated*** や ***hostname***) を指定します。
- オプションの感嘆符 (**!**) は、(比較処理の前に置く場合は) 比較処理の出力を無効にします。
- *<COMPARE_OPERATION>* 属性は、表20.1「プロパティベースの比較処理　」 内に一覧表示されているいずれかの比較処理を指定します。
- *<STRING>* 属性は、プロパティにより提供されるテキストの比較対象となる値を指定します。特定の文字 (引用符 (**"**) など) をエスケープさせるには、バックスラッシュ記号 (\) を使用します。

以下は、プロパティベースのフィルターの例です:

- 以下のフィルターは、メッセージテキスト内に文字列 **error** を含む syslog メッセージを選択します:

  ```
  :msg, contains, "error"
  ```

- 以下のフィルターは、ホスト名 **host1** から受信した syslog メッセージを選択します:

```
:hostname, isequal, "host1"
```

- 以下のフィルターは、**fatal** と **error** という単語の間にテキストがあるかどうかに関わらず (**fatal lib error** など)、それらの名前を含まない syslog メッセージを選択します:

```
:msg, !regex, "fatal .* error"
```

式ベースのフィルター

式ベースのフィルターは、定義済み算術演算、ブール演算または文字列演算に従って syslog メッセージを選択します。式ベースのフィルターは、**rsyslog** 自身のスクリプト言語を使用します。この言語の構文は、様々な式ベースのフィルターの例と共に **/usr/share/doc/rsyslog-*<version-number>*/rscript_abnf.html** で定義されています。

式ベースのフィルターを定義するには、以下の構文を使用します:

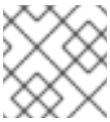
```
if <EXPRESSION> then <ACTION>
```

ここで:

- *<EXPRESSION>* 属性は、**$msg startswith 'DEVNAME'** や **$syslogfacility-text == 'local0'** など評価される式を表します。
- *<ACTION>* 属性は、式が **true** の値を返す場合に実行される動作を表します。

注記

式ベースのフィルターを定義する場合は、1 行内で定義しなければなりません。



注記

正規表現は、式ベースのフィルターでは現在サポートされていません。

BSD スタイルのブロック

rsyslog は、**/etc/rsyslog.conf** 設定ファイル内にある BSD スタイルのブロックをサポートします。各ブロックは、プログラムまたはホスト名のラベルの後に来るルールで構成されています。プログラムを含める場合は '!*<PROGRAM>*' ラベルを、除外する場合は '-*<PROGRAM>*' ラベルをそれぞれ使用します。また、ホスト名を含める場合は '+*<HOSTNAME>*' ラベルを、除外する場合は '-*<HOSTNAME>*' ラベルをそれぞれ使用します。

例20.1「BSD スタイルのブロック」 は、**yum** で生成された全メッセージをファイルに保存する BSD スタイルのブロックを示しています。

例20.1 BSD スタイルのブロック

```
!yum
*.*       /var/log/named.log
```

20.1.3.2. 動作

動作は、既に定義されたセレクターでフィルターされたメッセージで実行する内容を指定します。以下に、ルール内で定義できる動作をいくつか示します:

ログファイルへの syslog メッセージの保存

動作の多くは、syslog メッセージが保存されるログファイルを指定します。これは既に定義されたセレクターの後にファイルパスを指定することで実行できます。以下のルールは、すべての **cron** syslog メッセージを選択するセレクターとそれらのメッセージを **/var/log/cron.log** ログファイルに保存する動作で構成されています:

```
cron.* /var/log/cron.log
```

ダッシュ記号 (**-**) は、全 syslog メッセージの生成後に希望するログファイルの同期化を行わない場合に指定するファイルパスの接頭辞として使用します。

ファイルパスは、静的または動的のどちらかを指定できます。静的ファイルは上記の例のように、簡単なファイルパスで表されます。動的ファイルは、テンプレートと疑問符 (**?**) の接頭辞で表されます。テンプレートの詳細については、「動的なファイル名の生成」 を参照して下さい。

syslog メッセージは、指定したファイルが既存の **tty** である場合は標準出力 (特別な **tty**- 処理を使用)、指定したファイルが **/dev/console** のデバイスである場合はコンソール (特別な **/dev/console**- 処理を使用) へ X Window System 使用時にそれぞれ送信されます。

ネットワーク上での syslog メッセージの送信

rsyslog を使用すると、ネットワーク上で syslog メッセージを送受信できます。この機能により、1 台のマシン上で複数ホストの syslog メッセージを管理することができます。syslog メッセージをリモートマシンに転送するには、以下の構文を使用します:

```
@[(<OPTION>)]<HOST>:[<PORT>]
```

ここで:

- アットマーク (@) は、syslog メッセージが **UDP** プロトコルを使用してホストへ転送されることを示します。**TCP** プロトコルを使用するには、2 つのアットマークを空白なしで (@@) 使用します。
- *<OPTION>* 属性は、**z*<NUMBER>*** のようなオプションと置換できます。このオプションを使用すると、syslog メッセージの **zlib** 圧縮が可能です。*<NUMBER>* の属性は、圧縮レベルを指定します。複数のオプションを定義するには、それぞれカンマ (**,**) で区切るだけです。
- *<HOST>* 属性は、選択した syslog メッセージを受信するホストを指定します。
- *<PORT>* 属性は、ホストマシンのポートを指定します。

IPv6 アドレスをホストとして指定する場合は、アドレスを角括弧 (**[**, **]**) で囲みます。

以下は、ネットワーク上で syslog メッセージを転送する動作の例です (すべての動作には、いずれかの優先度を持つ全メッセージを選択するセレクターが前に付いている点に注意して下さい)。

```
*.* @192.168.0.1    # Forwards messages to 192.168.0.1 via the UDP
protocol
```

```
*.* @@example.com:18    # Forwards messages to "example.com" using port
18 and the TCP protocol
```

```
*.* @(z9)[2001::1]    # Compresses messages with zlib (level 9
compression)
                      # and forwards them to 2001::1 using the UDP
protocol
```

出力チャンネル

出力チャンネルは、主にログファイルのローテーション、つまりログファイルが拡張できる最大サイズを指定するために使用されます (ログファイルのローテーションに関する詳細は 「logrotate の設定」 を参照)。出力チャンネルを定義するには、以下の構文を使用します:

```
$outchannel <NAME>, <FILE_NAME>, <MAX_SIZE>, <ACTION>
```

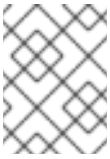
ここで:

- *<NAME>* 属性は、出力チャンネルの名前を指定します。
- *<FILE_NAME>* 属性は、出力ファイルの名前を指定します。
- *<MAX_SIZE>* 属性は、(*<FILE_NAME>* 内にある) 指定されたファイルが拡張できる最大サイズを表します。この値は **バイト** 単位で指定します。
- *<ACTION>* 属性は、*<MAX_SIZE>* で定義される最大サイズに到達した時に取るべき動作を指定します。

例20.2「出力チャンネルのログローテーション」 は、出力チャンネルを使用した簡単なログローテーションを示しています。最初に、出力チャンネルは ***$outchannel*** 指示文により定義され、その後ルールで使用されます。このルールは、いずれかの優先度を持つすべての syslog メッセージを選択して、取得した syslog メッセージで事前定義された出力チャンネルを使用します。上限 (例では **`100 MB`**) に到達すると、**`/home/joe/log_rotation_script`** が使用されます。このスクリプトに含まれるのは、異なるフォルダーへのファイルの移動、その中にある特有のコンテンツの編集/削除です。

例20.2 出力チャンネルのログローテーション

```
$outchannel log_rotation,/var/log/test_log.log, 104857600,
/home/joe/log_rotation_script

*.* $log_rotation
```



注記

現在、出力チャンネルは **rsyslog** でサポートされていますが、近い将来サポートされなくなります。

特定ユーザーへの syslog メッセージの送信

メッセージを送信したい相手のユーザー名を指定するだけで、**rsyslog** はそのユーザーに syslog メッセージを送信することができます。複数のユーザーを指定するには、各ユーザー名をカンマ (**,**) で区切ります。現在ログオンしている全ユーザーにメッセージを送るには、アスタリスク (*****) を使用します。

プログラムの実行

rsyslog により、選択された syslog メッセージ用のプログラムを実行することができ、**system()** 呼び出しを使用してシェル内でプログラムを実行します。実行するプログラムを指定するには、そのプログラムの前にキャレットの記号 (^) を付けます。指定後、受信したメッセージをフォーマットしてそれを 1 行のパラメーターとして指定した実行ファイルに渡すテンプレートを指定します (テンプレートの詳細については 「テンプレート」 を参照)。以下の例では、いずれかの優先度を持つ任意の syslog メッセージが選択され、***template*** テンプレートでフォーマットされ、パラメーターとして **test-program** プログラムに渡されます。その後、このプログラムは指定されたパラメーターで実行されます:

```
*.* ^test-program;template
```

警告

ホストからのメッセージを受け取り、シェル実行動作を使用する場合は、コマンドインジェクションの脆弱性が存在する場合があります。攻撃者は、ユーザーが実行するよう (操作により) 指定したプログラム内で攻撃者が指定するコマンドを挿入して実行するよう試行する場合があります。あらゆるセキュリティ脅威を回避するために、シェル実行動作の使用は十分に考慮して下さい。

データベースへの syslog メッセージの入力

選択された syslog メッセージは、**データベースライター** の動作を使用して、直接データベーステーブルに書き込むことができます。データベースライターは、以下の構文を使用します:

```
:<PLUGIN>:<DB_HOST>,<DB_NAME>,<DB_USER>,<DB_PASSWORD>;[<TEMPLATE>]
```

ここで:

- *<PLUGIN>* は、データベースの書き込みを処理する指定プラグインを呼び出します (例えば **ommysql** プラグイン)。
- *<DB_HOST>* 属性は、データベースのホスト名を指定します。
- *<DB_NAME>* 属性は、データベースの名前を指定します。
- *<DB_USER>* 属性は、データベースのユーザーを指定します。
- *<DB_PASSWORD>* 属性は、前述のデータベースユーザーが使用するパスワードを指定します。
- *<TEMPLATE>* 属性は、オプションである syslog メッセージを修正するテンプレートの使用を指定します。テンプレートの詳細については、「テンプレート」 を参照して下さい。

重要

現在、**rsyslog** は **MySQL** (詳細は **/usr/share/doc/rsyslog-*<version-number>*/rsyslog_mysql.html** を参照) データベースと **PostgreSQL** データベースのみに対応しています。**MySQL** および **PostgreSQL** のデータベースライター機能を使用するには、rsyslog-mysql および rsyslog-pgsql パッケージをそれぞれインストールします。また、**/etc/rsyslog.conf** 設定ファイルに適切なモジュールをロードするようにして下さい。

```
$ModLoad ommysql    # Output module for MySQL support
$ModLoad ompgsql    # Output module for PostgreSQL support
```

rsyslog モジュールの詳細については、「モジュール」 を参照して下さい。

別の方法として、**omlibdb** モジュールで提供される汎用のデータベースインターフェースも使用することができます。ただし、このモジュールは現在コンパイルされていません。

syslog メッセージの破棄

選択したメッセージを破棄するには、チルダの記号 (~) を使用します。以下のルールは、すべての cron syslog メッセージを破棄します:

```
cron.* ~
```

各セレクターごとに、複数の動作を指定することができます。1 つのセレクターに複数の動作を指定するには、各動作を別々の行に書き込み、それらの先頭にアンパサンドの記号 (&) を付けます。最初の動作だけがその行内でセレクターを指定できます。以下は、複数の動作を持つルールの一例です:

```
kern.=crit joe
& ^test-program;temp
& @192.168.0.1
```

上記の例では、クリティカルな優先度 (***crit***) を持つすべてのカーネル syslog メッセージが、ユーザー **joe** に送信され、テンプレート ***temp*** により処理された後、***test-program*** 実行ファイルに渡されます。その後、**UDP** プロトコルを介して **192.168.0.1** に転送されます。

指定されたセレクターが評価されるのは 1 回のみであるため、複数の動作を指定すると、希望する結果の全体的なパフォーマンスが向上します。

どの動作の後にも、メッセージをフォーマットするテンプレートを続けることができる点に注意して下さい。テンプレートを指定するには、動作の末尾にセミコロン (;) を付けてテンプレートの名前を指定します。

警告

テンプレートは動作で使用される前に定義する必要があります。そうしないと、無視されます。

テンプレートの詳細については 「テンプレート」 を参照して下さい。

20.1.3.3. テンプレート

rsyslog で生成される出力は、テンプレートを使用することでユーザーのニーズに応じた修正とフォーマットを行うことができます。テンプレートを作成するには、以下の構文を使用します:

```
$template <TEMPLATE_NAME>,"text %<PROPERTY>% more text", [<OPTION>]
```

ここで:

- ***$template*** は、後続のテキストがテンプレートを定義することを意味するテンプレート指示文です。
- ***<TEMPLATE_NAME>*** はテンプレートの名前です。この名前を使ってテンプレートを参照します。
- 二重引用符 ("...") の間にあるものはすべて、実際のテンプレートテキストです。このテキスト内では、改行のための **\n** や行頭復帰のための **\r** などの機能を使用する目的で文字をエスケープすることができます。**%** や **"** のような他の文字を文字どおりに使用したい場合にはエスケープする必要があります。

  2 つのパーセント記号 (%) 内で指定されたテキストは、プロパティの実際の値と結果的に置き換わる *プロパティ* を指定します。プロパティの詳細については、「プロパティ」 を参照して下さい。
- ***<OPTION>*** 属性は、テンプレートの機能を修正するオプションを指定します。これらをテンプレートテキストの内部 ("..." の間) で定義されているプロパティのオプションと混同しないで下さい。現在サポートされているテンプレートのオプションは、SQL クエリとしてテキストのフォーマット用に使用される ***sql*** と ***stdsql*** です。

注記

データベースライター (詳細は 「動作」 にある 『データベースへの syslog メッセージの入力』 の項を参照) は、***sql*** 及び ***stdsql*** のオプションがテンプレート内で指定されているかどうかを確認します。指定されていないと、データベースライターは何も動作を起こしません。これは SQL インジェクションなど、あらゆるセキュリティ脅威の可能性を排除するためです。

20.1.3.3.1. 動的なファイル名の生成

テンプレートを使用して、動的なファイルの名前を生成することができます。プロパティをファイルパスの一部として指定することで、それぞれ一意のプロパティに対して新しいファイルが作成されます。例として、***timegenerated*** プロパティを使用して、各 syslog メッセージ用に一意のファイル名を生成します:

```
$template DynamicFile,"/var/log/test_logs/%timegenerated%-test.log"
```

$template 指示文は、単にテンプレートを指定するだけである点に注意して下さい。この指示文を有効にするには、ルール内で使用しなければなりません。

```
*.* ?DynamicFile
```

20.1.3.3.2. プロパティ

テンプレート内 (2 つのパーセント記号 (%) の内側) で定義されたプロパティにより、*プロパティ置換* を使用することで syslog メッセージの各種コンテンツにアクセスできます。テンプレート内 (2 つの二重引用符 ("...") の間) でプロパティを定義するには、以下の構文を使用します:

```
%<PROPERTY_NAME>[:<FROM_CHAR>:<TO_CHAR>:<OPTION>]%
```

ここで:

- *<PROPERTY_NAME>* 属性は、プロパティの名前を指定します。すべての利用可能なプロパティとその詳しい説明を含む総合一覧は、**利用可能なプロパティ** のセクションの **/usr/share/doc/rsyslog-<*version-number*>/property_replacer.html** にあります。
- *<FROM_CHAR>* 属性と *<TO_CHAR>* 属性は、指定されたプロパティが動作する文字の範囲を表します。別の方法として、正規表現を使用して文字の範囲を指定することもできます。このためには、文字 **R** を *<FROM_CHAR>* 属性として指定し、希望する正規表現を *<TO_CHAR>* 属性として指定します。
- *<OPTION>* 属性は、プロパティのオプションを指定します。利用可能なプロパティとその詳しい説明を含む総合一覧は、**プロパティオプション** のセクションの **/usr/share/doc/rsyslog-<*version-number*>/property_replacer.html** にあります。

簡単なプロパティの例をいくつか以下に示します:

- 以下のプロパティは、syslog メッセージの全メッセージテキストを単に取得するだけです:

  ```
  %msg%
  ```

- 以下のプロパティは、syslog メッセージにあるメッセージテキストの最初の 2 文字を取得します:

  ```
  %msg:1:2%
  ```

- 以下のプロパティは、syslog メッセージの全メッセージテキストを取得して、最後のラインフィード文字を省きます:

  ```
  %msg:::drop-last-lf%
  ```

- 以下のプロパティは、syslog メッセージの受信時に生成されるタイムスタンプの最初の 10 文字を取得して、RFC 3999 日付基準に従ってそれをフォーマットします:

  ```
  %timegenerated:1:10:date-rfc3339%
  ```

20.1.3.3.3. テンプレートの例

本項は、**rsyslog** テンプレートの例をいくつか示しています。

例20.3「詳細な syslog メッセージのテンプレート」 は、syslog メッセージをフォーマットするテンプレートを示しています。そのため、メッセージの重要性、機能、メッセージ受信時のタイムスタンプ、ホスト名、メッセージタグ及びメッセージテキストを出力して、改行して終了します。

例20.3 詳細な syslog メッセージのテンプレート

```
$template
verbose,"%syslogseverity%,%syslogfacility%,%timegenerated%,%HOSTNAME%,%s
yslogtag%,%msg%
"
```

例20.4「ウォールメッセージのテンプレート」 は、従来型のウォールメッセージ (全ユーザーに送られるメッセージ。ここでのユーザーは、ログインしており、***mesg(1)*** パーミッションが ***yes*** に設定されているユーザーを意味します) に似たテンプレートを示しています。このテンプレートは、改行した行 (**\r** と **\n** を使用) にホスト名、メッセージタグ、タイムスタンプと共にメッセージテキストを出力してビープ音 (**\7** の使用) を鳴らします。

例20.4 ウォールメッセージのテンプレート

```
$template wallmsg,"\r
\7Message from syslogd@%HOSTNAME% at %timegenerated% ...\r
 %syslogtag% %msg%
\r"
```

例20.5「データベースフォーマットをしたメッセージのテンプレート」 は、syslog メッセージをフォーマットしてデータベースクエリとして使用できるようにするテンプレートを示しています。テンプレートオプションとして指定されている、テンプレートの最後にある ***sql*** オプションの使用に注目して下さい。これは、データベースライターに対してメッセージを MySQL **SQL** クエリとしてフォーマットするように指示します。

例20.5 データベースフォーマットをしたメッセージのテンプレート

```
$template dbFormat,"insert into SystemEvents (Message,
Facility,FromHost, Priority, DeviceReportedTime, ReceivedAt, InfoUnitID,
SysLogTag) values ('%msg%', %syslogfacility%,
'%HOSTNAME%',%syslogpriority%, '%timereported:::date-mysql%',
'%timegenerated:::date-mysql%', %iut%, '%syslogtag%')",sql
```

また、**rsyslog** には **RSYSLOG_** の接頭辞で特定される事前定義済みのテンプレートセットも含まれています。競合を回避するため、この接頭辞を使用したテンプレートは作成しないことが推奨されます。以下の一覧では、これらの事前定義されたテンプレートとその定義を示しています。

RSYSLOG_DebugFormat

```
"Debug line with all properties:
FROMHOST: '%FROMHOST%', fromhost-ip: '%fromhost-ip%', HOSTNAME:
'%HOSTNAME%', PRI: %PRI%,
syslogtag '%syslogtag%', programname: '%programname%', APP-NAME: '%APP-
NAME%', PROCID: '%PROCID%', MSGID: '%MSGID%',
TIMESTAMP: '%TIMESTAMP%', STRUCTURED-DATA: '%STRUCTURED-DATA%',
msg: '%msg%'
escaped msg: '%msg:::drop-cc%'
```

```
rawmsg: '%rawmsg%'

\"
```

RSYSLOG_SyslogProtocol23Format

```
"<%PRI%>1 %TIMESTAMP:::date-rfc3339% %HOSTNAME% %APP-NAME% %PROCID%
%MSGID% %STRUCTURED-DATA% %msg%
\"
```

RSYSLOG_FileFormat

```
"%TIMESTAMP:::date-rfc3339% %HOSTNAME% %syslogtag%%msg:::sp-if-no-1st-
sp%%msg:::drop-last-lf%
\"
```

RSYSLOG_TraditionalFileFormat

```
"%TIMESTAMP% %HOSTNAME% %syslogtag%%msg:::sp-if-no-1st-sp%%msg:::drop-
last-lf%
\"
```

RSYSLOG_ForwardFormat

```
"<%PRI%>%TIMESTAMP:::date-rfc3339% %HOSTNAME% %syslogtag:1:32%%msg:::sp-
if-no-1st-sp%%msg%\"
```

RSYSLOG_TraditionalForwardFormat

```
"<%PRI%>%TIMESTAMP% %HOSTNAME% %syslogtag:1:32%%msg:::sp-if-no-1st-
sp%%msg%\"
```

20.1.4. rsyslog コマンドラインの設定

sysklogd と同様に、一部の **rsyslog** の機能はコマンドラインオプションを使って設定できます。**rsyslog** のバージョン 3 では、このメソッドは非推奨であった点に注意して下さい。これらのオプションのいくつかを有効にするには、**rsyslog** が実行する互換性モードを指定しなければなりません。ただし、コマンドラインオプションを使った **rsyslog** の設定は避けることをお勧めします。

rsyslog が実行すべき互換モードを指定するには、**-c** オプションを使用します。パラメーターが指定されていない時は、**rsyslog** は **sysklogd** と互換性を持つよう試みます。これは、適宜設定を修正する設定指示文をアクティベートすることで部分的に可能です。そのため、使用している **rsyslog** のメジャーバージョンに一致する番号をこのオプションに指定して、**/etc/rsyslog.conf** 設定ファイルをそれに応じて更新することが推奨されます。例えば、**sysklogd** オプション (**rsyslog** のバージョン 3 で非推奨) を使用したい場合は、以下のコマンドを実行します:

```
~]# rsyslogd -c 2
```

後方互換モードを含む **rsyslogd** デーモンへ渡されるオプションは、**/etc/sysconfig/rsyslog** 設定ファイル内で指定できます。

各種 **rsyslogd** オプションの詳細については、**man rsyslogd** を参照して下さい。

20.2. ログファイルの場所の特定

ほとんどのログファイルは、**/var/log/** ディレクトリ内にあります。**httpd** や **samba** などの一部のアプリケーションは、**/var/log/** 内のディレクトリにログファイルを所有しています。

末尾に番号が付いた複数のファイル (例えば **cron-20100906**) が、**/var/log/** ディレクトリ内にあることに気づくでしょう。これらの番号は、ローテーションしたログファイルに付加されたタイムスタンプを表します。ログファイルは、ファイルサイズが大きくなり過ぎないようにローテートされます。**logrotate** パッケージには、cron タスクが含まれています。これは、**/etc/logrotate.conf** 設定ファイル及び **/etc/logrotate.d/** ディレクトリ内の設定ファイルに従ってログファイルを自動的にローテートします。

20.2.1. logrotate の設定

以下に、**/etc/logrotate.conf** 設定ファイルのサンプルを示します:

```
# rotate log files weekly
weekly
# keep 4 weeks worth of backlogs
rotate 4
# uncomment this if you want your log files compressed
compress
```

このサンプルの設定ファイル内にあるすべての行は、あらゆるログファイルに適用するグローバルオプションを定義します。この例では、ログファイルは毎週ローテートされ、ローテートされたログファイルは 4 週間保管されます。ローテートされたすべてのログファイルは、**gzip** により **.gz** 形式に圧縮されます。ハッシュ記号 (#) で始まる行は、コメントであり処理されません。

特定のログファイル用に設定オプションを定義して、それをグローバルオプション下に配置することもできます。ただし、**/etc/logrotate.d/** ディレクトリ内に特定ログファイル用の個別の設定ファイルを作成して、そこに設定オプションを定義することが推奨されます。

以下は、設定ファイルが **/etc/logrotate.d/** ディレクトリに配置されている例です:

```
/var/log/messages {
    rotate 5
    weekly
    postrotate
    /usr/bin/killall -HUP syslogd
    endscript
}
```

このファイル内の設定オプションは、**/var/log/messages** ログファイル専用の特有なものです。ここで指定された設定は、可能な場合はグローバルオプションを上書きします。そのため、ローテートされた **/var/log/messages** ログファイルは、グローバルオプションで定義された 4 週間ではなく、5 週間保管されます。

以下は、**logrotate** 設定ファイル内で指定できる指示文の一覧です:

- ***weekly*** — 週ごとにログファイルのローテーションを指定します。同様な指示文は、以下のとおりです:

- ***daily***
- ***monthly***
- ***yearly***

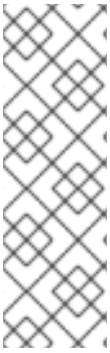
- ***compress*** — ローテートしたログファイルの圧縮を有効にします。同様な指示文は、以下のとおりです:
  - ***nocompress***
  - ***compresscmd*** — 圧縮に使用するコマンドを指定します。
  - ***uncompresscmd***
  - ***compressext*** — 圧縮に使用する拡張子を指定します。
  - ***compressoptions*** — 使用される圧縮プログラムに渡すオプションを指定できます。
  - ***delaycompress*** — ログファイルの圧縮を次回のログファイルのローテーションまで延期します。
- ***rotate <INTEGER>*** — ログファイルが削除されるか特定のアドレスにメール送信されるまでに、ログファイルが受けるローテーションの回数を指定します。**0** の値が指定されると、古いログファイルはローテートされる代わりに削除されます。
- ***mail <ADDRESS>*** — このオプションは、***rotate*** 指示文によって定義される回数だけローテートされたログファイルの指定したアドレスへのメール送信を有効にします。同様な指示文は、以下のとおりです:
  - ***nomail***
  - ***mailfirst*** — 間もなく期限切れになるログファイルではなく、ローテートされたばかりのログファイルがメール送信されるよう指定します。
  - ***maillast*** — ローテートされたばかりのログファイルではなく、間もなく期限切れになるログファイルがメール送信されるよう指定します。***mail*** が有効の場合は、これがデフォルトのオプションです。

指示文と各種設定オプションの総合一覧は、**logrotate** の man ページ (**man logrotate**) を参照して下さい。

20.3. ログファイルの表示

大半のログファイルはプレーンテキスト形式であるため、**Vi** や **Emacs** などのテキストエディターで閲覧できます。一部のログファイルは、システム上の全ユーザーにより読み取り可能ですが、ほとんどのログファイルは読み取りに root 権限が必要です。

インタラクティブなリアルタイムアプリケーションでシステムのログファイルを表示するには、**Log File Viewer (ログビューア)** を使用します。



注記

ログビューア を使用するには、最初に root で以下を実行して、ご使用のシステムに gnome-system-log パッケージがインストールされていることを確認します:

```
~]# yum install gnome-system-log
```

Yum を使用したパッケージのインストールについての詳細は、「パッケージのインストール」 を参照して下さい。

gnome-system-log パッケージをインストールしたら、**アプリケーション** → **システムツール** → **ログビューア** の順にクリックするか、シェルプロンプトで以下のコマンドを入力して、**ログビューア** を開きます。

```
~]$ gnome-system-log
```

このアプリケーションは、存在するログファイルのみを表示するだけです。そのため、図20.1「ログビューア」 で表示されている一覧とは異なる場合があります。

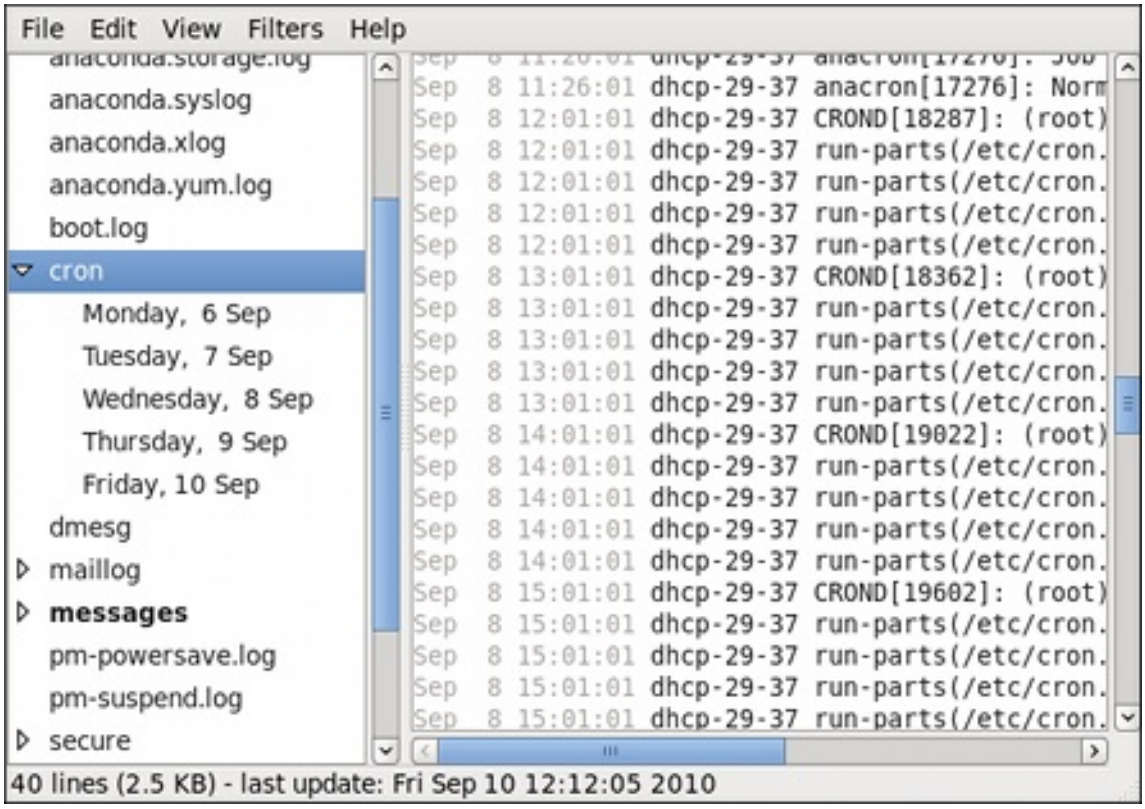


図20.1 ログビューア

ログビューア アプリケーションを使用すると、既存のログファイルをフィルターできます。目的のフィルターを定義/編集するためには、メニューから **フィルタ** をクリックして、**フィルターを管理** を選択します。

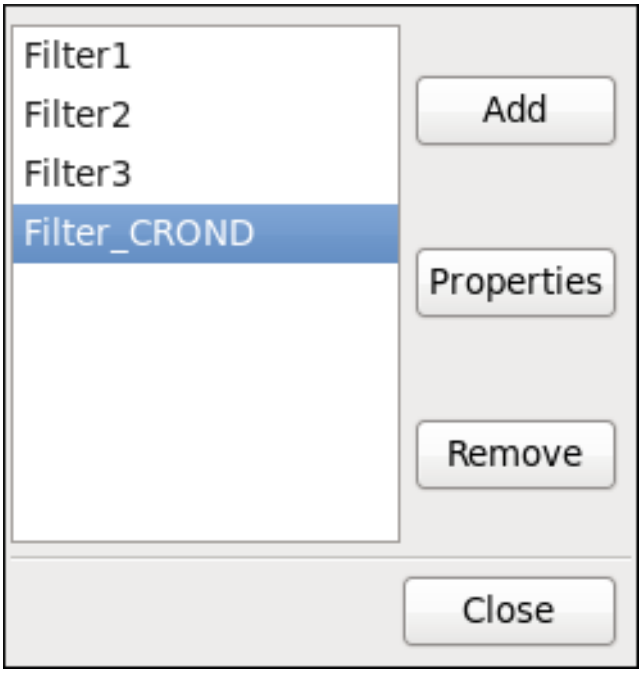


図20.2 ログビューア - フィルター

フィルターを追加/編集することで、図20.3「ログビューア - フィルターの定義」 のようにパラメーターを定義することができます。

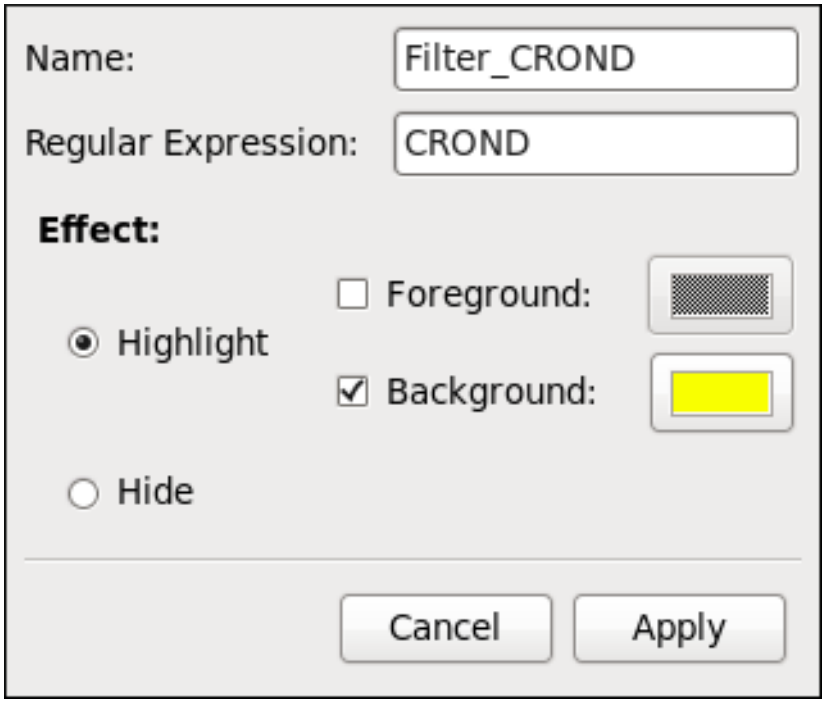


図20.3 ログビューア - フィルターの定義

フィルターを定義する場合は、以下のパラメーターを編集できます:

- **Name (名前)** — フィルターの名前を指定します。
- **Regular Expression (正規表現)** — ログファイルに適用され、その中の実行可能なテキストの文字列に一致するよう試行する正規表現を指定します。
- **Effect (効果)**

- **Highlight (強調表示)** — これが有効な場合は、検索結果は選択した色で強調されます。強調させる場所をテキストの背面/前面から選択できます。
- **Hide (非表示)** — これが有効な場合は、検索結果は閲覧中のログファイルからは非表示になります。

1 つ以上のフィルターを定義したら、**フィルター** メニューからそれを選択できます。フィルターは自動的に定義した文字列を検索して、現在閲覧しているログファイル内にあるすべての正しいマッチを強調/非表示します。

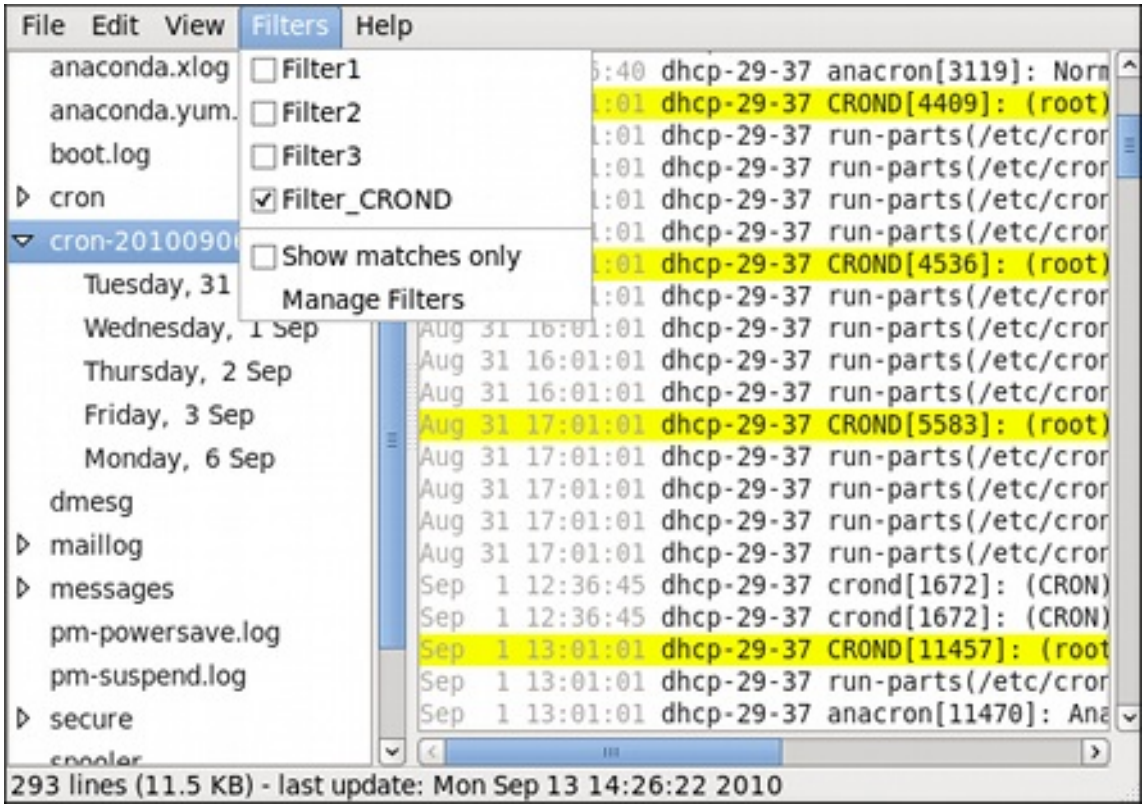


図20.4 ログビューア - フィルターの有効化

Show matches only (マッチしたもののみ表示) のオプションにチェックマークを入れると、現在閲覧しているログファイル内にマッチする文字列のみが表示されます。

20.4. ログファイルの追加

一覧で閲覧したいログファイルを追加するには、**ファイル** → **開く** の順に選択します。その後、閲覧したいログファイルのディレクトリとファイル名を選択できる **ファイルの選択** ウィンドウが表示されます。図20.5「ログビューア - ログファイルの追加」 は、**ファイルの選択** ウィンドウを示しています。

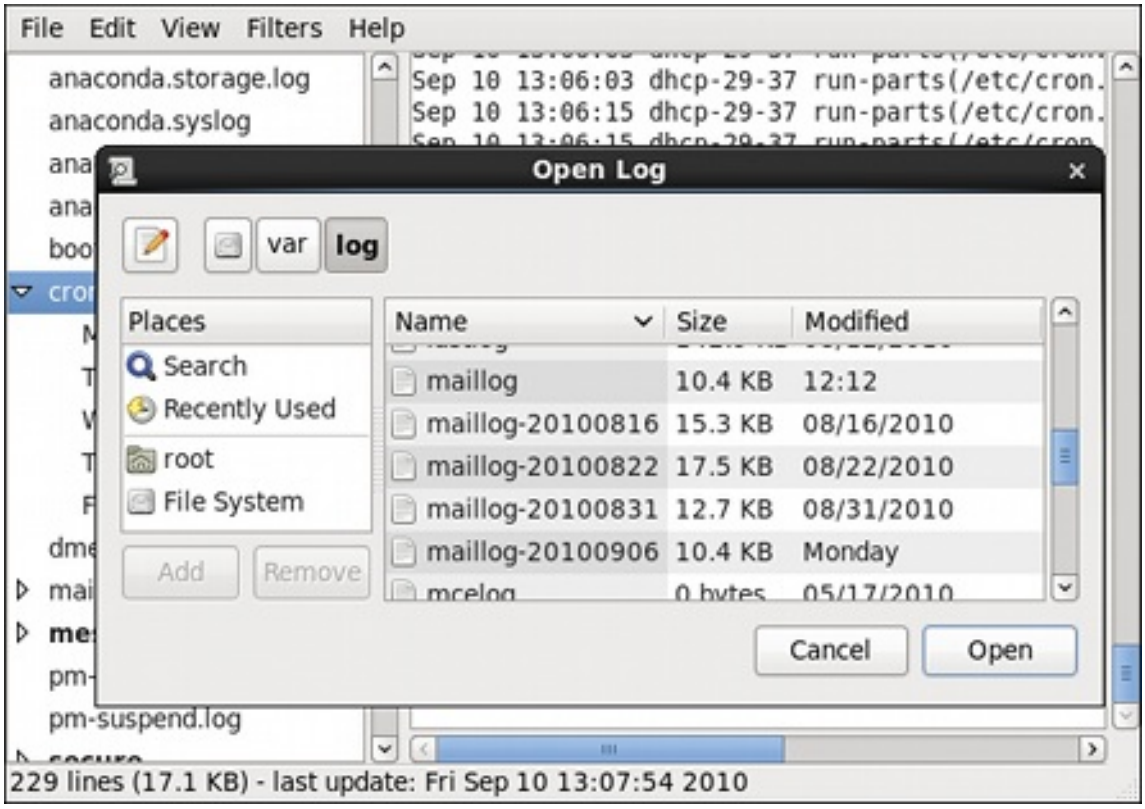
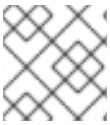


図20.5 ログビューア - ログファイルの追加

ファイルを開くには、**開く** ボタンをクリックします。ファイルは表示中の一覧に直ちに追加されるため、そのファイルを選択してコンテンツを表示することができます。



注記

ログビューア を使用すると **.gz** 形式で圧縮されたログファイルを開くこともできます。

20.5. ログファイルのモニタリング

ログビューア は、デフォルトで開いているすべてのログをモニターします。モニターされているログファイルに新しい行が追加されると、そのログ名はログの一覧に太字で表示されます。ログファイルが選択/表示されると、ログファイルの最下部に新しい行が太字で表示されます。図20.6「ログビューア - 新しいログ通知」 は、**cron** ログファイル及び **messages** ログファイル内の新しい通知を示しています。**cron** ログファイルをクリックすると、ファイル内のログが太字で新しい行に表示されます。

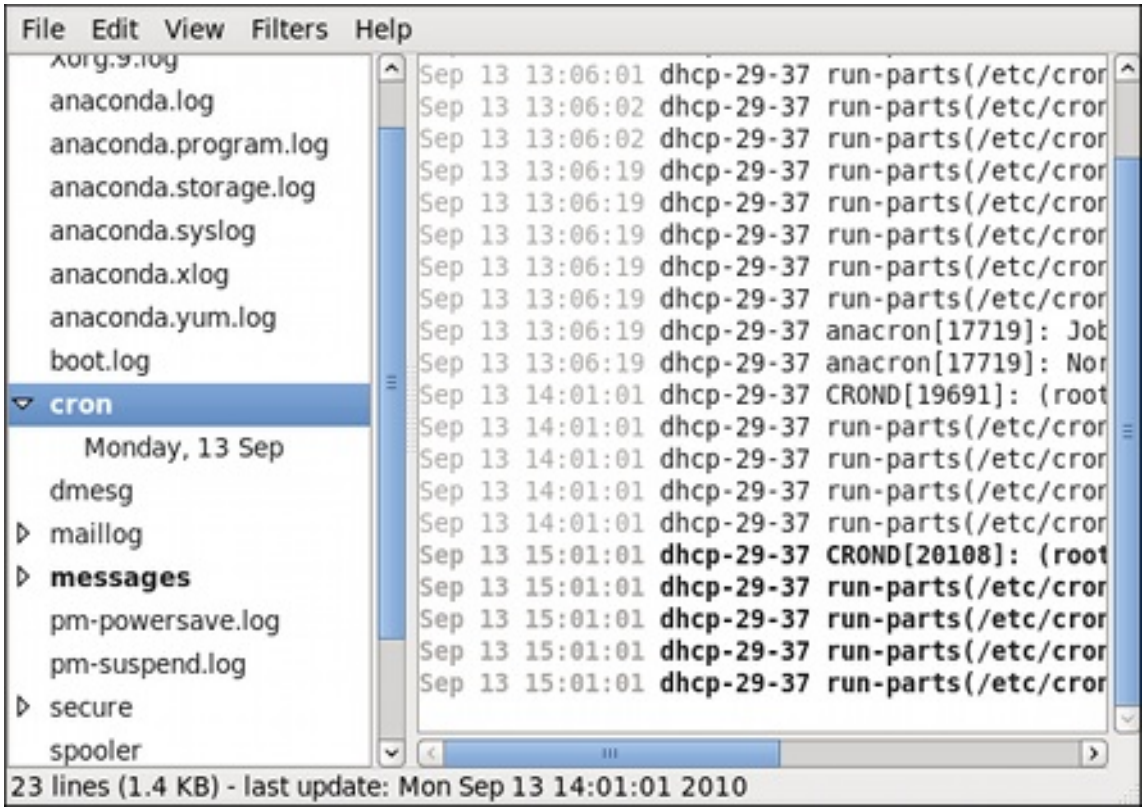


図20.6 ログビューア - 新しいログ通知

20.6. その他のリソース

rsyslog、**logrotate** 及びログファイル全般の詳細については、以下のリソースを参照して下さい。

20.6.1. インストールされているドキュメント

- **rsyslogd**(8) — **`rsyslogd`** デーモンの man ページが使用量についての詳細情報を提供します。
- **rsyslog.conf**(5) — **`/etc/rsyslog.conf`** 設定ファイルの man ページが、利用可能な設定オプションについての詳細情報を提供します。
- **logrotate**(8) — **logrotate** ユーティリティの man ページが設定および使用量に関する詳細情報を提供します。
- **`/usr/share/doc/rsyslog-<version-number>/`** — rsyslog パッケージをインストールした後に、このディレクトリには HTML 形式の広範囲に及ぶドキュメント **rsyslog** が格納されます。

20.6.2. 役立つ Web サイト

- http://www.rsyslog.com/ — rsyslog 機能の完全な技術的分析、ドキュメント、設定サンプル、及びビデオチュートリアルを提供しています。
- http://wiki.rsyslog.com/index.php/Main_Page — **`/etc/rsyslog.conf`** 設定の役立つサンプルが記載されています。

第21章 システムタスクの自動化

ジョブとしても知られるタスクは、指定した期日に指定した期間だけ、またはシステムのロード平均が 0.8 を下回る時に、自動的に実行するように設定できます。

Red Hat Enterprise Linux は、システムを最新状態に保つために重要なシステムタスクを実行するように事前設定されています。例えば、**locate** コマンドで使用される slocate データベースは毎日更新されます。システム管理者は、この自動化タスクを使用して定期的なバックアップ、システムの監視、カスタムスクリプトの実行などの作業を実施することができます。

Red Hat Enterprise Linux は、**cron**、**anacron**、**at**、 **batch** といった自動化タスクユーティリティを備えています。

ユーティリティはすべて、異なるジョブタイプのスケジュール管理を目的としています。Cron と Anacron は繰り返されるジョブのスケジュール管理をするのに対し、At と Batch は 1 回のみのジョブのスケジュール管理をします(「Cron と Anacron」 と 「At コマンドと Batch コマンド」 をそれぞれ参照してください)。

21.1. CRON と ANACRON

Cron と Anacron はいずれも、時刻、月のうちの日付、月、曜日、週で定義されたある時点で定期的なタスクの実行をスケジュールすることができるデーモンです。

Cron は、最高で 1 分おきの頻度でジョブの実行が可能です。しかし、ユーティリティはシステムが継続的に稼働状態であることを前提としており、ジョブがスケジュールされている時にシステムが稼働していない場合は、ジョブは実行されません。

一方でAnacron は、ジョブがスケジュールされている時点でシステムが稼働していなくても、ジョブを記憶しています。そして、次回にシステムが立ち上がった時にジョブを実行します。しかし、Anacron は 1 日に 1 回しかジョブを実行できません。

21.1.1. Cron と Anacron のインストール

Cron と Anacron をインストールするには、Cron は cronie パッケージを、Anacron はcronie-anacron パッケージを (cronie-anacron は cronie のサブパッケージ) をインストールする必要があります。

システムにパッケージがインストール済みかどうかを確認するには、**rpm -q cronie cronie-anacron** コマンドを発行します。インストール済みの場合は、cronie および cronie-anacron パッケージの完全名が返されるか、パッケージが利用可能でないことを告げられます。

これらのパッケージをインストールするには、以下の形式で **yum** コマンドを使用します。

```
yum install package
```

例えば、Cron と Anacron の両方をインストールするには、シェルプロンプトで以下を入力します。

```
~]# yum install cronie cronie-anacrone
```

このコマンドを実行するには、スーパーユーザー権限 (つまり **root** でログインしている) が必要である点に注意して下さい。Red Hat Enterprise Linux に新しいパッケージをインストールする方法の詳細については、「パッケージのインストール」 を参照して下さい。

21.1.2. Crond サービスの実行

cron ジョブと anacron ジョブの両方は、**crond** サービスが選びます。このセクションでは、**crond** サービスを起動/停止/再起動する方法、それを特定のランレベルで有効にする方法について説明します。ランレベルの概念や Red Hat Enterprise Linux におけるシステムサービスの管理方法全般の詳細については、10章 *サービスとデーモン* を参照して下さい。

21.1.2.1. Cron サービスの起動と停止

サービスが稼働中かどうかを確認するには、**/sbin/service dhcpd6 start** コマンドを使用します。

現行のセッションで **crond** サービスを実行するには、シェルプロンプトで **root** として以下を入力します。

```
service crond start
```

ブート時にサービスが自動的に起動するよう設定するには、以下のコマンドを使用します:

```
chkconfig crond on
```

このコマンドで、ランレベル 2、3、4、5 でサービスが有効になります。別の方法として、「サービスの有効化と無効化」 にあるように **サービスの設定** ユーティリティを使用することもできます。

21.1.2.2. Cron サービスの停止

crond サービスを停止するには、シェルプロンプトで **root** として以下を入力します。

```
service crond stop
```

ブート時のサービスの起動を無効にするには、以下のコマンドを使用します:

```
chkconfig crond off
```

このコマンドで、すべてのランレベルでサービスが無効になります。別の方法として、「サービスの有効化と無効化」 にあるように **サービスの設定** ユーティリティを使用することもできます。

21.1.2.3. Cron サービスの再起動

crond サービスを再起動するには、シェルプロンプトで以下を入力します。

```
service crond restart
```

このコマンドですばやくサービスを停止して、再度開始します。

21.1.3. Anacron ジョブの設定

ジョブをスケジュールするための主要設定ファイルは **/etc/anacrontab** ファイルで、root ユーザーのみがアクセスできます。このファイルには、以下のような行が記載されています。

```
SHELL=/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
# the maximal random delay added to the base delay of the jobs
```

```
RANDOM_DELAY=45
# the jobs will be started during the following hours only
START_HOURS_RANGE=3-22

#period in days   delay in minutes   job-identifier   command
1         5     cron.daily    nice run-parts /etc/cron.daily
7         25    cron.weekly   nice run-parts /etc/cron.weekly
@monthly  45    cron.monthly  nice run-parts /etc/cron.monthly
```

最初の 3 行は、anacron タスクが実行される環境を設定するための変数を定義します。

- **SHELL** — ジョブの実行に使用されるシェル環境です (例えば、Bash シェル)。
- **PATH** — 実行可能なプログラムへのパスです。
- **MAILTO** — anacron ジョブの出力を Email で受け取るユーザーのユーザー名です。

  MAILTO 変数が定義されていない場合、(**MAILTO=**)、Email は送信されません。

以下の 2 つの変数は、定義済みジョブのスケジュール時間を修正します。

- **RANDOM_DELAY** — ジョブごとに指定されている **delay in minutes** 変数に追加される最大の時間分数。

  最短の遅延時間は、デフォルトで 6 分に設定されています。

  例えば、**RANDOM_DELAY** が **12** に設定されていたとすると、その特定の anacrontab における各ジョブの **delay in minutes** に 6 分から 12 分が追加されます。**RANDOM_DELAY** は **6** 未満の設定も可能で、**0** の設定もできます。**0** に設定されると、ランダムな遅延は追加されません。これは例えば、1 つのネットワーク接続を共有する多くのコンピューターが毎日同じデータをダウンロードしなくてはならない場合などに有用です。

- **START_HOURS_RANGE** — スケジュールされているジョブが実行可能になる時間単位の間隔です。

  停電などで時間間隔を逸してしまった場合は、その日にスケジュールされていたジョブは実行されません。

/etc/anacrontab ファイル内の残りの行は、スケジュールされているジョブを表しており、以下の形式を取ります。

```
period in days   delay in minutes   job-identifier   command
```

- **period in days** — ジョブの実行頻度の日数。

  属性値は整数もしくはマクロ (**@daily**、**@weekly**、**@monthly**) として定義可能です。**@daily** は整数の 1 を、**@weekly** は整数の 7 を表し、and **@monthly** 月の長さに関係なく毎月 1 回ジョブが実行されることを指定します。

- **delay in minutes** — ジョブの実行までに anacron が待機する分数です。

  属性値は整数として定義されます。値が **0** に設定されると、待機は適用されません。

- **job-identifier** — ログファイルで使用される特定のジョブを指す一意な名前です。
- **command** — 実行するコマンドです。

このコマンドは、**ls /proc >> /tmp/proc** などのコマンドやカスタムスクリプトを実行するコマンドにすることができます。

先頭にハッシュ記号 (#) が付いた行はコメント行で、処理の対象外です。

21.1.3.1. Anacron ジョブの例

以下の例は、簡単な **/etc/anacrontab** ファイルを示しています:

```
SHELL=/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root

# the maximal random delay added to the base delay of the jobs
RANDOM_DELAY=30
# the jobs will be started during the following hours only
START_HOURS_RANGE=16-20

#period in days   delay in minutes   job-identifier   command
1         20    dailyjob      nice run-parts /etc/cron.daily
7         25    weeklyjob     /etc/weeklyjob.bash
@monthly  45    monthlyjob    ls /proc >> /tmp/proc
```

この **anacrontab** ファイル内で定義されているジョブはすべて、ランダムに 6 分から 30 分の間で遅延することになり、16:00 時から 20:00 時の間に実行できます。

最初に定義してあるジョブは毎日、16:26 から 16:50 の間に実行されます (RANDOM_DELAY が 6 分から 30 分の間で、delay in minutes 属性がこれに 20 分追加します) 。このジョブに指定されているコマンドは、**run-parts** スクリプトを使って **/etc/cron.daily** ディレクトリ内にあるすべての現行プログラムを実行します(**run-parts** スクリプトがディレクトリをコマンドラインの引数として受け取り、ディレクトリ内のプログラムを順にすべて実行していきます)。

2 つ目のジョブは、**/etc** ディレクトリの **weeklyjob.bash** スクリプトを週 1 回実行します。

3 つ目のジョブは、**/proc** のコンテンツを月 1 回 **/tmp/proc** file (**ls /proc >> /tmp/proc**) に書き込むコマンドを実行します。

21.1.4. Cron ジョブの設定

cron ジョブの設定ファイル **/etc/crontab** (root のみ修正可能) には、以下の行が記載されています。

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/
# For details see man 4 crontabs
# Example of job definition:
# .---------------- minute (0 - 59)
# | .------------- hour (0 - 23)
# | | .---------- day of month (1 - 31)
# | | | .------- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | .---- day of week (0 - 6) (Sunday=0 or 7) OR
sun,mon,tue,wed,thu,fri,sat
# | | | | |
# * * * * * username  command to be executed
```

最初の 3 行には、**anacrontab** ファイルと同様に、**SHELL** と **PATH**、**MAILTO** の変数定義が記載されています。これらの変数に関する詳細は、「Anacron ジョブの設定」 を参照してください。

さらにこのファイルは、**HOME** 変数を定義できます。**HOME** 変数は、ジョブが行うコマンドやスクリプトの実行時にホームディレクトリとして使用されるディレクトリを定義します。

/etc/crontab ファイルの残りの行はスケジュールされているジョブを示しており、以下の形式を取ります。

```
minute   hour   day   month   day of week   username   command
```

以下は、ジョブの実行時間を定義します。

- **minute** — 0 から 59 までの任意の整数
- **hour** — 0 から 23 までの任意の整数
- **day** — 1 から 31 までの任意の整数 (月が指定されている場合は、その月の有効な日)
- **month** — 1 から 12 までの任意の整数 (あるいは jan や feb など月の省略形)
- **day of week** — 0 から 7 までの任意の整数。0 および 7 は日曜日を示す (あるいは sun や mon など曜日の省略形)

以下は、他のジョブプロパティを定義します。

- **username** — ジョブが実行されるユーザーを指定します。
- **command** — 実行するコマンドを指定します。

  コマンドは、**ls /proc /tmp/proc** などのコマンド、もしくはカスタムスクリプトを実行するコマンドを指定可能です。

上記のいずれかの値にアスタリスク (*) を使用すると、すべての有効な値を指定することができます。例えば、月の値をアスタリスクで定義すると、ジョブは他の値による制約の範囲内で毎月実行されます。

整数の間のハイフン (-) は、整数の範囲を指定します。例えば、**1-4** は整数 1、2、3、4 を意味します。

コンマ (,) で区切られた値の一覧は、一覧を指定します。例えば、**3, 4, 6, 8** はそれらの 4 つの整数を示します。

スラッシュ (/) を使用すると、ステップ値を指定することができます。範囲の後に ***/integer*** を付けると、その範囲内で、指定した整数の値をスキップすることができます。例えば、**0-59/2** とすると、分フィールドで 1 分おきの間隔を定義することができます。ステップ値は、アスタリスクと併用することも可能です。例えば、月フィールドで値 ***/3** を指定すると、3 カ月ごとにタスクが実行されます。

先頭にハッシュ記号 (#) が付いた行はコメント行で、処理の対象外です。

root 以外のユーザーは、**crontab** ユーティリティを使用して cron タスクを設定できます。ユーザー定義の crontab は **/var/spool/cron/** ディレクトリに保存されており、それを作成したユーザーが実行するかのように実行されます。

ユーザーとして crontab を作成するには、そのユーザーとしてログインし、コマンド **crontab -e** を

入力してから、**VISUAL** または**EDITOR** 環境変数で指定されているエディターを使用してユーザーの crontab を編集します。このファイルは **/etc/crontab** と同じ形式を使用します。crontab への変更を保存すると、crontab はユーザー名に応じて保存され、ファイル **/var/spool/cron/*username*** に書き込まれます。自分の crontab ファイルのコンテンツを一覧表示するには、**crontab -l** コマンドを使用します。

注記

crontab ユーティリティでジョブを定義する際は、ユーザーを指定しないでください。

/etc/cron.d/ ディレクトリには、**/etc/crontab** ファイルと同じ構文のファイルが格納されています。このディレクトリ内でファイルの作成と変更ができるのは root のみです。

注記

cron デーモンは、変更を確認するために **/etc/anacrontab** ファイル、**/etc/crontab** ファイル、**/etc/cron.d/** ディレクトリ、**/var/spool/cron/** ディレクトリを毎分チェックします。何らかの変更が発見されると、メモリーにロードされます。そのため anacrontab または crontab のファイルが変更されても、デーモンを再起動する必要はありません。

21.1.5. Cron へのアクセスの制御

Cron へのアクセスを制御するには、**/etc/cron.allow** ファイルと**/etc/cron.deny** ファイルを使用します。この両方のアクセス制御ファイルの形式は各行にユーザー名を 1 つ配置します。どちらのファイルでも空白は許可されません。

cron.allow ファイルが存在する場合は、そのファイルに記載されているユーザーのみが cron の使用を許可され、**cron.deny** ファイルは無視されます。

cron.allow が存在しない場合は、**cron.deny** に表示されているユーザーは cron を使用できません。

アクセス制御ファイルが変更された場合でも、cron デーモン (**crond**) を再起動する必要はありません。アクセス制御ファイルは、ユーザーが cron ジョブの追加または削除を試みるたびにチェックされます。

アクセス制御ファイルに記載されているユーザー名に関わらず、root ユーザーは常時 cron を使用できます。

アクセスは、PAM (Pluggable Authentication Modules) を介して制御することもできます。これらの設定値は **/etc/security/access.conf** に格納されています。例えば、このファイルに以下の行を追記すると、root ユーザー以外のユーザーは crontab の作成ができなくなります。

```
-:ALL EXCEPT root :cron
```

禁止されているジョブは、適切なログファイルにログ記録されます。また、“crontab -e” を使用している時は、標準出力に返されます。詳細は、**access.conf.5** を参照して下さい (**man 5 access.conf** のコマンドで表示します)。

21.1.6. Cron ジョブのブラックリスト/ホワイトリスト

ジョブのブラックリストとホワイトリストは、ジョブの実行不要部分を定義するために使用しま

す。**/etc/cron.daily** などの cron ディレクトリ上の **run-parts** スクリプトを呼び出す際にこれが有用となります。ユーザーが、ディレクトリ内にあるプログラムをジョブのブラックリストに加えると、**run-parts** スクリプトはこのプログラムを実行しません。

ブラックリストを定義するには、ディレクトリ内で **run-parts** スクリプトの実行元となる **jobs.deny** ファイルを作成します。例えば、**/etc/cron.daily** から特定のプログラムを省略する必要がある場合には、**/etc/cron.daily/jobs.deny** ファイルを作成する必要があります。このファイル内で、実行を省略するプログラムの名前を指定します（同一ディレクトリ内にあるプログラムのみ可能）。**run-parts /etc/cron.daily** のように、cron.daily ディレクトリからプログラムを実行するコマンドをジョブが実行する場合は、**jobs.deny** ファイル内で定義されたプログラムは実行されません。

ホワイトリストを定義するには、**jobs.allow** ファイルを作成します。

jobs.deny と **jobs.allow** の原則は、「Cron へのアクセスの制御」 セクションに説明した **cron.deny** と **cron.allow** の原則と同じです。

21.2. AT コマンドと BATCH コマンド

cron は定期的なタスクのスケジュールに使うのに対して、**At** ユーティリティは特定の時刻での 1 回限りのタスクのスケジュールに使われます。また、**Batch** ユーティリティは、システム負荷平均が 0.8 を下回った時に実行される 1 回限りのタスクのスケジュールに使用します。

21.2.1. At と Batch のインストール

システムに at パッケージがインストール済みかどうかを確認するには、**rpm -q at** コマンドを発行します。インストール済みの場合は、at パッケージの完全名が返され、そうでない場合はパッケージが利用可能でないことを告げられます。

これらのパッケージをインストールするには、以下の形式で **yum** コマンドを使用します。

```
yum install package
```

At および Batch をインストールするには、シェルプロンプトで以下を入力します。

```
~]# yum install at
```

このコマンドを実行するには、スーパーユーザー権限 (つまり **root** でログインしている) が必要である点に注意して下さい。Red Hat Enterprise Linux に新しいパッケージをインストールする方法の詳細については、「パッケージのインストール」 を参照して下さい。

21.2.2. At サービスの実行

At ジョブと Batch ジョブの両方は、**atd** サービスが選びます。このセクションでは、**atd** サービスを起動/停止/再起動する方法、それを特定のランレベルで有効にする方法について説明します。ランレベルの概念や Red Hat Enterprise Linux におけるシステムサービスの管理方法全般の詳細については、10 章 *サービスとデーモン* を参照して下さい。

21.2.2.1. At サービスの起動と停止

サービスが稼働中かどうかを確認するには、**service atd status** コマンドを使用します。

現行のセッションで **atd** サービスを実行するには、シェルプロンプトで **root** として以下を入力します。

```
service atd start
```

ブート時にサービスが自動的に起動するよう設定するには、以下のコマンドを使用します。

```
chkconfig atd on
```

注記

ブート時にサービスが自動的に起動することが推奨されます。

このコマンドで、ランレベル 2、3、4、5 でサービスが有効になります。別の方法として、「サービスの有効化と無効化」 にあるように **サービスの設定** ユーティリティを使用することもできます。

21.2.2.2. At サービスの停止

atd サービスを停止するには、シェルプロンプトで **root** として以下を入力します。

```
service atd stop
```

ブート時のサービスの起動を無効にするには、以下のコマンドを使用します:

```
chkconfig atd off
```

このコマンドで、すべてのランレベルでサービスが無効になります。別の方法として、「サービスの有効化と無効化」 にあるように **サービスの設定** ユーティリティを使用することもできます。

21.2.2.3. At サービスの再起動

atd サービスを再起動するには、シェルプロンプトで以下を入力します。

```
service atd restart
```

このコマンドですばやくサービスを停止して、再度開始します。

21.2.3. At ジョブの設定

At ユーティリティを使って特定の時刻に 1 回限りのジョブをスケジュールするには、以下を実行します。

1. コマンドラインで、コマンド **at *TIME*** を入力します。ここでの ***TIME*** は、コマンドを実行する時刻です。

   TIME 引数は以下のいずれかの形式で定義できます。

   - ***HH*:*MM*** は、特定の時間と分数を指定します。例えば、**04:00** は 4:00 a.m. を指定します。
   - **midnight** は、12:00 a.m. を指定します。
   - **noon** は、12:00 p.m. を指定します。

- **teatime** は、4:00 p.m. を指定します。
- ***MONTHDAYYEAR*** 形式；例えば、**January 15 2012** 2002 年 1 月 15 日を指定します。年表示はオプションです。
- ***MMDDYY***、***MM/DD/YY***、***MM.DD.YY*** のいずれかの形式; 例えば、**011512** 2012 年 1 月 15 日となります。
- **now + *TIME*** ここでの *TIME* は、整数と分、時間、日、週の値として定義されます。例えば、**now + 5 days** は、コマンドが5日後の現在と同じ時刻に実行されることを指定します。

  オプションの日付の前に時間を指定する必要があります。時間の形式に関する詳細は、**/usr/share/doc/at-<*version*>/timespec** のテキストファイルを参照してください。

もし指定の時刻が過ぎてしまった場合には、翌日の同じ時刻に実行されます。

2. 表示されている **at>** プロンプトで、ジョブコマンドを定義します。
   - ジョブが実行するコマンドを入力し、**Enter** を押します。複数のコマンドがある場合は、このステップを繰り返します。
   - プロンプトでシェルスクリプトを入力し、スクリプトの各行の後で **Enter** を押します。

     ジョブは、ユーザーの **SHELL** 環境で設定されているシェル、ユーザーのログインシェル、**/bin/sh** のいずれか (最初に検出されたもの) を使用します。
3. 完了したら、空白行で **Ctrl**+**D** を押してプロンプトを終了します。

コマンドセットやスクリプトが標準出力に情報を表示しようと試みる場合は、その出力はユーザーに電子メールで送信されます。

保留中のジョブを表示するには、**atq** コマンドを使用します。詳細は、「保留中のジョブの表示」 を参照して下さい。

at コマンドの使用を制限することもできます。詳細は、「at と batch へのアクセスの制御」 を参照してください。

21.2.4. Batch ジョブの設定

Batch アプリケーションは、システム負荷平均が 0.8 を下回った時に、定義済みの 1 回限りのタスクを実行します。

Batch ジョブは、以下の手順で定義します。

1. コマンドラインで **batch** コマンドを入力します。
2. 表示されている **at>** プロンプトで、ジョブコマンドを定義します。
   - ジョブが実行するコマンドを入力し、**Enter** を押します。複数のコマンドがある場合は、このステップを繰り返します。
   - プロンプトでシェルスクリプトを入力し、スクリプトの各行の後で **Enter** を押します。

スクリプトが入力される場合は、 ジョブは、ユーザーの **SHELL** 環境で設定されているシェル、ユーザーのログインシェル、**/bin/sh** のいずれか (最初に検出されたもの) を使用します。

3. 完了したら、空白行で **Ctrl+D** を押してプロンプトを終了します。

コマンドセットやスクリプトが標準出力に情報を表示しようと試みる場合は、その出力はユーザーに電子メールで送信されます。

保留中のジョブを表示するには、**atq** コマンドを使用します。詳細は、「保留中のジョブの表示」 を参照して下さい。

batch コマンドの使用を制限することもできます。詳細は、「at と batch へのアクセスの制御」 を参照してください。

21.2.5. 保留中のジョブの表示

保留中の **At** ジョブと **Batch** ジョブを表示するには、**atq** コマンドを実行します。**atq** コマンドは、1 行につき 1 ジョブが記述された保留中ジョブの一覧を表示します。各行は、ジョブ番号、日付、時間、ジョブクラス、ユーザー名の形式で記述されます。一般ユーザーが表示できるのは、自分のジョブのみです。root ユーザーが**atq** コマンドを実行すると、全ユーザーのすべてのジョブが表示されます。

21.2.6. 追加のコマンドラインオプション

at と **batch** には、以下のような追加のコマンドラインオプションがあります。

表21.1 at と batch のコマンドラインオプション

オプション	詳細
-f	コマンドやシェルスクリプトをプロンプトで指定するのではなく、ファイルから読み込む
-m	ジョブが完了した時点で、ユーザーに電子メールを送信する
-v	ジョブが実行される時刻を表示する

21.2.7. at と batch へのアクセスの制御

/etc/at.allow と **/etc/at.deny** のファイルを使用して、**at** コマンドと **batch** コマンドへのアクセスを制限できます。この両方のアクセス制御ファイルの形式は各行にユーザー名を 1 つ配置します。どちらのファイルでも空白は許可されません。

at.allow ファイルが存在する場合は、そのファイルに記載されているユーザーのみが **at** または **batch** の使用を許可され、**at.deny** ファイルは無視されます。

at.allow が存在しない場合は、**at.deny** に記載されているユーザーは **at** または **batch** を使用できません。

アクセス制御ファイルが変更された場合でも、**at** デーモン (**atd**) を再起動する必要はありません。アクセス制御ファイルは、ユーザーが **at** または **batch** のコマンドの実行を試みるたびに読み込まれます。

アクセス制御ファイルのコンテンツに関係なく、root ユーザーは常に **at** および **batch** のコマンドを実行することができます。

21.3. その他のリソース

自動化タスク設定についての詳しい情報は、以下のインストール済みのドキュメントを参照してください。

- **cron** man ページ — cron の概要を記載しています。
- **cron** man ページのセクション 1 および 5
  - セクション 1 には、**crontab** ファイルの概要が記載されています。
  - セクション 5 には、ファイル形式とサンプルエントリーが記載されています。
- **anacron** man ページ — anacron の概要を記載しています。
- **anacrontab** man ページ — **anacrontab** ファイルの概要を記載しています。
- **/usr/share/doc/at-*<version>*/timespec** には cron ジョブで指定できる時間の値について詳しく説明しています。
- **at** man ページ — **at** と**batch** およびそれらのコマンドラインオプションについて説明しています。

第22章 自動バグ報告ツール (ABRT)

通常 **ABRT** と短縮される **自動バグ報告ツール** は、**abrtd** というデーモンや検出された問題を処理、分析、報告する数多くのシステムサービスおよびユーティリティで構成されています。デーモンはほとんどの場合、バックグラウンドで静かに実行されていて、アプリケーションがクラッシュしたり、カーネル oops が検出されると迅速に作動します。そして、コアファイル (存在する場合) やクラッシュしたアプリケーションのコマンドラインパラメーター、フォレンシックユーティリティのその他のデータなど、関連する問題データを収集します。ABRT の最重要コンポーネントについての概要は、表22.1「基本的な ABRT コンポーネント」 を参照してください。

重要

Red Hat Enterprise Linux 6.2 では、自動バグ報告ツールがバージョン 2.0 にアップグレードされました。**ABRT** 2 シリーズでは、自動バグ検出/報告機能が大幅に改善されています。

表22.1 基本的な ABRT コンポーネント

コンポーネント	パッケージ	詳細
abrtd	abrt	root ユーザー下でバックグラウンドサービスとして実行される **ABRT** デーモン。
abrt-applet	abrt-gui	新しい問題発生時に **abrtd** からメッセージを受けてユーザーに通知するプログラム。
abrt-gui	abrt-gui	収集した問題データを表示して、ユーザーが処理できるようにする GUI アプリケーション。
abrt-cli	abrt-cli	GUI と同様の機能を提供するコマンドラインインターフェース。
abrt-ccpp	abrt-addon-ccpp	C/C++ 問題のアナライザーを提供する **ABRT** サービス。
abrt-oops	abrt-addon-kerneloops	カーネル oops のアナライザーを提供する **ABRT** サービス。
abrt-vmcore	abrt-addon-vmcore	カーネルのパニックアナライザーおよびレポーターを提供する **ABRT** サービス。

ABRT は現在、C/C++ および Python 言語で書かれたアプリケーションのクラッシュならびにカーネル oops の検出をサポートしています。Red Hat Enterprise Linux 6.3では、追加の abrt-addon-vmcore パッケージがインストールされていて **kdump** クラッシュダンピングメカニズムがそれに応じてシステム上で有効化、設定されていると、**ABRT** は カーネルパニックも検出できます。

ABRT は、リモートの問題トラッカーに問題を報告することができます。レポーティングは、問題が検出された時には常に自動的に報告するようするか、問題データをローカルに保管してユーザーが手動でレビュー/報告/削除を行うように設定することが可能です。 レポーティングツールは、問題データを Bugzilla データベースや Red Hat Technical Support (RHTSupport) サイトに送信できる他、**FTP**/**SCP** を使用したアップロード、電子メールでの送信、ファイルへの書き込みなどにも対応しています。

既存の問題データを処理する **ABRT** の部分は (新規問題データ作成などとは対照的に)、**libreport** という別のプロジェクトで対応することになりました。**libreport** ライブラリは問題の分析/報告のための包括的メカニズムを提供し、**ABRT** 以外のアプリケーションで使用されますが、**ABRT** と **libreport** の操作と設定は綿密に統合されているため、本ガイドでも取り上げています。

問題が検出されると、**ABRT** はその問題を既存の問題データと比較し、同じ問題が記録されているかどうかを確認します。記録されている場合には、その既存の問題データが更新され、最新の (重複している) 問題は再度記録されません。この問題が **ABRT** で認識されない場合は、**problem data directory (問題データディレクトリ)** が作成されます。問題データディレクトリは、通常、以下のようなファイルで構成されます:
`analyzer`、**`architecture`**、**`coredump`**、**`cmdline`**、**`executable`**、**`kernel`**、**`os_release`**、**`reason`**、**`time`**、**`uid`**。

分析中には、**`backtrace`** など、上記以外のファイルが作成される場合があります。これは、使用するアナライザーメソッドやその設定によって異なります。これらのファイルには、それぞれシステム及び問題自体についての固有の情報が格納されます。例えば、**`kernel`** ファイルには、クラッシュしたカーネルのバージョンが記録されます。

問題ディレクトリが作成され、問題データが収集された後、 **ABRT** GUI もしくは コマンドラインの **abrt-cli** ユーティリティを使ってその問題をさらに処理、分析、レポートすることができます。これらのツールについての詳細は、それぞれ 「グラフィカルユーザーインターフェイスの使用」 と 「コマンドラインインターフェイスの使用」 を参照してください。

注記

検出された問題のさらなる分析やレポートに **ABRT** を使わずに、レガシーの問題報告ツールである **report** を使って問題を報告している場合は、新たなバグの提出ができなくなっていることに注意してください。**report** ユーティリティは現在、RHTSupport または Bugzilla データベースにある既存バグを新たなコンテンツにアタッチすることにしか使えません。以下のコマンドで実行します。

```
report [[-v]] --target TARGET --ticket ID FILE
```

ここでは *TARGET* は RHTSupport にレポートする **strata** または Bugzilla レポートする **bugzilla** となります。*ID* は各データベースの既存問題ケースを識別する番号で、*FILE* はその問題ケースに追加する情報を含んでいるファイルとなります。

新たな問題をレポートする際に **abrt-cli** を使いたくない場合は、**report** の代わりに **report-cli** ユーティリティをつかうことができます。以下のコマンド発行で **report-cli** が問題報告プロセスをガイドしてくれます。

```
report-cli -r DUMP_DIRECTORY
```

ここでは *DUMP_DIRECTORY* は、**ABRT** または **`libreport`** を使用する他のアプリケーションが作成した問題データディレクトリです。**report-cli** に関する詳細情報は、**`man report-cli`** を参照してください。

22.1. ABRT のインストールとサービスの起動

ABRT を使用するには、まず最初に、root ユーザーとして以下のコマンドを実行し、**`abrt-desktop`** パッケージをシステムに確実にインストールします:

```
~]# yum install abrt-desktop
```

abrt-desktop をインストールした場合は、グラフィカルインターフェースのみで **ABRT** を使用することができます。**ABRT** をコマンドラインで使用する場合には、abrt-cli パッケージをインストールして下さい:

```
~]# yum install abrt-cli
```

Yum パッケージマネージャーでパッケージをインストールする方法については、「パッケージのインストール」 を参照してください。

次のステップは、**abrtd** が稼働していることを確認することです。このデーモンは通常、ブート時に起動するように設定されています。以下のコマンドを root ユーザーとして使用すると、このデーモンの現在の状態を確認できます:

```
~]# service abrtd status
abrtd (pid 1535) is running...
```

service コマンドは、デーモンが稼働していない場合には **abrt is stopped** (abrt は停止しています) と報告します。現行セッションでこのデーモンを起動するには、以下のコマンドを入力します:

```
~]# service abrtd start
Starting abrt daemon:                                      [  OK  ]
```

また、**ABRT** で C/C++ クラッシュを検出するには、上記と同じ手順に従って **abrt-ccpp** サービスを確認/開始することができます。**ABRT** がカーネル oops を検出するように設定するには、同じ手順を **abrt-oops** サービスに使用します。システム障害や、システムが反応しなくなったり、即座に再起動するような状態にするカーネル oops は、このサービスではキャッチできない点に注意して下さい。**ABRT** でそのようなカーネルを検出できるようにするには、**abrt-vmcore** サービスをインストールする必要があります。この機能が必要な場合は、詳細情報について 「カーネルパニックを検出する ABRT の設定」 を参照してください。

ABRT パッケージをインストールする際には、各 ABRT サービスは自動的に **runlevels 3 and 5** で有効化されます。ABRT サービスの無効化と有効化、有効時のランレベルの設定は **chkconfig** ユーティリティでできます。詳細情報については、「chkconfig ユーティリティの使用」 を参照してください。

警告

ABRT をインストールすると、コアダンプファイルの命名に使用するテンプレートを含む **/proc/sys/kernel/core_pattern** ファイルを上書きすることに注意してください。このファイルのコンテンツは、以下のように上書きされます。

```
|/usr/libexec/abrt-hook-ccpp %s %c %p %u %g %t e
```

最後に、グラフィカルデスクトップ環境で ABRT を実行している場合、以下の方法で **ABRT notification applet (通知アプレット)** が稼働していることを確認できます。

```
~]$ ps -el | grep abrt-applet
0 S   500  2036  1824  0  80   0 - 61604 poll_s ?        00:00:00 abrt-
applet
```

ABRT 通知アプレットが稼働していない場合には、以下のように **abrt-applet** プログラムを実行すると、現行のデスクトップセッションでアプレットを手動で起動することができます:

```
~]$ abrt-applet &
[1] 2261
```

このアプレットは、グラフィカルデスクトップセッションの開始時に自動的に起動するように設定することができます。トップパネルの**システム** → **設定** → **自動起動するアプリ** メニューを選択することで、**ABRT** 通知アプレットをプログラムの一覧に追加し、システムの起動時に実行されるように選択することができます。

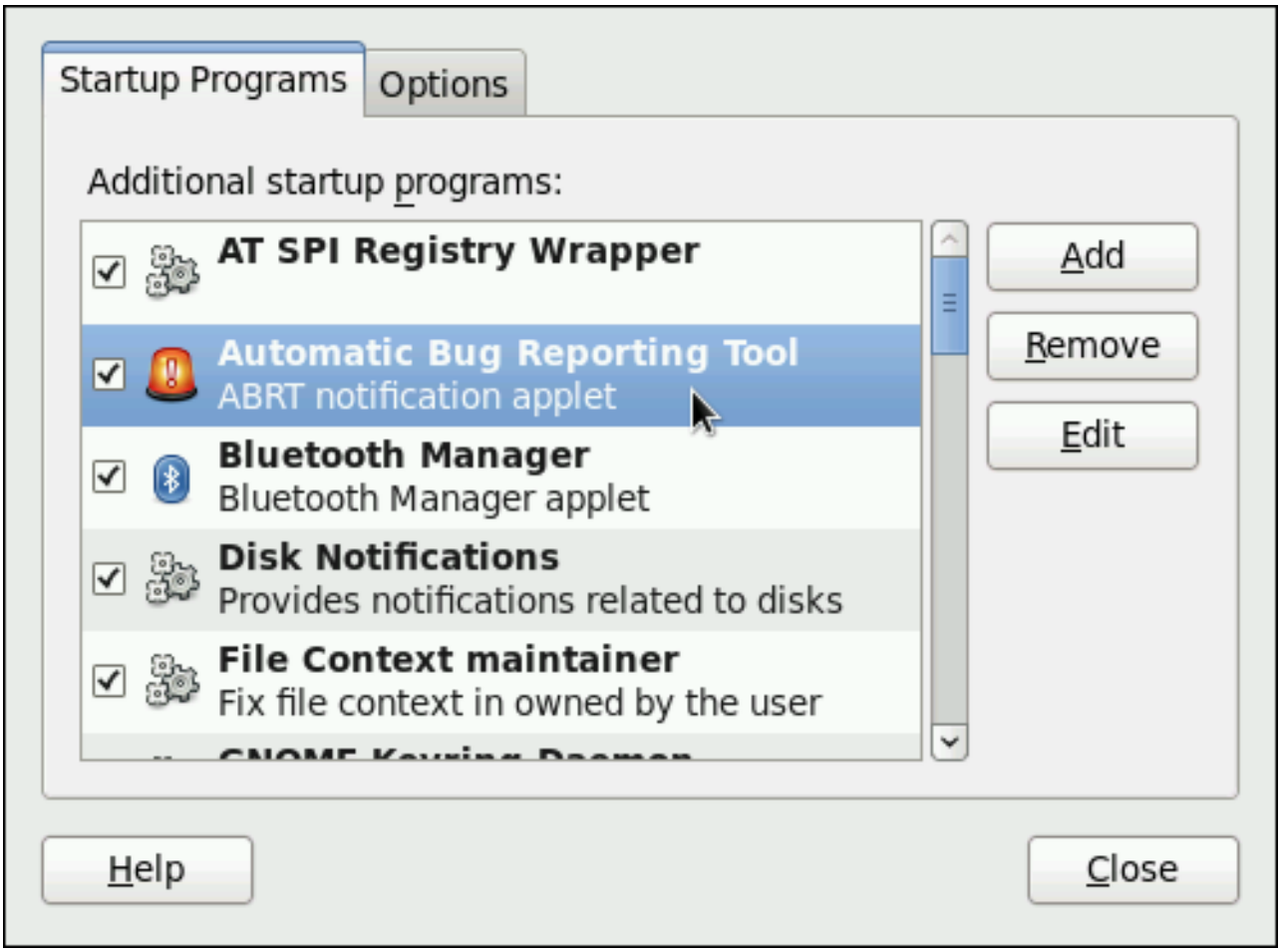


図22.1 ABRT 通知アプレットを自動的に実行するための設定

22.2. グラフィカルユーザーインターフェイスの使用

ABRT デーモンは、問題レポートの作成時に、毎回ブロードキャスト D-Bus メッセージを送信します。**ABRT** 通知アプレットが実行中の場合には、このメッセージをとらえて、通知スペースにオレンジ色の警告アイコンを表示します。このアイコンを使用して **ABRT GUI** アプリケーションを開くことができます。もしくは、**アプリケーション** → **システムツール** → **自動バグ報告ツール** のメニューアイテムを選択して、**ABRT** GUI を表示することもできます。

或いは、以下のようにコマンドラインから **ABRT** GUI を実行することもできます:

```
~]$ abrt-gui &
```

ABRT GUI は、検出された問題を確認/報告/削除する、簡単で直感的方法を提供します。**ABRT** ウィンドウには検出された問題の一覧が表示されます。各問題エントリは、障害が発生したアプリケーション名、アプリケーションがクラッシュした理由、その問題が最後に発生した日付によって構成されます。

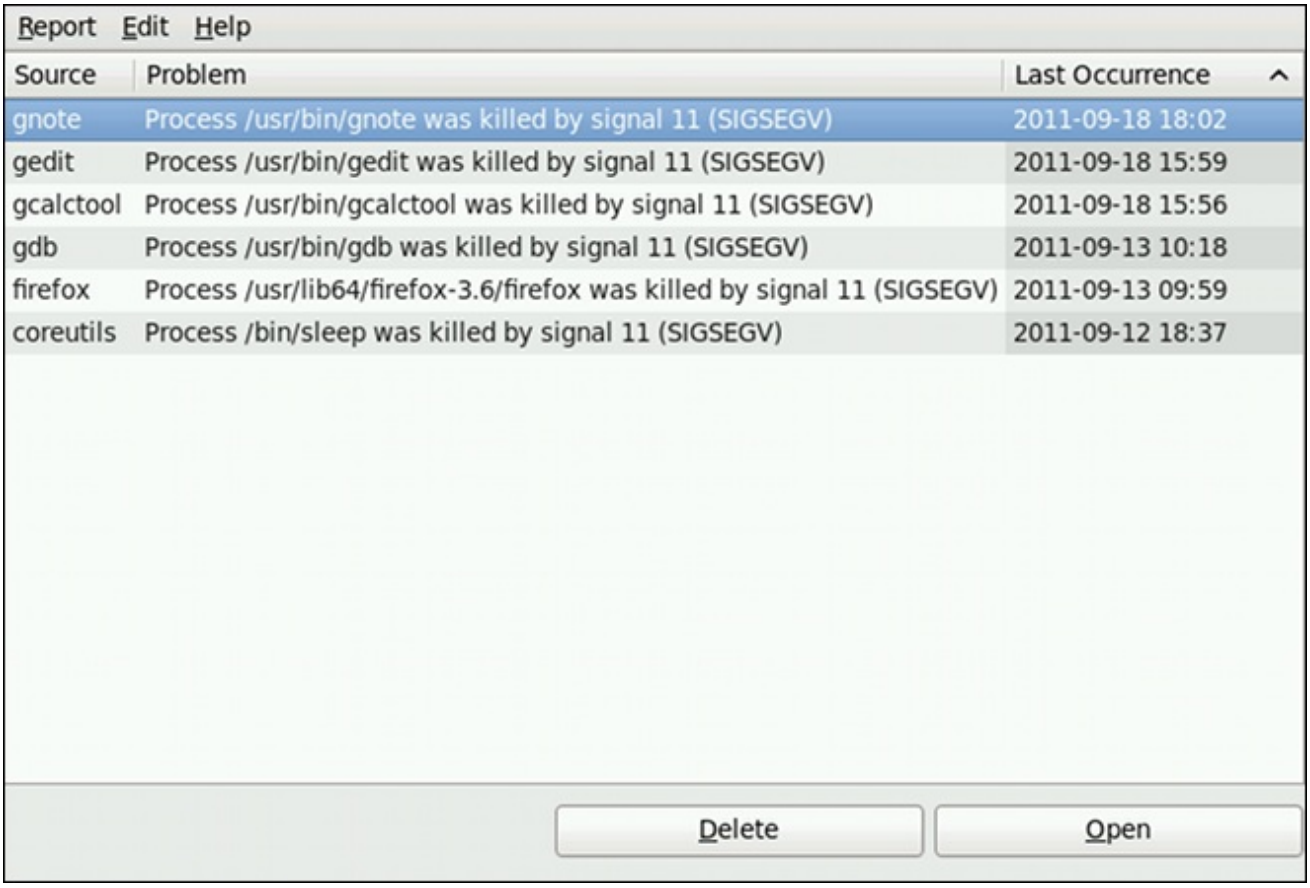


図22.2 ABRT GUI の実行例

問題報告の行をダブルクリックすると、問題の詳細情報にアクセスして、問題の分析方法と報告先を決定することができます。

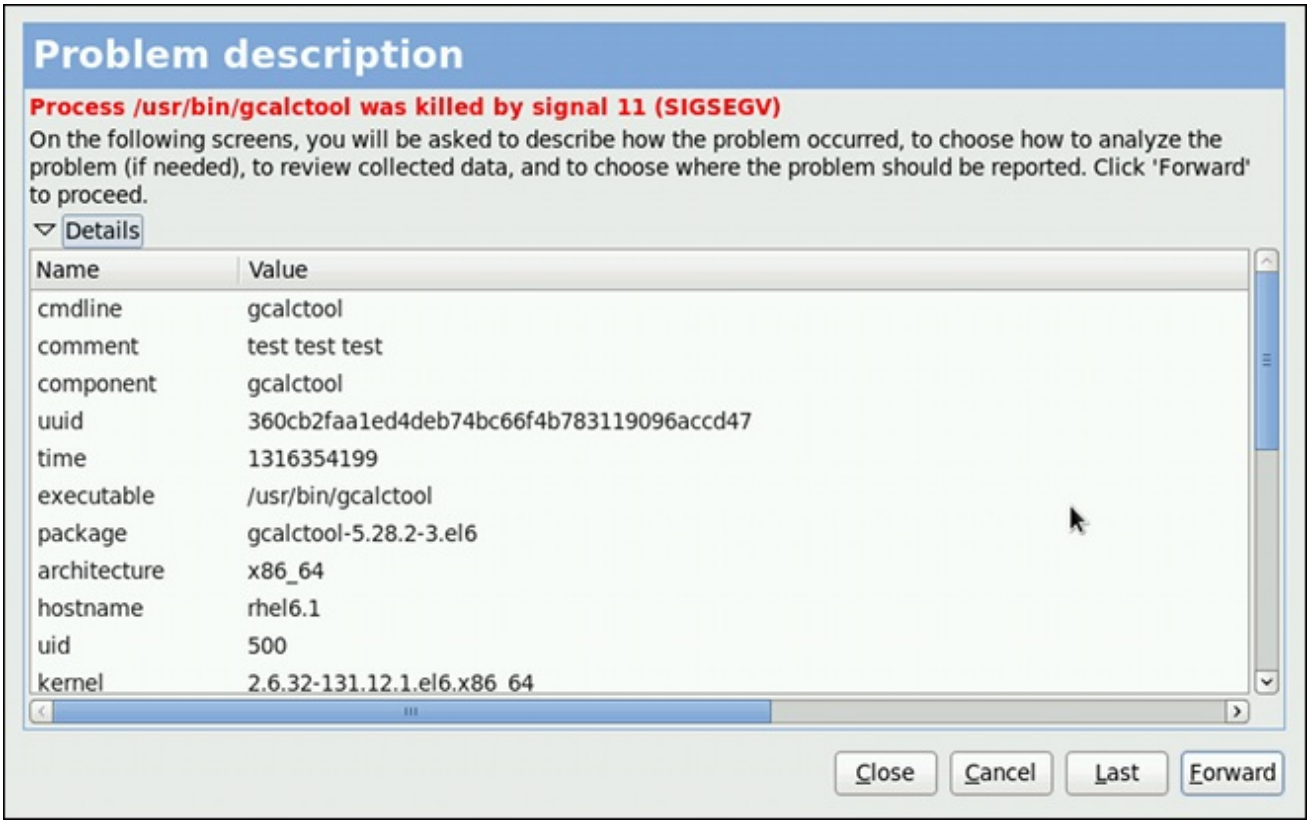


図22.3 詳細な問題データの例

まず最初に、発生した問題についての追加情報を提供するように要求されます。問題がどのように発生したか、どのような手順でその問題を再現することができるかについての情報を提供する必要があります。次のステップでは、問題の分析方法を選択し、設定に応じてバックトレースを生成します。分析とバックトレース生成を省略することもできますが、開発者は問題について可能な限り多くの情報を必要とすることを念頭に置いて下さい。バックトレースは常時修正可能なので、問題データを送信する前には、提供したくない機密情報を削除することができます。

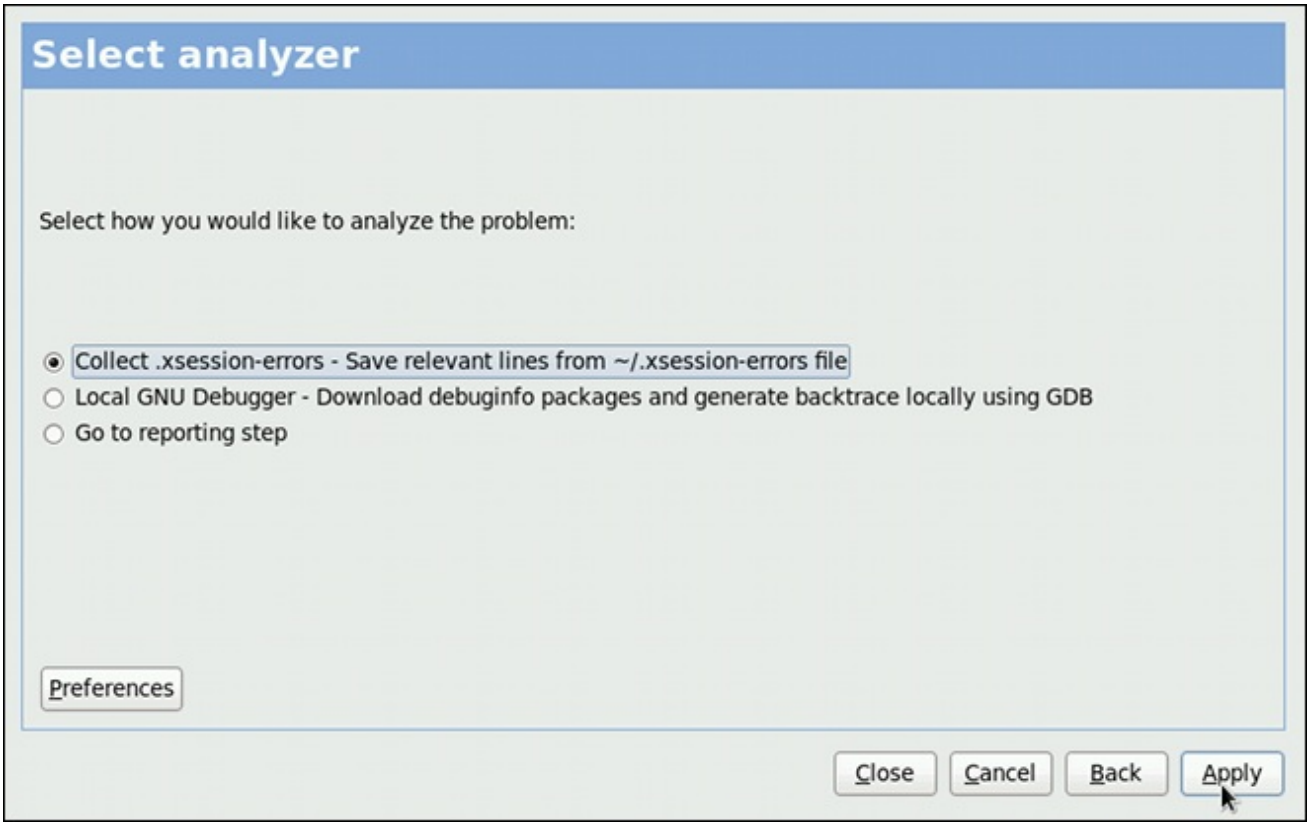


図22.4 問題分析方法の選択

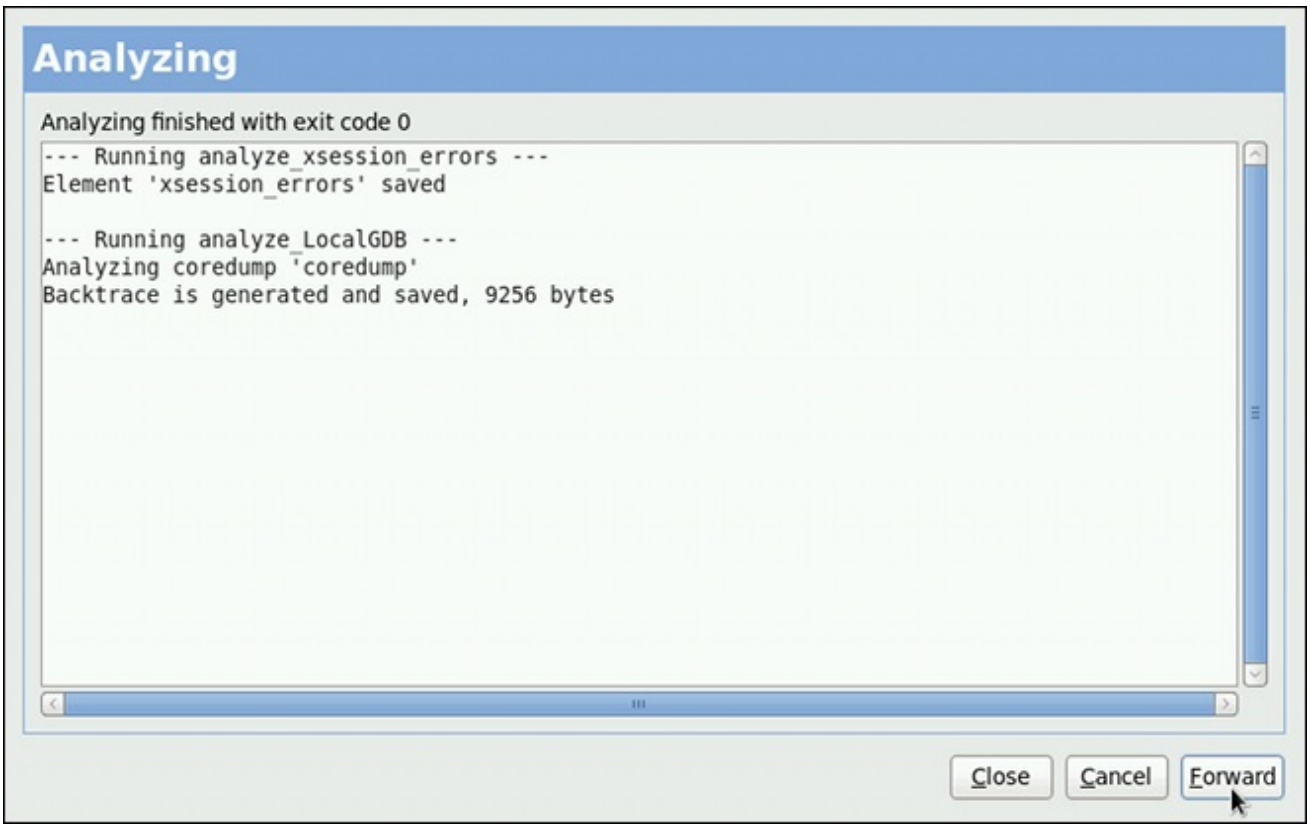


図22.5 問題分析中の ABRT

次に、問題を報告する方法を選択します。Red Hat Enterprise Linux をご使用の場合には、Red Hat カスタマーサポートを選択することを推奨します。

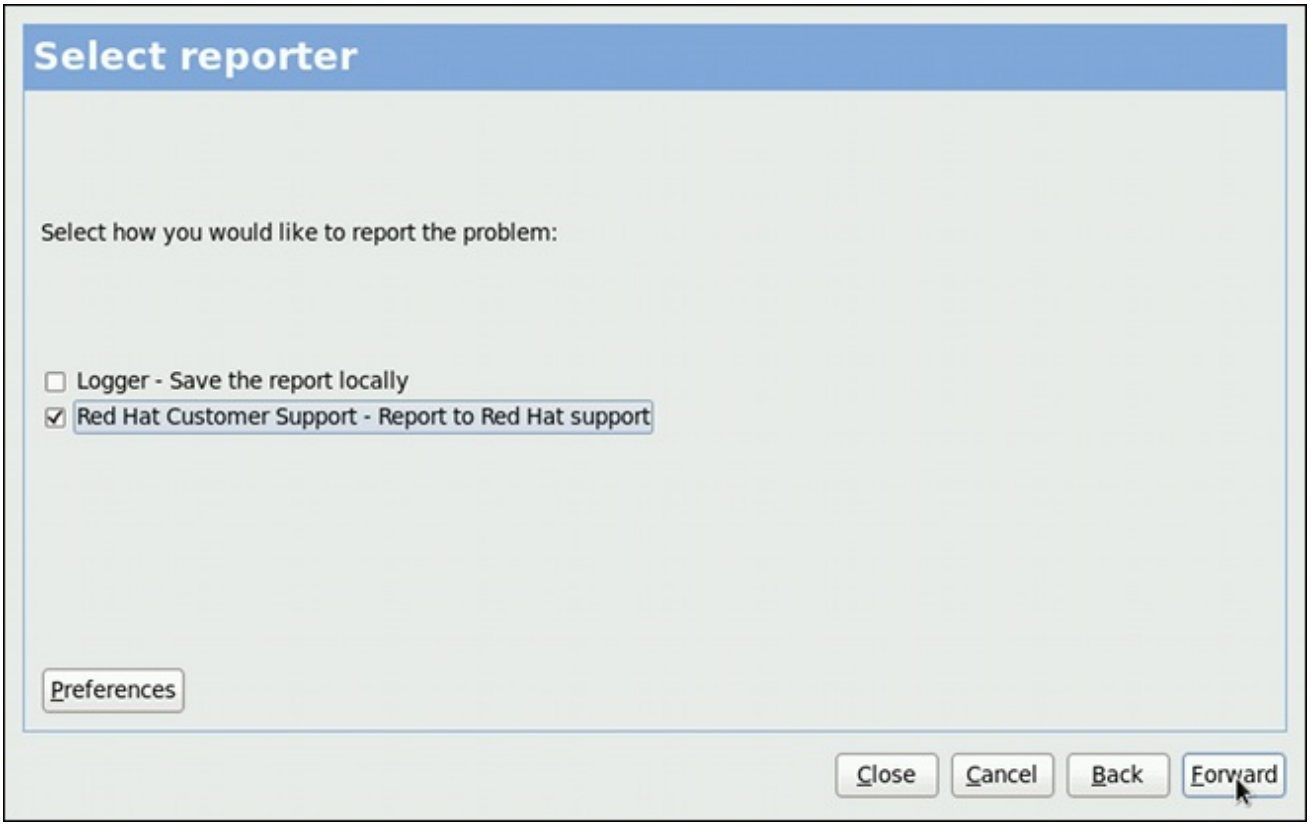


図22.6 問題レポーターの選択

Red Hat カスタマーサポートを報告先に選択した際に、このイベントが未設定の場合には、このイベントが適切に設定されていないという警告が表示され、設定を行うためのオプションが提供されます。

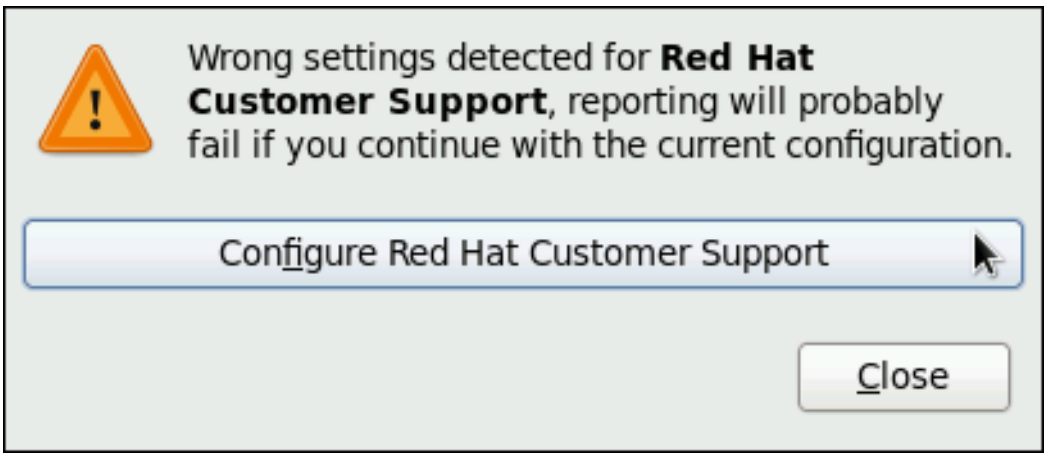


図22.7 警告 - Red Hat カスタマーサポートが未設定の場合

ここでは、Red Hat ログイン情報 (取得方法及びイベント設定に関する詳細は 「ABRT GUI でのイベント設定」 を参照) を提供する必要があります。適切な情報を提供しないと、問題の報告は失敗してしまいます。

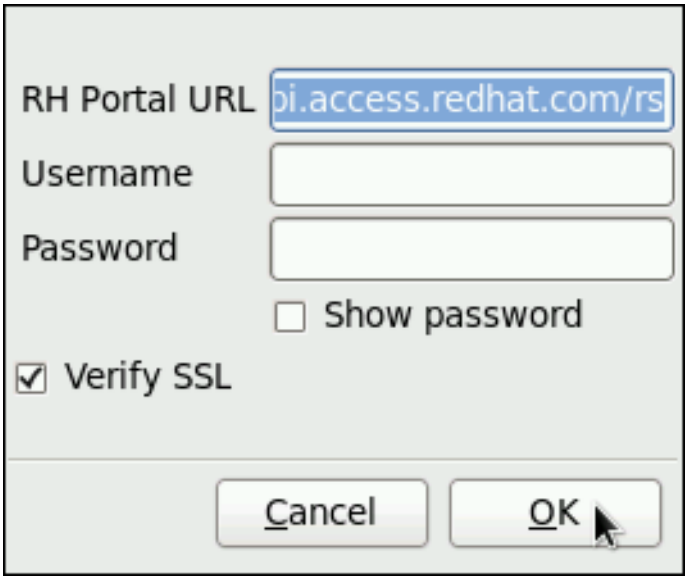


図22.8 Red Hat カスタマーサポートの設定ウィンドウ

報告の方法を選択し、正しく設定を済ませたら、バックトレースを再確認して、報告するデータを確定します。

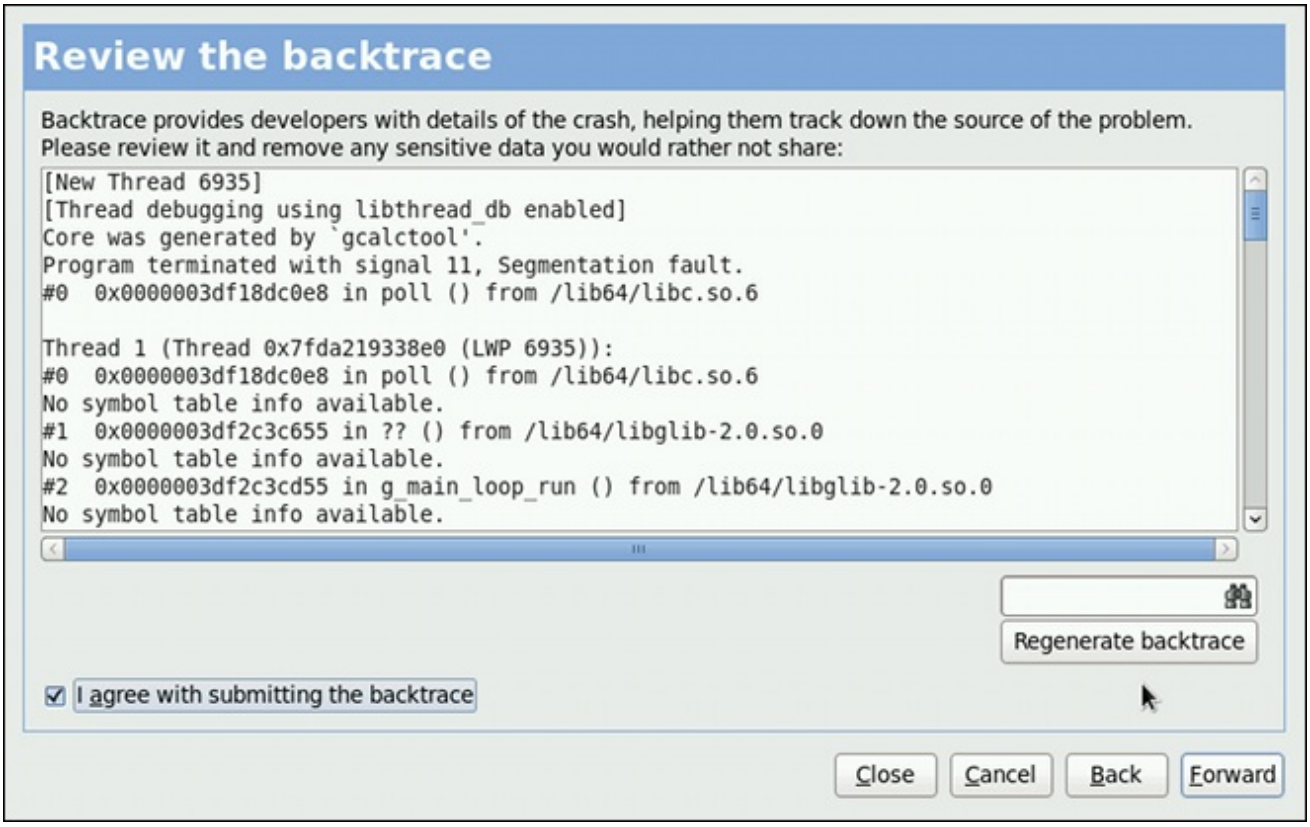


図22.9 問題バックトレースの再確認

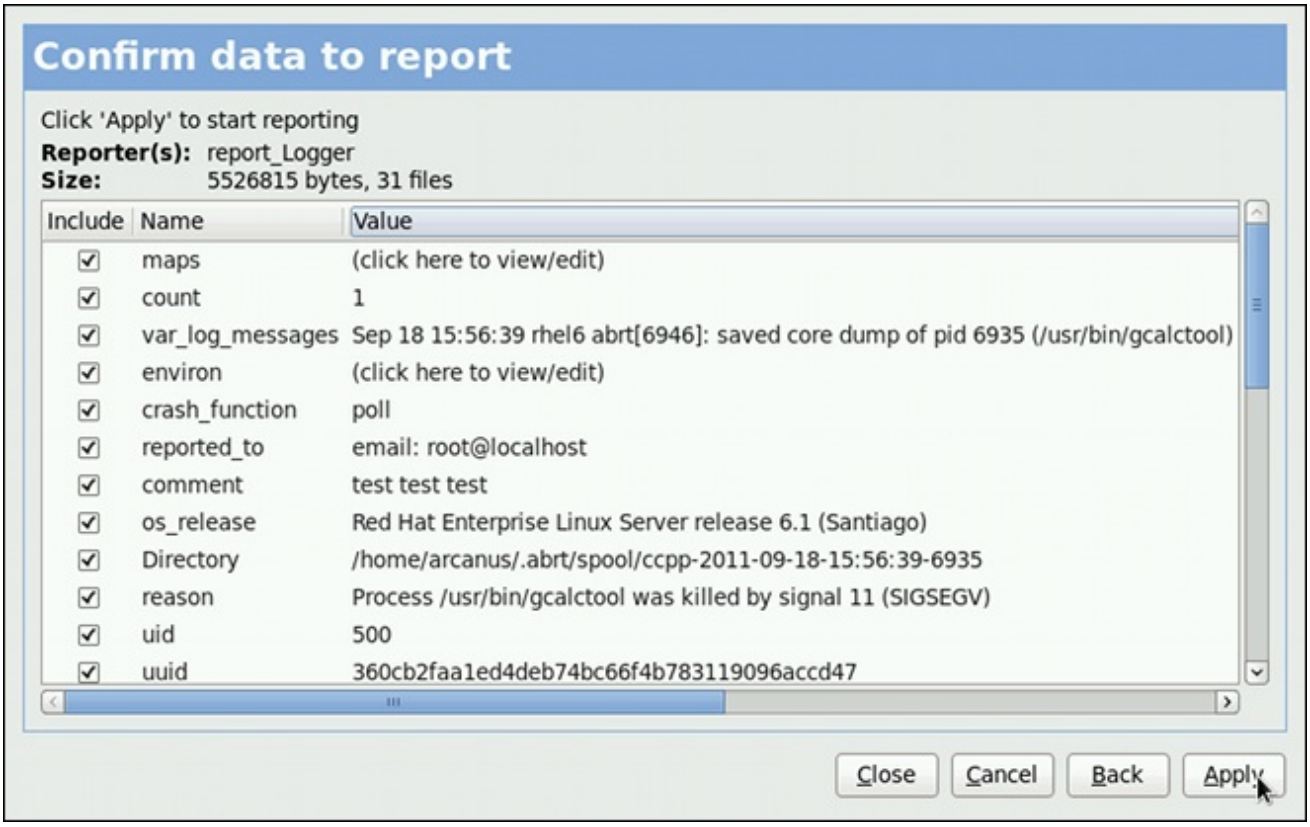


図22.10 報告するデータの確認

最後に、問題データが選択した報告先に送信されたら、別の方法で問題の報告を継続するか、問題の処理を終了するかを選択することができます。Red Hat カスタマーサポートデータベースに報告した場合には、問題ケースがデータベースに提出されます。この時点から、報告プロセス中に提供した問題の解決の進捗状況が電子メールで通知されるようになります。又、問題ケースが作成された際に **ABRT** GUI で提供された URL や Red Hat サポートからの電子メールで問題ケースを確認することもできます。

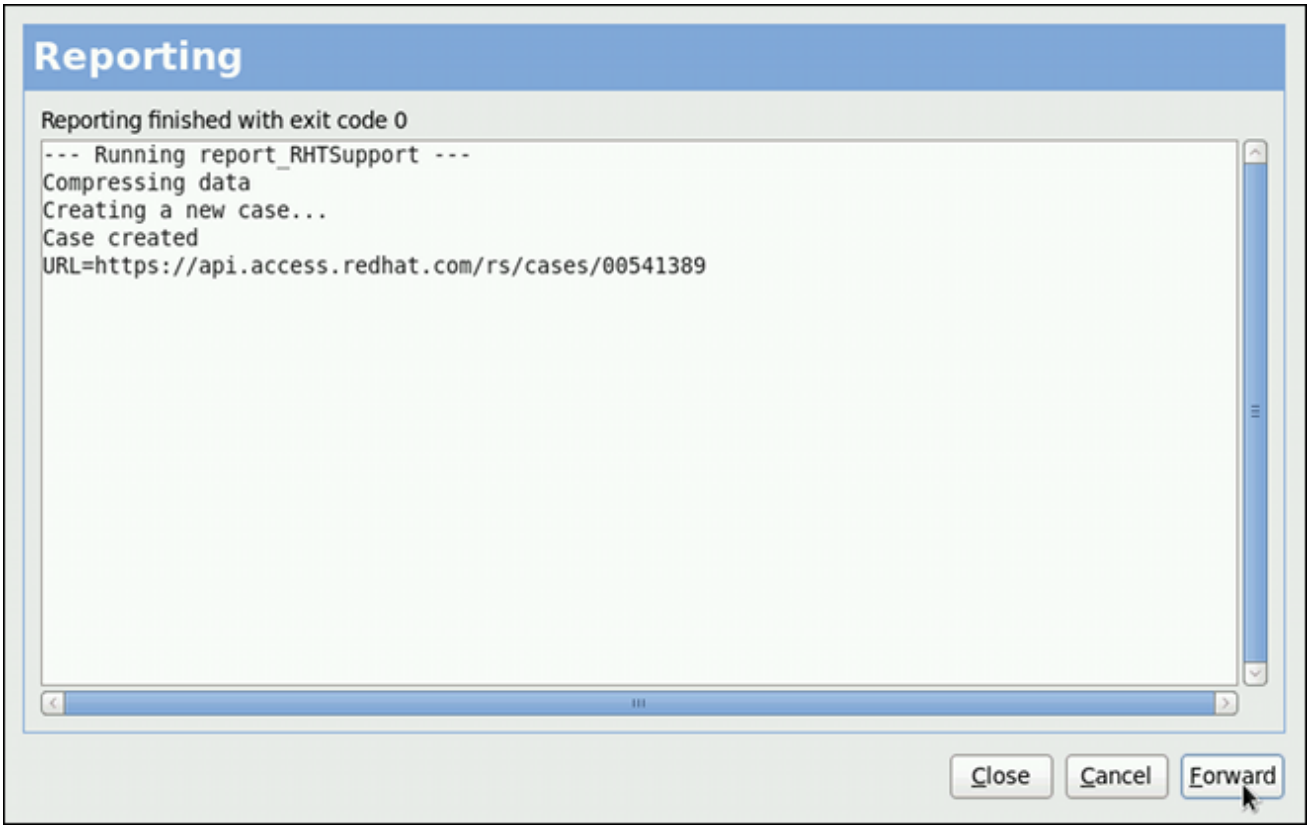


図22.11 Red Hat カスタマーサービスデータベースに問題を報告中

22.3. コマンドラインインターフェイスの使用

abrtd で保存された問題データは、コマンドラインインターフェイスを使用して表示/報告/削除することができます。

以下の構文は、**abrt-cli** ツールの一般的な使用方法です:

```
abrt-cli [--version] <command> [<args>]
```

...ここで *<args>* は、問題データディレクトリ及び/またはコマンドを修正するオプション、*<command>* は、以下にあげるサブコマンドのいずれか一つです:

- **list** — 問題を一覧表示して、問題データを表示します。
- **report** — 問題の分析と報告を行います。
- **rm** — 不要な問題を削除します。
- **info** — 特定の問題に関する情報を提供します。

特定の **abrt-cli** コマンドの使用方法に関するヘルプを表示するには、以下のコマンドを入力します:

```
abrt-cli <command> --help
```

abrt-cli に使用するその他のコマンドについては、以下のセクションで説明します。

22.3.1. 問題の表示

検出された問題を表示するには、**abrt-cli list** コマンドを入力します:

```
~]# abrt-cli list
Directory:      /var/spool/abrt/ccpp-2011-09-13-10:18:14-2895
count:          2
executable:     /usr/bin/gdb
package:        gdb-7.2-48.el6
time:           Tue 13 Sep 2011 10:18:14 AM CEST
uid:            500

Directory:      /var/spool/abrt/ccpp-2011-09-21-18:18:07-2841
count:          1
executable:     /bin/bash
package:        bash-4.1.2-8.el6
time:           Wed 21 Sep 2011 06:18:07 PM CEST
uid:            500
```

- **Directory** — 問題に関する全情報を格納する問題データディレクトリを示します。
- **count** — この特定の問題が発生した回数を示します。
- **executable** — クラッシュしたバイナリまたは実行可能なスクリプトを示します。
- **package** — 問題の原因となったプログラムが格納されているパッケージの名前を示します。
- **time** — 問題が最後に発生した日付と時刻を示します。
- **uid** — クラッシュしたプログラムを実行していたユーザー ID を示します。

以下の表は、**abrt-cli list** コマンドで使用できるオプションをまとめたものです。オプションはすべて相互包括的なので、必要に応じて組み合わせることができます。コマンドの出力は、全オプションを組み合わせた場合には最も包括的となり、追加のオプションを使用しない場合には最小限の情報が表示されます。

表22.2 abrt-cli list コマンドのオプション

オプション	詳細
	abrt-cli list コマンドで追加のオプションを使用しないと、未報告の問題に関する基本的な情報のみが表示されます。
-d, **--detailed**	一覧表示されている問題に関した、格納されている全情報が表示されます。生成済みの場合には **backtrace** も含まれます。
-f, **--full**	報告済みの問題を含む、全問題に関する基本的な情報が表示されます。
-v, **--verbose**	操作に関する追加情報を提供します。

特定の問題に関する情報のみを表示するには、以下のコマンドを使用します:

```
abrt-cli info <directory>
```

ここで *<directory>* は、表示する問題の **問題データディレクトリ** です。以下の表は、**abrt-cli info**

コマンドで使用できるオプションをまとめたものです。オプションはすべて相互包括的なので、必要に応じて組み合わせることができます。コマンドの出力は、全オプションを組み合わせた場合には最も包括的となり、追加のオプションを使用しない場合には最小限の情報が表示されます。

表22.3 `abrt-cli info` コマンドのオプション

オプション	詳細
	`abrt-cli info` コマンドで追加のオプションを使用しない場合は、**問題データディレクトリ** 引数で指定した問題に関する基本的な情報のみが表示されます。
`-d`, **`--detailed`**	**問題データディレクトリ** 引数で指定した問題に関した、格納されているすべての情報が表示されます。生成済みの場合には **`backtrace`** も含まれます。
`-v`, **`--verbose`**	**`abrt-cli info`** の操作に関する追加情報を提供します。

22.3.2. 問題の報告

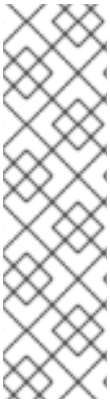
特定の問題を報告するには、以下のコマンドを使用します:

```
abrt-cli report <directory>
```

ここで *<directory>* は、報告する問題の **問題データディレクトリ** です。以下が例となります:

```
~]$ abrt-cli report /var/spool/abrt/ccpp-2011-09-13-10:18:14-2895
How you would like to analyze the problem?
1) Collect .xsession-errors
2) Local GNU Debugger
Select analyzer: _
```

ABRT は、報告する問題のアナライザーイベントの選択を要求します。イベントを選択した後に、問題が分析されます。これにはかなり時間がかかる場合があります。 問題を報告する準備が整うと、**`abrt-cli`** によりテキストエディターが開かれ、レポートの内容が表示されます。報告される内容を確認し、クラッシュを再現する手順及びその他のコメントを記入します。又、バックトレースも確認しておくべきです。バックトレースは、レポーターイベントの設定によっては、パブリックサーバーに送信されて、誰でも閲覧できることになる可能性があるためです。



注記

レポートのチェックには、任意のテキストエディターを選択することができます。**`abrt-cli`** では、**`ABRT_EDITOR`** 環境変数で定義されたエディターが使用されます。変数が定義されていない場合には、**`VISUAL`** 及び **`EDITOR`** の変数を確認します。これらのいずれも設定されていない場合には、**`vi`** が使用されます。**`.bashrc`** 設定ファイルで、任意のエディターを設定することができます。例えば、GNU Emacs を使用する場合には、このファイルに以下の行を追記して下さい:

```
export VISUAL=emacs
```

レポートの編集が完了したら、変更を保存して、エディターと終了します。レポートの送信に使用する設定済み **ABRT** レポーターの選択を求められます。

```
How would you like to report the problem?
1) Logger
2) Red Hat Customer Support
Select reporter(s): _
```

報告の方法を選択したら、次にそのレポートと共に送信されるデータの再確認に進みます。以下の表には、**abrt-cli report** コマンドで使用できるオプションをまとめています。

表22.4 abrt-cli report コマンドのオプション

オプション	詳細
	追加のオプションを使用しない場合には、**abrt-cli report** コマンドは通常の出力を表示します。
-v, **--verbose**	**abrt-cli report** の操作に関する追加情報を提供します。

22.3.3. 問題の削除

特定の問題を報告したくないことが確実に分かっている場合には、その問題を削除することができます。問題を削除して、その情報が**ABRT** に保存されないようにするには、以下のコマンドを使用します:

```
abrt-cli rm <directory>
```

ここで *<directory>* は削除される問題の問題データディレクトリです。以下が例となります:

```
~]$ abrt-cli rm /var/spool/abrt/ccpp-2011-09-12-18:37:24-4413
rm '/var/spool/abrt/ccpp-2011-09-12-18:37:24-4413'
```

注記

ABRT は、新たな問題をローカルに保存されている全問題と比較することにより、問題の重複を検出する点に注意して下さい。繰り返し発生しているクラッシュの場合、**ABRT** が対応を要求するのは一回のみですが、その問題のクラッシュダンプを削除してしまうと、その特定の問題が次回発生した際、**ABRT** はその問題を新規クラッシュとして処理することになります。このため、**ABRT** は警告を表示し、その問題の詳細を記入して報告するように要求します。**ABRT** が繰り返し発生する問題を通知しないようにするには、その問題データは削除しないで下さい。

以下の表は、**abrt-cli rm** コマンドで使用できるオプションをまとめたものです。

表22.5 abrt-cli rm コマンドのオプション

オプション	詳細

オプション	詳細
	追加オプションを使用しない場合、**abrt-cli rm** コマンドは、指定した**問題データディレクトリ** とその全コンテンツを削除します。
-v, **--verbose**	**abrt-cli rm** の操作に関する追加情報を提供します。

22.4. ABRT の設定

問題 ライフサイクルは、**ABRT** 内の **イベント** によって決定されます。以下が例となります:

- イベント 1 — 問題データディレクトリの作成
- イベント 2 — 問題データの分析
- イベント 3 — 問題を Bugzilla に報告

問題が検出されると、その問題を定義するデータが保存され、問題のデータディレクトリに対してイベントを実行することにより、問題が処理されます。イベント及びその定義方法についてに詳しい情報は、「ABRT イベント」 を参照して下さい。**ABRT** の標準インストールは、現在いくつかのデフォルトイベントをサポートしています。これらは、問題報告の過程で選択して使用することができます。これらのイベントの一覧は、「ABRT の標準インストールでサポートされているイベント」 を参照して下さい。

ABRT 及び **libreport** をインストールすると、それぞれの設定ファイルがシステム上の複数のディレクトリに配置されます:

- **/etc/libreport/** — **report_event.conf** という主要設定ファイルが格納されます。この設定ファイルに関する詳しい情報は、「ABRT イベント」 を参照して下さい。
- **/etc/libreport/events/** — 事前定義されたイベントのデフォルト設定を指定するファイルが格納されます。
- **/etc/libreport/events.d/** — イベントを定義する設定ファイルが格納されます。
- **/etc/libreport/plugins/** — イベントに関与するプログラムの設定ファイルが格納されます。
- **/etc/abrt/** — **ABRT** のサービス及びプログラムの動作を修正するのに使用する、**ABRT** 固有の設定が格納されます。特定の設定ファイルに関する詳しい説明は、「ABRT 固有の設定」 に記載しています。
- **/etc/abrt/plugins/** — **ABRT** のサービス及びプログラムのデフォルト設定をオーバーライドするのに使用する設定ファイルが格納されています。特定の設定ファイルに関する詳しい説明は、「ABRT 固有の設定」 を参照して下さい。

22.4.1. ABRT イベント

各イベントは、それぞれの設定ファイル内の単一のルール構造によって定義されます。これらの設定ファイルは、通常 **/etc/libreport/events.d/** ディレクトリに格納されます。これらの設定ファイルは、主要設定ファイル **/etc/libreport/report_event.conf** によって使用されます。

/etc/libreport/report_event.conf ファイルは、**include 指示文** と **ルール** で構成されます。

ルールは通常、**/etc/libreport/events.d/** ディレクトリ内の別の設定ファイルに格納されています。標準のインストールでは、**/etc/libreport/report_event.conf** ファイルに記載されている include 指示文は一つのみです:

```
include events.d/*.conf
```

このファイルを変更する場合には、シェルのメタ文字 (*、$、? など) が優先され、その位置に対して相対パスが解釈される点に注意して下さい。

各 **ルール** の行頭には空白文字を入れない行で開始します。その後に続く、**空白** 文字または **タブ** 文字で始まる行はすべて、このルールの一部とみなされます。各 **ルール** は、**条件** 部と **プログラム** 部の 2 部で構成されます。条件部には、以下のような形式で条件が記載されます:

- *VAR*=*VAL*,
- *VAR*!=*VAL*, or
- *VAL*~=*REGEX*

…ここで:

- *VAR* は、**EVENT** のキーワードもしくは問題データディレクトリエレメントの名前です (例: **executable**、**package**、**hostname** など)。
- *VAL* は、イベントまたは問題データエレメントの名前です。
- *REGEX* は正規表現です。

プログラム部は、プログラム名とシェルが解釈できるコードによって構成されます。条件の部分にある全条件が有効な場合、プログラム部がシェルで実行されます。以下は **イベント** の一例です:

```
EVENT=post-create date > /tmp/dt
        echo $HOSTNAME `uname -r`
```

このイベントは、現在の日付と時刻で **/tmp/dt** ファイルの内容を上書きし、マシンのホスト名とカーネルのバージョンを標準出力に表示します。

更に複雑なイベント (実際に、事前設定済みイベントの一つである場合) の例は以下のようになります。このイベントは、**abrt-ccpp** サービスを使用して処理された問題の問題レポートに **~/.xsession-errors** ファイルの関連する行を保存します。クラッシュしたアプリケーションは、クラッシュが発生した時点で、いずれかの X11 ライブラリをロードしていました:

```
EVENT=analyze_xsession_errors analyzer=CCpp dso_list~=.*/libX11.*
        test -f ~/.xsession-errors || { echo "No ~/.xsession-errors";
exit 1; }
        test -r ~/.xsession-errors || { echo "Can't read ~/.xsession-
errors"; exit 1; }
        executable=`cat executable` &&
        base_executable=${executable##*/} &&
        grep -F -e "$base_executable" ~/.xsession-errors | tail -999
>xsession_errors &&
        echo "Element 'xsession_errors' saved"
```

設定可能なイベントのセットは、固定されてはいません。システム管理者は、必要に応じて、イベントを追加することができます。現在、**ABRT** と **libreport** の標準インストールでは、以下のようなイベント名が提供されています:

`post-create`

このイベントは、**`abrtd`** により、新規作成された問題データディレクトリに対して実行されます。**`post-create`** イベントが実行されると、**`abrtd`** は、新規問題データの UUID が既存の問題ディレクトリの UUID と一致するかどうかを確認します。そのような問題ディレクトリが存在する場合には、新規問題データは削除されます。

`analyze_<NAME_SUFFIX>`

…ここで、*<NAME_SUFFIX>* はイベント名の調整可能な部分です。このイベントは、データ収集の処理に使用されます。例えば、**`analyze_LocalGDB`** が GNU Debugger (**GDB**) ユーティリティをアプリケーションのコアダンプに対して実行し、プログラムのバックトレースを生成します。**abrt-gui** を使用すると、分析イベントの一覧を表示して選択することができます。

`collect_<NAME_SUFFIX>`

…ここで、*<NAME_SUFFIX>* はイベント名の調整可能な部分です。このイベントは、問題に関する追加情報を収集するのに使用されます。**abrt-gui** を使用すると、収集イベントの一覧を表示して選択することができます。

`report_<NAME_SUFFIX>`

…ここで、*<NAME_SUFFIX>* は、イベント名の調整可能な部分です。このイベントは、問題の報告に使用されます。**abrt-gui** を使用すると、報告イベントの一覧を表示して、選択することができます。

イベントに関する追加情報 (例: 環境変数としてイベントに渡すことができるパラメーターの説明、名前、タイプ、その他の属性など) は、**`/etc/libreport/events/<event_name>.xml`** ファイルに格納されます。これらのファイルは、**abrt-gui** 及び **abrt-cli** でユーザーインターフェースをより使いやすくするために使用されます。標準のインストールを変更したくない場合には、このファイルは編集しないで下さい。

22.4.2. ABRT の標準インストールでサポートされているイベント

ABRT の標準インストールは、現在数多くの分析/収集/報告イベントを提供しています。これらの一部は、**ABRT** GUI アプリケーションを使用して設定することもできます (**ABRT** GUI を使用したイベント設定に関する詳しい説明は、「ABRT GUI でのイベント設定」 を参照)。**ABRT** GUI では、ユーザーが読みやすいように、完全なイベント名ではなく、イベント名の一意の部分のみが表示されます。例えば、**`analyze_xsession_errors`** イベントは、**ABRT** GUI では **`Collect .xsession-errors`** と表示されます。**ABRT** の標準インストールで提供されるデフォルトの分析/収集/報告イベントの一覧は以下の通りです:

analyze_VMcore — VM コアの分析core

アプリケーションの問題データに対して、**GDB** (GNU デバッガ) を実行し、カーネルの **`backtrace`** を生成します。このイベントは、**`/etc/libreport/events.d/vmcore_event.conf`** 設定ファイルで定義されています。

analyze_LocalGDB — Local GNU Debugger (ローカル GNU デバッガ)

アプリケーションの問題データに対して、**GDB** (GNU デバッガ) を実行し、プログラムの **`backtrace`** を生成します。このイベントは、**`/etc/libreport/events.d/ccpp_event.conf`** 設定ファイルで定義されています。

analyze_xsession_errors — Collect .xsession-errors

~/.xsession-errors ファイルから関連性のある行を問題レポートに保存します。このイベントは、**/etc/libreport/events.d/ccpp_event.conf** 設定ファイルで定義されています。

report_Logger — Logger (ロガー)

問題レポートを作成し、指定のローカルファイルに保存します。このイベントは、**/etc/libreport/events.d/print_event.conf** 設定ファイルで定義されています。

report_RHTSupport — Red Hat Customer Support (Red Hat カスタマーサポート)

問題を Red Hat テクニカルサポートシステムに報告します。このオプションは、Red Hat Enterprise Linux のユーザーが対象です。このイベントは、**/etc/libreport/events.d/rhtsupport_event.conf** 設定ファイルで定義されています。

report_Mailx — Mailx

Mailx ユーティリティを使用して、レポートを指定の電子メールアドレスに送信します。このイベントは、**/etc/libreport/events.d/mailx_event.conf** 設定ファイルで定義されています。

report_Kerneloops — Kerneloops.org

カーネルの問題を oops トラッカーに送信します。このイベントは、**/etc/libreport/events.d/koops_event.conf** 設定ファイルで定義されています。

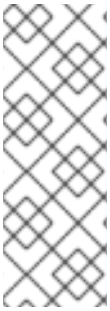
report_Uploader — Report uploader (レポートアップローダー)

FTP または **SCP** プロトコルを使用して、tarball (.tar.gz) アーカイブ及び問題データを、選択したデスティネーションにアップロードします。このイベントは、**/etc/libreport/events.d/uploader_event.conf** 設定ファイルで定義されています。

22.4.3. ABRT GUI でのイベント設定

イベントは、環境変数として渡されたパラメーターを使用することができます (例えば、**report_Logger** イベントは、出力名をパラメーターとして受け入れます)。それぞれの**/etc/libreport/events/<*event_name*>.xml** ファイルを使用して、 **ABRT** GUI は、選択されたイベントに対して、どのパラメーターを指定することができるかを判断した上で、ユーザーがそれらのパラメーターの値を設定するのを許可します。 設定された値は、**ABRT** GUI によって保存され、イベントが後で呼び出される際に再利用されます。

Edit (編集) → **Preferences (設定)** の順にクリックして、**Event Configuration** ウィンドウを開きます。このウィンドウには、使用可能な全イベントの一覧が表示され、報告過程で選択することができます。設定可能なイベントを一つ選択すると、**Configure Event** ボタンがクリックできる状態となり、そのイベントの設定値を設定することができます。いずれかのイベントパラメーターを変更した場合には、**Gnome** キーリングに保存され、それ以降の GUI セッションで使用されます。



注記

/etc/libreport/ ディレクトリ階層内のファイルはすべて、全ユーザーに読み取り権限があり、グローバル設定として使用するためにあります。このため、ユーザー名、パスワード、その他の機密データは、この中には保管しないことを推奨します。ユーザー別の設定 (GUI アプリケーションで設定され、**$HOME** の所有者のみが読み取り可能) は、**Gnome** キーリングに保管されます。又、**abrt-cli** で使用するためには、**$HOME/.abrt/*.conf** 内のテキストファイルに保管することもできます。

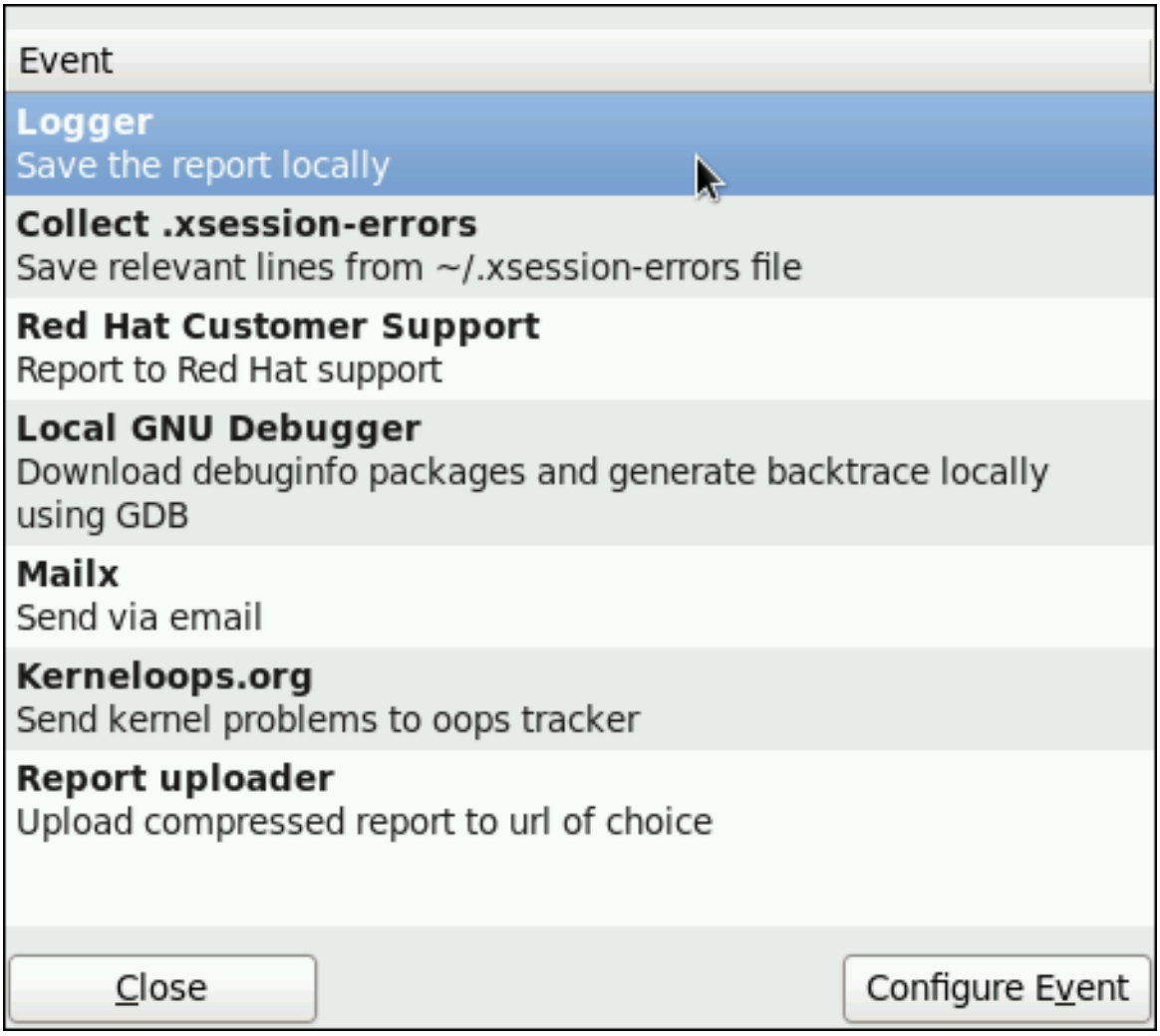


図22.12 Event 設定ウィンドウ

以下は、**ABRT** GUI アプリケーションで設定可能な各事前定義済みイベントに使用できる全設定オプションの一覧です。

ロガー

ロガー イベントの設定ウィンドウでは、以下のようなパラメーターを設定することができます:

- **`Log file (ファイルをログする)`** — クラッシュレポートの保存先となるファイルを指定します (デフォルトでは **`/var/log/abrt.log`** に設定されています)。

`Append (追記する)` オプションにチェックマークが付いている場合には、ロガーイベントは、新規クラッシュレポートを **ロガーファイル** オプションで指定したログファイルに追記します。チェックマークが付いていない場合には、以前のレポートは、常に新規クラッシュレポートに置き換えられます。

Red Hat カスタマーサポート

`Red Hat カスタマーサポート` のイベント設定ウィンドウでは、以下のパラメーターを設定することができます:

- **`RH Portal URL`** — クラッシュダンプの送信先となる、Red Hat カスタマーサービスの URL を指定します (デフォルトでは、https://api.access.redhat.com/rs に設定されています)。

- **`Username (ユーザー名)`** — Red Hat カスタマーサポートへのログインと 報告するクラッ

シュ用の Red Hat カスタマーサポートデータベースエントリを作成する際に使用されるユーザーログイン。http://www.redhat.com/、Red Hat カスタマーポータル (https://access.redhat.com/home)、または Red Hat Network (https://rhn.redhat.com/) でアカウントを作成すると取得できる **Red Hat ログイン** を使用して下さい。

- **Password (パスワード)** — Red Hat カスタマーサポートへのログインに使用するパスワード (**Red Hat ログイン** に関連付けされているパスワード)

ネットワーク上でデータを送信する際に **SSL verify (SSL を照合する)** のオプションにチェックマークが付いている場合には、**SSL** プロトコルが使用されます。

MailX

MailX イベントの設定ウィンドウでは、以下のパラメーターを設定することができます:

- **Subject (題目)** — **Mailx** によって送信される問題レポート電子メールの **Subject** のフィールドに表示される文字列 (デフォルトでは、**"[abrt] detected a crash"** に設定されています)。
- **Sender (送信者)** — 問題レポート電子メールの **From** のフィールドに表示される文字列。
- **Recipient (受信者)** — 問題レポート電子メールの受信者の電子メールアドレス。

Send Binary Data (バイナリデータを送信する) オプションにチェックマークが付いている場合には、問題レポート電子メールには、その問題に関連した全バイナリファイルが添付されます。コアダンプファイルも、添付ファイルとして送信されます。

Kerneloops.org

Kerneloops.org イベント設定ウィンドウでは、以下のパラメーターを設定することができます:

- **Kerneloops URL** — カーネルの問題の報告先となる URL を指定します (デフォルトでは、http://submit.kerneloops.org/submitoops.php に設定されています)。

レポートアップローダー

Report Uploader (レポートアップローダー) のイベント設定ウィンドウでは、以下のパラメーターを設定することができます:

- **URL** — **FTP** または **SCP** のプロトコルを使用して、圧縮された問題データを格納する tarball をアップロードする場合のアップロード先 URL を指定します (デフォルトでは、**ftp://localhost:/tmp/upload** に設定されています)。

22.4.4. ABRT 固有の設定

ABRT の標準インストールでは、現在以下のような **ABRT** 固有の設定ファイルを提供しています:

- **/etc/abrt/abrt.conf** — **abrtd** サービスの動作を変更することができます。
- **/etc/abrt/abrt-action-save-package-data.conf** — **abrt-action-save-package-data** プログラムの動作を変更することができます。
- **/etc/abrt/plugins/CCpp.conf** — allows you to modify the behavior of **ABRT** のコアキャッチングフックの動作を変更することができます。

/etc/abrt/abrt.conf ファイルでサポートされている設定指示文は以下の通りです:

WatchCrashdumpArchiveDir = /var/spool/abrt-upload

このディレクトリーは、デフォルトではコメントアウトされています。**abrtd** が指定のディレクトリ内にあるクラッシュダンプの tarball アーカイブ (.tar.gz) を自動的にアンパックするようにしたい場合に、これを有効にします。上記の例では、**/var/spool/abrt-upload/** ディレクトリとなっています。このディレクティブでディレクトリを指定する場合は必ず、そのディレクトリが実際に存在し、**abrtd** による書き込みが可能であることを確認する必要があります。**ABRT** デーモンは、このディレクトリを自動的には作成しません。このオプションのデフォルト値を変更する場合は、**ABRT** の適正な機能を確保するために、このディレクトリが **DumpLocation** オプションで指定されたディレクトリと同一のもので **あってはならない** ことに注意してください。

警告

クラッシュダンプアーカイブの場所の変更を、先に SELinux ルールで反映しないで行うと、SELinux に拒否されてしまいます。SELinux における **ABRT** の実行に関する詳しい情報は、**abrt_selinux(8)** の man ページを参照して下さい。

SELinux の使用中にこのオプションを有効にする場合は、以下のコマンドを実行して、適切なブール値を設定し、**ABRT** が public_content_rw_t ドメインに書き込みできるようにする必要があることを念頭に置いて下さい。

```
setsebool -P abrt_anon_write 1
```

MaxCrashReportsSize = *<size_in_megabytes>*

このオプションは、**ABRT** が全ユーザーからの全問題情報を格納するのに使用するストレージの容量をメガバイト単位で設定します。デフォルトの設定値は **1000** MB です。ここで指定したクォータに達した後に **ABRT** が問題の取得を継続すると、新たなクラッシュダンプ用にスペースを空けるために、最も古く、最も大きなファイルが削除されます。

DumpLocation = /var/spool/abrt

このディレクティブは、デフォルトではコメントアウトされています。問題のコアダンプとその他すべての問題データを格納する、問題データディレクトリが作成される場所を指定します。デフォルトの場所は、**/var/spool/abrt** ディレクトリに指定されています。このディレクティブでディレクトリを指定する場合には必ず、そのディレクトリが実際に存在し、**abrtd** による書き込みが可能であることを確認する必要があります。このオプションのデフォルト値を変更する場合は、**ABRT** の適正な機能を確保するために、このディレクトリが **WatchCrashdumpArchiveDir** オプションで指定されたディレクトリと同一のもので **あってはならない** ことに注意してください。

警告

ダンプの場所の変更を、先に SELinux ルールで反映しないで行うと、SELinux に拒否されてしまいます。SELinux における **ABRT** の実行に関する詳しい情報は、**abrt_selinux(8)** の man ページを参照して下さい。

SELinux の使用中にこのオプションを有効にする場合は、以下のコマンドを実行して、適切なブール値を設定し、**ABRT** が public_content_rw_t ドメインに書き込みできるようにする必要があることを念頭に置いて下さい。

```
setsebool -P abrt_anon_write 1
```

以下の設定指示文は、**/etc/abrt/abrt-action-save-package-data.conf** ファイルでサポートされています:

OpenGPGCheck = *<yes/no>*

OpenGPGCheck 指示文を **yes** (デフォルト設定) に設定すると、**ABRT** は、**/etc/abrt/gpg_keys** ファイルに場所が記載されている GPG キーで署名されたパッケージで提供されているアプリケーションでのクラッシュの分析と処理***のみ***を実行するように指示されます。***OpenGPGCheck*** を **no** に設定すると、**ABRT** は、すべてのプログラム内のクラッシュをキャッチするように指示されます。

BlackList = nspluginwrapper, valgrind, strace, [*<MORE_PACKAGES>*]

BlackList 指示文の後に一覧表示してあるパッケージやバイナリ内のクラッシュは、**ABRT** では処理されません。**ABRT** がその他のパッケージやバイナリを無視するように設定したい場合は、ここにカンマで区切って列挙します。

ProcessUnpackaged = *<yes/no>*

この指示文は **ABRT** に対して、どのパッケージにも属しない実行ファイルにおけるクラッシュを処理するかどうかを指示します。デフォルトの設定値は **no** です。

BlackListedPaths = /usr/share/doc/*, */example*

これらのパスにある実行ファイルでのクラッシュは、**ABRT** には無視されます。

以下の設定指示文は、**/etc/abrt/plugins/CCpp.conf** ファイルでサポートされています:

MakeCompatCore = *<yes/no>*

この指示文は、**ABRT** がインストールされていない場合に実行されるように、**ABRT** のコアキャッチングフックがコアファイルを作成すべきかどうかを指定します。コアファイルは、通常クラッシュしたプログラムの現行ディレクトリ内に作成されますが、これは、**ulimit -c** の設定値で許可されている場合のみとなります。この指示文は、デフォルトでは **yes** に設定されています。

SaveBinaryImage = *<yes/no>*

この指示文は、**ABRT** のコアキャッチングフックがバイナリイメージをコアダンプに保存すべきかどうかを指定します。バイナリイメージは、削除されたバイナリ内で発生したクラッシュをデバッグする際に役立ちます。デフォルトの設定は **no** です。

22.4.5. カーネルパニックを検出する ABRT の設定

Red Hat Enterprise Linux 6.3 では、**ABRT** は abrt-addon-vmcore パッケージが提供する **abrt-vmcore** サービスを使ってカーネルパニックを検出できます。このサービスはシステム起動時に自動的にスタートし、**/var/crash/** ディレクトリのコアダンプファイルを検索します。コアダンプファイルが見つかると、**abrt-vmcore** が **/var/spool/abrt/** ディレクトリに **problem data directory** を作成し、コアダンプファイルをこの新規作成の problem data directory に移動します。**/var/crash/** ディレクトリの検索が終わると、サービスは次のシステム起動時まで停止します。

ABRT がカーネルパニックを検出するように設定するには、以下のステップを実行します。

1. **kdump** サービスがシステム上で有効になっていることを確認します。特に、kdump カーネル用に確保されたメモリー容量が正しく設定されている必要があります。この設定は、 **system-config-kdump** の使用または、**/etc/grub.conf** 設定ファイルにあるカーネルオプション一覧の **crashkernel** パラメーターを特定することで出来ます。**kdump** の有効化と設定の詳細については 26章*kdump クラッシュリカバリーサービス* を参照してください。

2. **Yum**パッケージインストーラーを使って abrt-addon-vmcore パッケージをインストーラーします。

   ```
   ~]# yum install abrt-addon-vmcore
   ```

   これで各サポートおよび設定ファイルとともに **abrt-vmcore** サービスがインストールされます。

3. 変更を有効にするためにシステムを再起動します。

ABRT が異なる設定になっている場合を除き、検出されたカーネルパニックの問題データは **/var/spool/abrt/** ディレクトリに保存され、検出された他のカーネル oops と同様に **ABRT** による処理が可能になります。

22.4.6. 自動ダウンロードおよび Debuginfo パッケージのインストール

ABRT は、特定の問題のデバッグログレベルに必要なパッケージを自動的にダウンロード、インストールするように設定できます。この機能は、自社環境内で問題をローカルでデバッグする際に有用です。debuginfo の自動ダウンロードおよびインストールを有効にするには、システムが以下の条件を満たしていることを確認します。

- **/etc/libreport/events.d/ccpp_event.conf** ファイルに、デフォルト設定でコメント解除されて存在する以下のアナライザーイベントが含まれている。

  ```
  EVENT=analyze_LocalGDB analyzer=CCpp
          abrt-action-analyze-core --core=coredump -o build_ids &&
          # In RHEL we don't want to install anything by default
          # and also this would fail, as the debuginfo repositories.
          # are not available without root password rhbz#759443
          # /usr/libexec/abrt-action-install-debuginfo-to-abrt-cache -
  -size_mb=4096 &&
          abrt-action-generate-backtrace &&
          abrt-action-analyze-backtrace
  ```

- **/etc/libreport/events.d/ccpp_event.conf** ファイルに、ABRT のバイナリー実行で分析対象の問題用の debuginfo パッケージがインストールされる以下の行が含まれている。不要なコンテンツのインストールを避けるために、この行はデフォルトでコメントアウトされているので、有効化するには先端の # 文字を削除する必要があります。

```
/usr/libexec/abrt-action-install-debuginfo-to-abrt-cache --
size_mb=4096 &&
```

- gdb パッケージは問題分析中のバックトレース生成を可能にするもので、システム上にインストールされます。**Yum** パッケージマネージャーでのパッケージのインストール方法については、「パッケージのインストール」 を参照してください。

重要

debuginfo パッケージは、root 権限を必要とする **rhnplugin** を使ってインストールすることに注意してください。このため、debuginfo パッケージをインストールするには、ABRT を **root** として実行する必要があります。

22.4.7. 自動報告機能の設定

ABRT は、検出された問題またはクラッシュをユーザーの介入なしで自動的に報告するように設定することが可能です。これは、analyze-and-report ルールを **post-create** ルールとして指定することで設定できます。例えば、ABRT が Python クラッシュをユーザーの介入なしで即時に Bugzilla に報告するようにするには、**/etc/libreport/events.d/python_event.conf** ファイルで、このルールを有効にして、**EVENT=report_Bugzilla** 条件を **EVENT=port-create** 条件に置き換えます。新たなルールは以下のようになります。

```
EVENT=post-create analyzer=Python
        test -f component || abrt-action-save-package-data
        reporter-bugzilla -c /etc/abrt/plugins/bugzilla.conf
```

警告

post-create イベントは、通常 root 権限で実行する **abrtd** によって実行される点に注意して下さい。

22.4.8. プロキシサーバーを使用したアップロードと報告

reporter-bugzilla 及び **reporter-upload** のツールは、 **http_proxy** 及び **ftp_proxy** の環境変数を優先します。報告イベントの一部として環境変数を使用する場合には、報告を実行するプロセス (通常は **abrt-gui** または **abrt-cli**) からそれらの値を継承します。従って、作業環境でこれらの変数を使用すると、**HTTP** または **FTP** のプロキシサーバーを指定することができます。

これらのツールが **post-create** イベントの一部になるようにすると、**abrtd** プロセスの子プロセスとして実行されます。これらの変数を設定するには、abrtd の環境の調整またはルールの変更を行う必要があります。以下が例となります:

```
EVENT=post-create analyzer=Python
        test -f component || abrt-action-save-package-data
        export http_proxy=http://proxy.server:8888/
        reporter-bugzilla -c /etc/abrt/plugins/bugzilla.conf
```

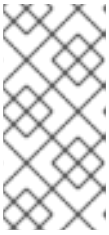
22.5. 集中型のクラッシュ収集の設定

ABRT では、クラッシュレポートを複数のシステムから収集して専用システムに送信し、更に処理するようにを設定することができます。これは、管理者が数百に及ぶシステムにログインして**ABRT** によって検出されたクラッシュを手動でチェックする手間を省きたい場合に役立ちます。この方法を使用するには、**libreport-plugin-reportuploader** プラグインをインストールする必要があります (**yum install libreport-plugin-reportuploader**)。以下のセクションでは、ABRT の集中型クラッシュ収集を使用するためのシステムの設定方法について説明します。

22.5.1. 専用システム上で必要な設定手順

専用の (サーバー) システム上で、以下の手順を実行して下さい:

1. クラッシュレポートのアップロード先となるディレクトリを作成します。通常は、**/var/spool/abrt-upload/** を使用します (本ガイドでは、これ以降、このディレクトリを使用することを前提としています)。abrt ユーザーが、このディレクトリに書き込み可能であることを確認して下さい。

   注記

   abrt-desktop パッケージがインストールされると、新たなシステムユーザーとグループが作成され、いずれも **abrt** と命名されます。このユーザーは、**abrtd** デーモンによって使用されます (例: **/var/spool/abrt/*** ディレクトリの オーナー:グループ)。

2. **/etc/abrt/abrt.conf** 設定ファイルで、**WatchCrashdumpArchiveDir** 指示文を以下のように設定します:

   ```
   WatchCrashdumpArchiveDir = /var/spool/abrt-upload/
   ```

3. 任意のアップロードメカニズム (例: **FTP**、または、**SCP**) を選択します。**FTP** の設定方法についての詳細は、「FTP」 を参照して下さい。**SCP** の設定方法の詳細は、「**scp** ユーティリティの使用」 を参照して下さい。

   アップロードメソッドが機能するかどうかを確認しておくことを推奨します。例えば、**FTP** を使用する場合、インタラクティブな **FTP** クライアントを使用して、ファイルをアップロードします:

   ```
   ~]$ ftp
   ftp> open <servername>
   Name: <username>
   Password: <password>
   ftp> cd /var/spool/abrt-upload
   250 Operation successful
   ftp> put <testfile>
   ftp> quit
   ```

   <testfile> が、サーバーシステム上の正しいディレクトリに表示されるかどうかを確認します。

4. クラッシュデータのサイズがデフォルトの **1000** MB を超えると予想される場合は、**MaxCrashReportsSize** 指示文 (**/etc/abrt/abrt.conf** 設定ファイル内) をより大きな値に設定する必要があります。

5. C/C++ クラッシュのバックトレースを生成するかどうか検討します。

   バックトレースを全く生成したくない場合や、問題が発生したマシン上にローカルでバックトレースを作成するようにしたい場合には、サーバー上のバックトレース生成を無効にすることができます。ABRT の標準インストールでは、C/C++ クラッシュのバックトレースは、**/etc/libreport/events.d/ccpp_events.conf** 設定ファイル内の以下のルールで生成されます:

   ```
   EVENT=analyze_LocalGDB analyzer=CCpp
           abrt-action-analyze-core.py --core=coredump -o build_ids &&
           abrt-action-install-debuginfo-to-abrt-cache --size_mb=4096
   &&
           abrt-action-generate-backtrace &&
           abrt-action-analyze-backtrace
   ```

   アップロードされるデータにこのルールが適用されないようにするには、ルールに **remote!=1** の条件を追加します。

6. 問題データ内のパッケージ情報 (**package** 及び **component** のエレメント) を収集するかどうかを決定します。集中型クラッシュ収集設定におけるパッケージ情報収集の必要性及びその適切な設定方法については、「パッケージ情報の保存」 を参照して下さい。

22.5.2. クライアントシステム上で必要な設定ステップ

集中管理メソッドを使用する全クライアントシステム上で、以下の手順を実行します:

1. バックトレースを生成したくない場合、或いはサーバーシステム上で生成したい場合には、**/etc/libreport/events.d/ccpp_events.conf** ファイル内の該当するルールを削除またはコメントアウトする必要があります。設定例は、「専用システム上で必要な設定手順」を参照して下さい。

2. クライアントマシン上のパッケージ情報を収集しない場合には、**/etc/libreport/events.d/abrt_event.conf** ファイル内で、abrt-action-save-package-data を実行するルールを削除、コメントアウト、或いは変更します。集中型クラッシュ収集の設定におけるパッケージ情報収集の必要性及びその適切な設定方法については、「パッケージ情報の保存」 を参照して下さい。

3. 該当する設定ファイルに、サーバーシステムへの問題レポートアップロードのルールを追加します。例えば、すべての問題を検出され次第自動的にアップロードする場合には、**/etc/libreport/events.d/abrt_event.conf** 設定ファイルで以下のルールを使用します:

   ```
   EVENT=post-create
           reporter-upload -u scp://user:password@server.name/directory
   ```

   問題データをローカルで保管した後に、ABRT GUI/CLI を使用してアップロードする場合には、reporter-upload プログラムを **report_*SFX*** イベントとして実行する同様のルールを使用することもできます。このイベントの例は以下のようになります:

   ```
   EVENT=report_UploadToMyServer
           reporter-upload -u scp://user:password@server.name/directory
   ```

22.5.3. パッケージ情報の保存

単一マシンの **ABRT** インストールでは、問題は通常 RHTSupport や Bugzilla などの外部のバグデータベースに報告されます。これらのバグデータベースに報告するには、通常、問題が発生したコンポーネントとパッケージについての情報が必要となります。**post-create** イベントは、(数ある手順の中で) **abrt-action-save-package-data** ツールを実行し、**ABRT** の標準インストール内にある、この情報を提供します。

集中型クラッシュ収集システムを設定する場合には、要件が大幅に異なる場合があります。必要に応じて、以下のような 2 つのオプションがあります:

問題の社内分析

外部のバグデータベースに報告せずに、社内で問題を分析する場合には、問題データを収集した後にパッケージ情報を収集する必要はありません。自社開発のプログラムまたはシステム上にインストールされたサードパーティのアプリケーションで発生したクラッシュの収集が有益な場合もあります。もしプログラムが RPM パッケージの一部ならば、可能なのは **クライアントシステム** 上と **専用のクラッシュ収集システム** 上で **/etc/abrt/gpg_keys** ファイルにそれぞれの GPG ファイルを追加するか、**/etc/abrt/abrt-action-save-package-data.conf** ファイルに以下の行を設定することのみです。

```
OpenGPGCheck = no
```

プログラムがいずれの RPM パッケージに属していない場合は、**クライアントシステム** 上と **専用のクラッシュ収集システム** 上の両方で以下のステップを実行します。

- **/etc/libreport/events.d/abrt_event.conf** ファイルから以下のルールを削除します:

```
EVENT=post-create component=
        abrt-action-save-package-data
```

- いずれのインストール済みパッケージにも対応しない問題データディレクトリの削除を防止するには、**/etc/abrt/abrt-action-save-package-data.conf** ファイルで以下の指示文を設定します:

```
ProcessUnpackaged = yes
```

外部バグデータベースへの報告

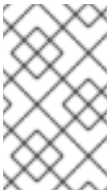
RHTSupport または Bugzilla にクラッシュを報告する場合には、パッケージ情報を収集する必要があります。通常、クライアントマシンとクラッシュ収集専用システムにインストールされているパッケージセットは同じではありません。このため、クライアントからアップロードされた問題データが、クラッシュ収集専用システムにインストールされているパッケージと一致しないことがあります。このような場合、**ABRT** の標準設定では、問題データが削除されてしまいます (ABRT が、その問題をアンパッケージされていない実行可能ファイル内のクラッシュと判断するため)。こういった事態を防ぐには、**専用システム** で **ABRT** の設定ファイルを以下のように変更しておく必要があります:

- クライアントマシンからアップロードされた問題データのパッケージ情報を間違えて収集しないようにするには、**/etc/libreport/events.d/abrt_event.conf** ファイルに **remote!=1** 条件を追加します:

```
EVENT=post-create remote!=1 component=
        abrt-action-save-package-data
```

- いずれのインストール済みパッケージにも該当しない問題データディレクトリが削除されないようにするには、**/etc/abrt/abrt-action-save-package-data.conf** で以下の指示文を設定します:

```
ProcessUnpackaged = yes
```



注記

この場合、クライアントシステムでは設定を変更する必要がない点に注意して下さい。クライアントシステムは、パッケージ情報の収集を継続し、アンパッケージされた実行可能ファイル内のクラッシュは引き続き無視します。

22.5.4. ABRT のクラッシュ検出テスト

設定プロセスの手順がすべて完了したら、基本セットアップは終了です。このセットアップが適切に機能するかどうかをテストするには、**kill -s SEGV *<PID>*** のコマンドを使ってクライアントシステム上のプロセスを終了します。例えば、**sleep** プロセスを開始し、以下のように **kill** コマンドを使用して終了します。

```
~]$ sleep 100 &
[1] 2823
~]$ kill -s SEGV 2823
```

ABRT は、**kill** コマンドを実行した直後にクラッシュを検出するはずです。クライアントシステム上で **ABRT** によってクラッシュが検出されているかどうかをチェックし (これは、**abrt-cli list --full** コマンドを実行して、該当する syslog ファイルを調べるか、**/var/spool/abrt** ディレクトリ内に作成されたクラッシュダンプを調べることによって、確認できます)、それがサーバーシステムにコピーされて、サーバーシステム上でアンパックされ、サーバーシステム上で **abrt-cli** または **abrt-gui** を使用して表示と処理が可能であることを確認して下さい。

第23章 OPROFILE

OProfile は維持負担の少ないシステム全体のパフォーマンス監視ツールです。プロセッサ上にあるパフォーマンス監視ハードウェアを使用して、メモリーの参照時期、L2 キャッシュ要求の回数、及び受け取ったハードウェア割り込みの回数など、システム上のカーネルと実行可能ファイルに関する情報を取り込みます。Red Hat Enterprise Linux システムでは、このツールを使用するために **oprofile** パッケージをインストールする必要があります。

多くのプロセッサには、専用のパフォーマンス監視ハードウェアが含まれています。このハードウェアにより特定のイベント (要求されたデータがキャッシュ内に存在しないなど) の発生を検出することができるようになります。このハードウェアは通常、単数または複数の *カウンター* の形式を取り、イベントの発生の度にカウントが増加します。カウンターの値が実際に繰り上げになると、割り込みが生成されます。これによりパフォーマンス監視で生成された詳細 (及び、維持負担) の量の制御が可能になります。

OProfile は、カウンターが割り込みを生成する度にこのハードウェア (または、パフォーマンス監視ハードウェア不在の場合には、タイマーベースの代用品) を使用してパフォーマンス関連のデータの *サンプル* を収集します。これらのサンプルは定期的にディスクに書き込まれて、その後、これらのサンプル内に含まれるデータはシステムレベル、及びアプリケーションレベルのパフォーマンスに関する情報の生成に利用されます。

OProfile は役に立つツールですが、その使用の際にはいくつかの制約があります:

- **共有ライブラリの使用** — 共有ライブラリ内のコードサンプルは、**--separate=library** オプションを使用しない限りは特定のアプリケーションへの属性を持ちません。
- **パフォーマンス監視サンプルは不正確** — パフォーマンス監視登録がサンプルを立ち上げる際に、割り込みの処理はゼロでの割り算例外エラーのように正確ではありません。プロセッサによる命令の無秩序な実行により、サンプルは近隣の命令に記録される可能性があります。
- **opreport はサンプルをインライン関数用に不的確に関連付け** — **opreport** は、単純なアドレス範囲のメカニズムを使用してどの関数にアドレスがあるかを判定します。インライン関数のサンプルは、インライン関数に帰属するのではなく、むしろインライン関数が挿入されている関数に帰属します。
- **OProfile 複数の実行によるデータを集積** — OProfile は、システム全体のプロファイラーであり複数のスタートアップとシャットダウンのプロセスを想定します。そのため、複数実行からのサンプルが集積します。コマンド **opcontrol --reset** を使用すると以前の実行からのサンプルをクリアすることができます。
- **ハードウェアパフォーマンスカウンターはゲスト仮想マシン上では無能** — ハードウェアパフォーマンスカウンターは仮想システム上では利用できないため、**timer** モードを使用する必要があります。コマンド **opcontrol --deinit** を実行して、それから **modprobe oprofile timer=1** の実行により **timer** モードを有効にします。
- **CPU-限定パフォーマンス以外の問題** — OProfile はCPU 限定のプロセスで問題を見つけるように設定されています。ロック状態か、あるいは他のイベントの発生を待つために (例えば、 I/O が動作を終了するまで) 待機して眠っているプロセスを OProfile は識別しません。

23.1. ツールの概要

表23.1「OProfile コマンド」 は、**oprofile** パッケージに装備されているツールの概要を提供します。

表23.1 OProfile コマンド

コマンド	詳細
ophelp	システムプロセッサで使用可能なイベントとその簡単な説明を表示します。
opimport	サンプルデータベースファイルをシステム用に外部のバイナリ形式からネイティブの形式に変換します。異なるアーキテクチャからのサンプルデータベースを解析する時にのみこのオプションを使用して下さい。
opannotate	アプリケーションがデバッグシンボルでコンパイルされている場合は、実行可能ファイル用の注釈付きのソースを作成します。詳細については、「**opannotate** の使用」 を参照して下さい。
opcontrol	どのデータが収集されるかを設定します。詳細については、「OProfile の設定」 を参照して下さい。
opreport	プロファイルデータを取り込みます。詳細については、「**opreport** の使用」 を参照して下さい。
oprofiled	デーモンとして実行して定期的にサンプルデータをディスクに書き込みます。

23.2. OPROFILE の設定

OProfile を実行できるようにするには設定が必要です。最低でも、カーネルを監視する選択 (またはカーネルを監視しない選択) が必要です。以下のセクションでは、OProfile を設定するための **opcontrol** ユーティリティの使用法を説明しています。**opcontrol** コマンドが実行されると、そのセットアップオプションは **/root/.oprofile/daemonrc** ファイル内に保存されます。

23.2.1. カーネルの指定

先ず最初に OProfile がカーネルを監視すべきかどうかを設定します。これが OProfile を開始する前に要求される唯一の設定オプションです。他のすべてはオプションです。

カーネルを監視するには、root として以下のコマンドを実行します:

```
~]# opcontrol --setup --vmlinux=/usr/lib/debug/lib/modules/`uname -
r`/vmlinux
```

重要

カーネルを監視するためには、カーネル用の debuginfo パッケージ (展開したカーネルを含む) をインストールする必要があります。

カーネルを監視しないように OProfile を設定するには、root として以下のコマンドを実行します:

```
~]# opcontrol --setup --no-vmlinux
```

このコマンドはまた、**oprofile** カーネルモジュールがまだロードされていない場合にそれをロードします。そして**/dev/oprofile/** ディレクトリがまだ存在しない場合には、そのディレクトリを作成します。このディレクトリに関する詳細には 「**/dev/oprofile/** を理解する」 を参照して下さい。

カーネル内でサンプルが収集されるべきかどうかのセッティングはどのデータが収集されるかのみを変更します。収集されたデータの保存方法や保存場所は変更しません。カーネル及びアプリケーションライブラリ用に異なるサンプルファイルを生成する方法については 「カーネルとユーザースペースのプロファイルの分離」 を参照して下さい。

23.2.2. 監視するイベントのセッティング

ほとんどのプロセッサには *カウンター* が装備されており、OProfile はこれを使用して特定のイベントを監視します。表23.2「OProfile プロセッサ及びカウンター」 で示してあるように、利用可能なカウンターの数はプロセッサにより異なります。

表23.2 OProfile プロセッサ及びカウンター

プロセッサ	cpu_type	カウンターの数
AMD64	x86-64/hammer	4
AMD Athlon	i386/athlon	4
AMD Family 10h	x86-64/family10	4
AMD Family 11h	x86-64/family11	4
AMD Family 12h	x86-64/family12	4
AMD Family 14h	x86-64/family14	4
AMD Family 15h	x86-64/family15	6
IBM eServer System i 及び IBM eServer System p	タイマー	1
IBM POWER4	ppc64/power4	8
IBM POWER5	ppc64/power5	6
IBM PowerPC 970	ppc64/970	8
IBM S/390 及び IBM System z	タイマー	1
Intel Core i7	i386/core_i7	4
Intel Nehalem マイクロアーキテクチャ	i386/nehalem	4
Intel Pentium 4 (ハイパースレッドなし)	i386/p4	8
Intel Pentium 4 (ハイパースレッドあり)	i386/p4-ht	4
Intel Westmere マイクロアーキテクチャ	i386/westmere	4

プロセッサ	cpu_type	カウンターの数
TIMER_INT	タイマー	1

表23.2「OProfile プロセッサ及びカウンター」 を使用すると、正しいプロセッサタイプが検出されたことを立証し、更に同時に監視できるイベントの数を決定できます。サポートされているパフォーマンス監視ハードウェアをプロセッサが装備していない場合、**timer** が、プロセッサタイプとして使用されます。

timer が使用される場合、ハードウェアはハードウェアパフォーマンスカウンター用のサポートを持たないため、イベントをプロセッサ用に設定できません。その代わりに、タイマー割り込み (timer interrupt) がプロファイル用に使用されます。

timer がプロセッサタイプとして使用されない場合は、監視されるイベントは変更できません。そしてプロセッサ用にカウンター 0 がデフォルトでタイムベースのイベントにセットされます。プロセッサ上に複数のカウンターが存在する場合、カウンター 0 以外のカウンターはデフォルトではイベントにセットされません。監視されるデフォルトのイベントは 表23.3「デフォルトのイベント」 に示してあります。

表23.3 デフォルトのイベント

プロセッサ	カウンター用のデフォルトイベント	詳細
AMD Athlon 及び AMD64	CPU_CLK_UNHALTED	プロセッサのクロックは停止していません
AMD Family 10h, AMD Family 11h, AMD Family 12h	CPU_CLK_UNHALTED	プロセッサのクロックは停止していません
AMD Family 14h, AMD Family 15h	CPU_CLK_UNHALTED	プロセッサのクロックは停止していません
IBM POWER4	サイクル	プロセッサのサイクル
IBM POWER5	サイクル	プロセッサのサイクル
IBM PowerPC 970	サイクル	プロセッサのサイクル
Intel Core i7	CPU_CLK_UNHALTED	プロセッサのクロックは停止していません
Intel Nehalem マイクロアーキテクチャ	CPU_CLK_UNHALTED	プロセッサのクロックは停止していません
Intel Pentium 4 (ハイパースレッド有りと無し)	GLOBAL_POWER_EVENTS	プロセッサが停止していない期間の長さ

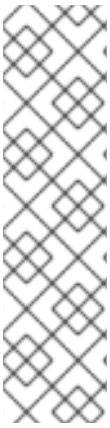
プロセッサ	カウンター用のデフォルトイベント	詳細
Intel Westmere マイクロアーキテクチャ	CPU_CLK_UNHALTED	プロセッサのクロックは停止していません
TIMER_INT	(なし)	各タイマー割り込みのサンプル

一度に監視できるイベントの数はプロセッサのカウンターの数で決定されます。しかし、それは 1 対 1 の関係ではなく、一部のプロセッサでは一定のイベントは特定のカウンターにマップしなければなりません。利用可能なカウンターを判定するには、以下のコマンドを実行します:

```
~]# ls -d /dev/oprofile/[0-9]*
```

利用可能なイベントはプロセッサタイプによって変わります。プロファイリング用に利用可能なイベントを判定するには、root として以下のコマンドを実行します。(一覧はシステムのプロセッサタイプに特有のものです):

```
~]# ophelp
```



注記

OProfile 適正に設定されていないと、**ophelp** は以下のエラーメッセージが出て失敗します。

```
Unable to open cpu_type file for reading
Make sure you have done opcontrol --init
cpu_type 'unset' is not valid
you should upgrade oprofile or force the use of timer mode
```

OProfile の設定については、「OProfile の設定」 に記載の指示にしたがってください。

各カウンター用のイベントはコマンドライン、またはグラフィカルインターフェイスを介して設定できます。グラフィカルインターフェイスに関する情報については 「グラフィカルインターフェース」 を参照して下さい。カウンターが特定のイベントにセットできない場合は、エラーメッセージが表示されます。

コマンドラインを介して各設定可能なカウンター用にイベントをセットするには、**opcontrol** を使用します:

```
~]# opcontrol --event=event-name:sample-rate
```

event-name を **ophelp** からのイベントの実際の名前で入れ替え、*sample-rate* をサンプル間のイベントの数で入れ替えます。

23.2.2.1. サンプルのレート

デフォルトでは、タイムベースのイベントセットが選択されます。これはプロセッサ別に100,000 クロックサイクル毎にサンプルを 1 つ作成します。タイマー割り込みが使用される場合は、タイマーは jiffy レートにセットされて、ユーザー設定ではありません。**cpu_type** が **timer** でない場合、各イベ

ントは *サンプルのレート* をそれ自身にセットすることができます。サンプルレートとは、各サンプルのスナップショットの間隔内に於けるイベントの数です。

カウンター用にイベントをセットする際には、サンプルレートも指定できます:

```
~]# opcontrol --event=event-name:sample-rate
```

sample-rate を再度サンプルを取るまでのイベントの数で入れ替えます。カウントが少ないとサンプル取りの回数が増えます。頻繁に発生しないイベントには、イベントのインスタンスをキャプチャするために低いカウントが必要になるでしょう。

警告

サンプルレートをセットする際には細心の注意が必要です。サンプルの頻度が高すぎると、システム負荷が大きくなり過ぎてシステムがフリーズしたかのように見えるか、または実際にフリーズする原因になります。

23.2.2.2. ユニットマスク

一部のユーザーパフォーマンス監視イベントでは、イベントを更に定義するためにユニットマスクも必要かも知れません。

各イベントのユニットマスクは **ophelp** コマンドで一覧表示されます。各ユニットマスクの値は 16進法形式で一覧になっています。複数のユニットマスクを指定するには、16進法の値をビットサイズの *or* オペレーションの使用と組み合わせる必要があります。

```
~]# opcontrol --event=event-name:sample-rate:unit-mask
```

23.2.3. カーネルとユーザースペースのプロファイルの分離

デフォルトでは、カーネルモードとユーザーモードの情報は各イベントで収集されます。特定のカウンターの為にカーネルモードのイベントを無視するように OProfile を設定するには、以下のコマンドを実行します:

```
~]# opcontrol --event=event-name:sample-rate:unit-mask:0
```

そのカウンターの為のカーネルモードのプロファイリングを再度スタートするには、以下のコマンドを実行します:

```
~]# opcontrol --event=event-name:sample-rate:unit-mask:1
```

特定のカウンターのためにユーザーモードのイベントを無視するように OProfile を設定するには、以下のコマンドを実行します:

```
~]# opcontrol --event=event-name:sample-rate:unit-mask:kernel:0
```

そのカウンターのためのユーザーモードのプロファイリングを再度スタートするには、以下のコマンドを実行します:

```
~]# opcontrol --event=event-name:sample-rate:unit-mask:kernel:1
```

OProfile デーモンがサンプルファイルに対してプロファイルデータを書き込む際には、カーネルとライブラリプロファイルデータを別々のサンプルファイルに分離することができます。デーモンがサンプルファイルに書き込む方法を設定するには、root として以下のコマンドを実行します:

```
~]# opcontrol --separate=choice
```

choice は以下のいずれかです:

- **none** — プロファイルを分離しません (デフォルト)
- **library** — ライブラリ用にアプリケーション単位のプロファイルを生成します
- **kernel** — カーネルとカーネルモジュール用にアプリケーション単位のプロファイルを生成します
- **all** — ライブラリ用にアプリケーション単位のプロファイルを、カーネルとカーネルモジュール用にアプリケーション単位のプロファイルを生成します

--separate=library が使用される場合は、サンプルファイルの名前は実行可能ファイルの名前とライブラリの名前を含みます。

注記

これらの設定の変更は OProfile プロファイラーが再起動した時点に反映されます。

23.3. OPROFILE の開始と停止

OProfile でシステム監視を開始するには、root として以下のコマンドを実行します:

```
~]# opcontrol --start
```

出力は以下の表示のようになります:

```
Using log file /var/lib/oprofile/oprofiled.log Daemon started. Profiler
running.
```

/root/.oprofile/daemonrc のセッティングが使用されます。

OProfile のデーモン、**oprofiled** が開始されます。これは定期的にサンプルデータを **/var/lib/oprofile/samples/** ディレクトリに書き込みます。このデーモンのログファイルは **/var/lib/oprofile/oprofiled.log** にあります。

重要

Red Hat Enterprise Linux 6 システムでは、**nmi_watchdog** は **perf** サブシステムで登録をします。このため、**perf** サブシステムはブート時にパフォーマンスカウンター登録の制御を獲得し、OProfile の動作を阻止してしまいます。

これを解決するためには、**nmi_watchdog=0** のカーネルパラメータセットでブートするか、あるいは、ランタイム時に **nmi_watchdog** を無効にするために以下のコマンドを実行します:

```
~]# echo 0 > /proc/sys/kernel/nmi_watchdog
```

nmi_watchdog を再度有効にするには、以下のコマンドを実行します:

```
~]# echo 1 > /proc/sys/kernel/nmi_watchdog
```

プロファイラーを停止するには、root として以下のコマンドを実行します:

```
~]# opcontrol --shutdown
```

23.4. データの保存

時には、特定の時期にサンプルを保存することが役に立ちます。例えば、実行可能ファイルをプロファイルする際に異なる入力データセットを基にした異なるサンプルを収集することが役に立ちます。監視するイベントの数がプロセッサで利用できるカウンターの数を超過する場合は、OProfile の複数回実行によりデータを収集することで、その度にサンプルデータを異なるファイルに保存することができます。

サンプルファイルの現在のセットを保存するには、以下のコマンドを実行します。*name* は現在のセッションに特有な記述的名前で入れ替えます。

```
~]# opcontrol --save=name
```

ディレクトリ **/var/lib/oprofile/samples/*name*/** が作成されて、現在のサンプルファイルがそこにコピーされます。

23.5. データの分析

OProfile デーモンである **oprofiled** は定期的にサンプルを収集して、それらを**/var/lib/oprofile/samples/** ディレクトリに書き込みます。そのデータを読み込む前に、root として以下のコマンドを実行することでこのディレクトリにすべてのデータが書き込まれていることを確認します:

```
~]# opcontrol --dump
```

各サンプル名は実行可能ファイルの名前を基にしてあります。例えば、Pentium III プロセッサ上の **/bin/bash** 用のデフォルトイベントのサンプルは以下のようになります:

```
\{root\}/bin/bash/\{dep\}/\{root\}/bin/bash/CPU_CLK_UNHALTED.100000
```

収集された後にそのサンプルデータのプロファイルに使用できるツールは以下のようになります:

- **opreport**
- **opannotate**

プロファイルされたバイナリとの併用でこれらのツールを使用すると更なる分析が可能になるレポートを生成できます。

警告

データを分析するためには、プロファイルされる実行可能ファイルがこれらのツールで使用される必要があります。データの収集後に実行可能ファイルの変更が必要な場合はサンプルの作成に使用された実行可能ファイル、及びサンプルファイルをバックアップしておきます。サンプルファイルとバイナリは一致しなければならないことに注意して下さい。これらが一致しないとバックアップの作成はうまくできません。この問題の対処には **oparchive** を使用することができます。

各実行可能ファイルのサンプルは単独のサンプルファイルに書き込まれます。動的にリンクした各ライブラリからのサンプルも単独のサンプルファイルに書き込まれます。監視されている実行可能ファイルが OProfile の実行中に変化して、実行可能ファイルのサンプルファイルが存在する場合は、存在するサンプルファイルは自動的に削除されます。そのため、存在中のサンプルファイルが必要な場合はバックアップが必要です。これは実行可能ファイルを新バージョンで入れ替える前にサンプルの作成に使用した実行可能ファイルと共にバックアップする必要があります。OProfile の分析ツールはサンプルの作成に使った実行可能ファイルを分析で使用します。実行可能ファイルが変化すると分析ツールは関連したサンプルを分析できなくなります。サンプルファイルのバックアップ方法については 「データの保存」を参照して下さい。

23.5.1. opreport の使用

opreport ツールはプロファイルされるすべての実行可能ファイルについての概要を提供します。

サンプル出力の一部を以下に示します:

```
Profiling through timer interrupt
TIMER:0|
samples|      %|
------------------
25926 97.5212 no-vmlinux
359  1.3504 pi
65  0.2445 Xorg
62  0.2332 libvte.so.4.4.0
56  0.2106 libc-2.3.4.so
34  0.1279 libglib-2.0.so.0.400.7
19  0.0715 libXft.so.2.1.2
17  0.0639 bash
8  0.0301 ld-2.3.4.so
8  0.0301 libgdk-x11-2.0.so.0.400.13
6  0.0226 libgobject-2.0.so.0.400.7
5  0.0188 oprofiled
4  0.0150 libpthread-2.3.4.so
4  0.0150 libgtk-x11-2.0.so.0.400.13
```

```
3  0.0113 libXrender.so.1.2.2
3  0.0113 du
1  0.0038 libcrypto.so.0.9.7a
1  0.0038 libpam.so.0.77
1  0.0038 libtermcap.so.2.0.8
1  0.0038 libX11.so.6.2
1  0.0038 libgthread-2.0.so.0.400.7
1  0.0038 libwnck-1.so.4.9.0
```

各実行可能ファイルは独立した行で一覧表示されています。最初のカラムは実行可能ファイル用に記録されたサンプルの数です。２つめのカラムはサンプル合計に対するサンプルの相対的パーセンテージです。３つめのカラムは実行可能ファイルの名前を示します。

利用可能なコマンドラインオプションの一覧については **opreport** man ページを参照して下さい。サンプルの最小数からサンプルの最大数までの実行可能ファイルの出力を配列するために使用される **-r** オプションなどがあります。

23.5.2. 単独実行可能ファイルで opreport の使用

特定の実行可能ファイルに関する詳細にプロファイルされた情報を取り込むには、**opreport** を使用します:

```
~]# opreport mode executable
```

executable は、分析される実行可能ファイルへの完全パスでなければなりません。*mode* は以下のいずれかです:

-l

シンボル別にサンプルデータを一覧表示します。例えば、以下の例はコマンド **opreport -l /lib/tls/libc-*version*.so** の実行による出力の一部を示しています:

```
samples % symbol name
12 21.4286 __gconv_transform_utf8_internal
5 8.9286 _int_malloc 4 7.1429 malloc
3 5.3571 __i686.get_pc_thunk.bx
3 5.3571 _dl_mcount_wrapper_check
3 5.3571 mbrtowc
3 5.3571 memcpy
2 3.5714 _int_realloc
2 3.5714 _nl_intern_locale_data
2 3.5714 free
2 3.5714 strcmp
1 1.7857 __ctype_get_mb_cur_max
1 1.7857 __unregister_atfork
1 1.7857 __write_nocancel
1 1.7857 _dl_addr
1 1.7857 _int_free
1 1.7857 _itoa_word
1 1.7857 calc_eclosure_iter
1 1.7857 fopen@@GLIBC_2.1
1 1.7857 getpid
1 1.7857 memmove
1 1.7857 msort_with_tmp
1 1.7857 strcpy
1 1.7857 strlen
```

```
1 1.7857 vfprintf
1 1.7857 write
```

最初のカラムはそのシンボルのサンプル数で、２つめのカラムは実行可能ファイル用の全体のサンプルに対して相対的なこのシンボルのサンプルのパーセンテージで、そして３つめのカラムはシンボルの名前です。

サンプルの最大数から最小数 (逆順) に出力を配列するには、**-l** オプションを**-r** オプションと併用して使用します。

-i *symbol-name*

シンボル名に特有のサンプルデータを一覧表示します。例えば、以下の例はコマンド **opreport -l -i __gconv_transform_utf8_internal /lib/tls/libc-*version*.so** の実行による出力を示しています:

```
samples % symbol name
12 100.000 __gconv_transform_utf8_internal
```

最初の行はシンボルと実行可能ファイルの合同要約です。

最初のカラムはメモリーシンボルのサンプルの数です。２つめのカラムはシンボルのサンプル合計数に対して相対的なメモリーアドレスのサンプルのパーセンテージです。３つめのカラムはシンボル名です。

-d

シンボル別のサンプルデータを **-l** よりももっと詳細に一覧表示します。例えば、以下の例はコマンド **opreport -l -d __gconv_transform_utf8_internal /lib/tls/libc-*version*.so** の実行による出力を示しています:

```
vma samples % symbol name
00a98640 12 100.000 __gconv_transform_utf8_internal
00a98640 1 8.3333
00a9868c 2 16.6667
00a9869a 1 8.3333
00a986c1 1 8.3333
00a98720 1 8.3333
00a98749 1 8.3333
00a98753 1 8.3333
00a98789 1 8.3333
00a98864 1 8.3333
00a98869 1 8.3333
00a98b08 1 8.3333
```

データは **-l** オプションと同じですが、それ以外に各シンボル毎に使用される各仮想メモリーアドレスが示してあります。各仮想メモリーアドレスについては、そのサンプルの数とシンボル用のサンプル合計数に対するメモリーアドレスサンプルのパーセンテージが表示されています。

-x *symbol-name*

出力からコンマで隔離されたシンボルの一覧を除外します。

session:*name*

/var/lib/oprofile/samples/ ディレクトリに対して相対的となるセッション、またはディレクトリへの完全パスを指定します。

23.5.3. モジュールについてより詳細な出力を取得

OProfile はシステム全体に於いてカーネルとマシン上で実行中のユーザースペースコードのデータを収集します。しかし、モジュールがカーネルにロードされるとカーネルモジュールの発生源に関する情報は消失します。モジュールはブート時に **initrd** ファイルから来たか、各種カーネルモジュールのあるディレクトリから来たか、またはローカルで作成されたカーネルモジュールから来たのかも知れません。その結果、OProfile が 1 つのモジュールのサンプルを記録する際に実行可能ファイルのモジュール群サンプルを root ディレクトリ内に一覧表示しますが、この場所はモジュールの実際のコードがある場所だとは断定できません。分析ツールが実行可能ファイルを確実に取得するようにいくつかの手順を実行する必要があります。

モジュールのアクションについてより明確な眺望を得るには、モジュールを "unstripped" 状態にする (即ち、カスタムビルドからインストールした状態にする) か、またはカーネル用に debuginfo パッケージをインストールする必要があります。

実行中のカーネルを判定するには、**uname -a** コマンドを使用します。そして適切な debuginfo パッケージを入手してそれをマシンにインストールします。

それから以前の実行によるサンプルを排除するために、以下のコマンドを入力します:

```
~]# opcontrol --reset
```

例えば Westmere プロセッサを持つマシンで監視プロセスを開始するには、以下のコマンドを実行します:

```
~]# opcontrol --setup --vmlinux=/usr/lib/debug/lib/modules/`uname -
r`/vmlinux --event=CPU_CLK_UNHALTED:500000
```

その後、例えば ext4 モジュール用の詳細情報は以下のようにして取得できます:

```
~]# opreport /ext4 -l --image-path /lib/modules/`uname -r`/kernel
CPU: Intel Westmere microarchitecture, speed 2.667e+06 MHz (estimated)
Counted CPU_CLK_UNHALTED events (Clock cycles when not halted) with a unit
mask of 0x00 (No unit mask) count 500000
warning: could not check that the binary file /lib/modules/2.6.32-
191.el6.x86_64/kernel/fs/ext4/ext4.ko has not been modified since the
profile was taken. Results may be inaccurate.
samples  %        symbol name
1622      9.8381  ext4_iget
1591      9.6500  ext4_find_entry
1231      7.4665  __ext4_get_inode_loc
783       4.7492  ext4_ext_get_blocks
752       4.5612  ext4_check_dir_entry
644       3.9061  ext4_mark_iloc_dirty
583       3.5361  ext4_get_blocks
583       3.5361  ext4_xattr_get
479       2.9053  ext4_htree_store_dirent
469       2.8447  ext4_get_group_desc
414       2.5111  ext4_dx_find_entry
```

23.5.4. opannotate の使用

opannotate ツールは特定の指示用のサンプルをソースコード内の該当する行に一致させる試みをします。その結果生成されたファイルは左側にある行のサンプルを持つはずです。これはまた、各関数の先頭にコメントを入れて関数の合計サンプル一覧表示します。

このユーティリティが機能するためには、実行可能ファイル用の適切な debuginfo パッケージがシステム上にインストールされる必要があります。Red Hat Enterprise Linux では、debuginfo パッケージは、実行可能ファイルを含む対応パッケージと自動的にインストールされません。これらは別個に入手してインストールする必要があります。

opannotate 用の一般的な構文は次のようになります:

```
~]# opannotate --search-dirs src-dir --source executable
```

ソースコード及び分析されるべき実行可能ファイルを収納しているディレクトリは指定する必要があります。その他のコマンドラインオプションの一覧については **opannotate** man ページを参照して下さい。

23.6. /DEV/OPROFILE/ を理解する

/dev/oprofile/ ディレクトリには、OProfile 用のファイルシステムが含まれています。**cat** コマンドの使用でこのファイルシステム内の仮想ファイルの値を表示することができます。例えば、以下のコマンドは OProfile が検出するプロセッサのタイプを表示します:

```
~]# cat /dev/oprofile/cpu_type
```

/dev/oprofile/ には、各カウンター用のディレクトリが存在します。例えば、２つのカウンターがある場合は、**/dev/oprofile/0/** ディレクトリと **dev/oprofile/1/** ディレクトリが存在します。

カウンター用の各ディレクトリには以下のファイルが含まれています:

- **count** — サンプル間の間隔
- **enabled** — 0 の場合は、カウンターはオフでありサンプルは収集されません。 1 の場合は、カウンターはオンとなりサンプルは収集されます。
- **event** — 監視対象のイベント
- **extra** — 監視するイベントを更に指定するために Nehalem プロセッサを持つマシン上で使用されます。
- **kernel** — 0 の場合は、プロセッサがカーネルスペース内にある時にこのカウンターイベント用のサンプルは収集されません。 1 の場合は、プロセッサがカーネルスペース内にあってもサンプルは収集されます。
- **unit_mask** — カウンター用に有効になっているユニットマスクを定義します。
- **user** — 0 の場合、プロセッサがユーザースペース内にある時にカウンターイベント用のサンプルは収集されません。 1 の場合、プロセッサがユーザースペース内にあってもサンプルは収集されます。

これらのファイルの値は **cat** コマンドで取り込むことができます。例えば:

```
~]# cat /dev/oprofile/0/count
```

23.7. 使用法の例

OProfile は開発者がアプリケーションのパフォーマンスを分析するために使用できる他、システム管理者がシステム分析を実行するのにも使用できます。例えば:

- **システム上でどのアプリケーションとサービスが最も使用されるのかを判定** — **opreport** を使用すると、アプリケーションまたはサービスがプロセッサタイムを使用する度合いを判定できます。システムが複数のサービスに使用されていて、パフォーマンスが低い場合は、最もプロセッサタイムを消費しているサービスを専用システムに移動することができます。

- **プロセッサの使用レベルを判定** — **CPU_CLK_UNHALTED** イベントを監視することにより任意の期間内でのプロセッサ負荷を判定できます。このデータはその後、追加のプロセッサまたはより高速なプロセッサがシステムのパフォーマンスを向上するかどうかの判定に使用できます。

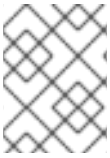
23.8. JAVA 対応の OPROFILE サポート

OProfile の使用により JVM (Java 仮想マシン) の動的にコンパイルしたコード (別名 "just-in-time" または JIT コード) をプロファイルできるようになります。Red Hat Enterprise Linux 6 内の OProfile には、JVMTI (JVM ツールインターフェイス) のエージェントライブラリのビルトインサポートが含まれています。これは Java 1.5 およびそれ以降をサポートします。

23.8.1. Java コードのプロファイル作成

JVMTI エージェントを持つ Java 仮想マシンから JIT コードをプロファイルするには、以下を JVM スタートアップパラメータに追加します:

```
-agentlib:jvmti_oprofile
```



注記

OProfile で JIT コードをプロファイルするには、システム上に oprofile-jit パッケージをインストールする必要があります。

OProfile での Java に関する詳細には、「その他のリソース」 からリンクできる OProfile マニュアルを参照して下さい。

23.9. グラフィカルインターフェース

一部の OProfile 設定項目はグラフィカルインターフェイスでセットできます。開始するには、シェルプロンプトで root として **oprof_start** コマンドを実行します。グラフィカルインターフェイスを使用するには、**oprofile-gui** パッケージをインストールする必要があります。

いずれかのオプションを変更した後には、**保存して終了** ボタンをクリックして変更を保存します。設定は **/root/.oprofile/daemonrc** に書き込まれて、アプリケーションは終了します。

注記

アプリケーションの終了は、OProfile のサンプリングを停止するものではありません。

「監視するイベントのセッティング」 で説明してあるように、**Setup** タブでプロセッサカウンター用にイベントをセットするには、プルダウンメニューからカウンターを選択してその一覧からイベントを選択します。するとそのイベントの簡単な説明が一覧の下のテキストボックスに出てきます。特定のカウンター及び特定のアーキテクチャ用に利用可能なイベントのみが表示されます。このグラフィカルインターフェイスはプロファイラーが実行中であるかどうかとその簡単な統計も表示します。

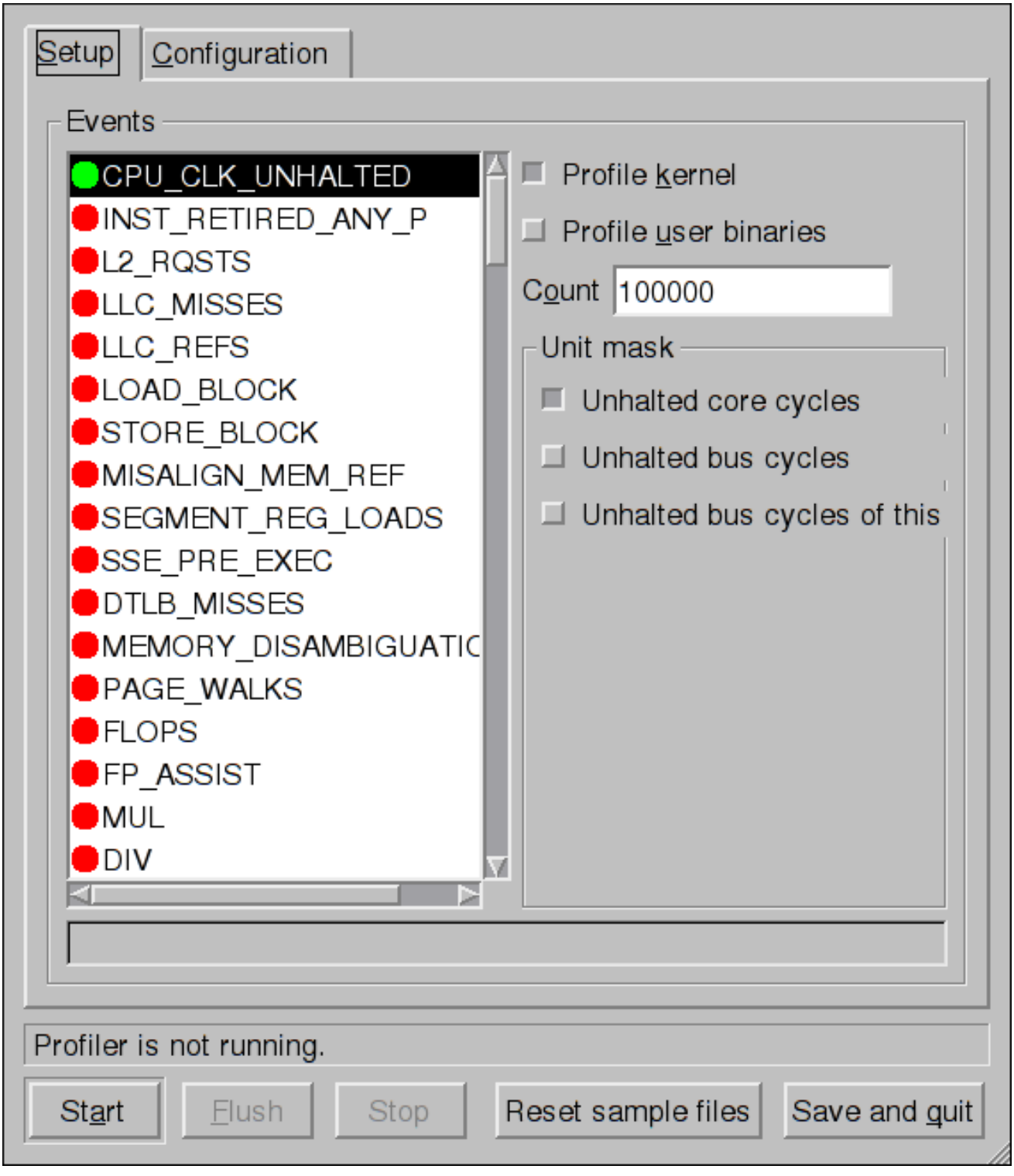


図23.1 OProfile のセットアップ

「カーネルとユーザースペースのプロファイルの分離」 で説明してあるように、タブの右側で、現在選択されているイベント用にカーネルモードでイベントをカウントするには **Profile kernel** オプションを選択します。このオプションが選択されていないと、カーネル用のサンプルは収集されません。

「カーネルとユーザースペースのプロファイルの分離」 で説明してあるように、現在選択されているイ

ベント用にユーザーモードでイベントをカウントするには **Profile user binaries** オプションを選択します。このオプションが選択されていないと、ユーザーアプリケーション用のサンプルは収集されません。

「サンプルのレート」 で説明してあるように、現在選択されているイベント用にサンプルレートをセットするには **Count** テキストフィールドを使用します。

「ユニットマスク」 で説明してあるように、現在選択されているイベント用にいずれかのユニットマスクが利用可能である場合は、それは **Setup** タブの右側にある **Unit Masks** エリアに表示されます。イベント用に有効にすべきユニットマスクの横のチェックボックスを選択します。

Configuration タブでカーネルをプロファイルするには、**Kernel image file** テキストフィールド内に監視するカーネルの名前と**vmlinux**ファイルの場所を入力します。カーネルを監視しないように OProfile を設定するには、**No kernel image** を選択します。

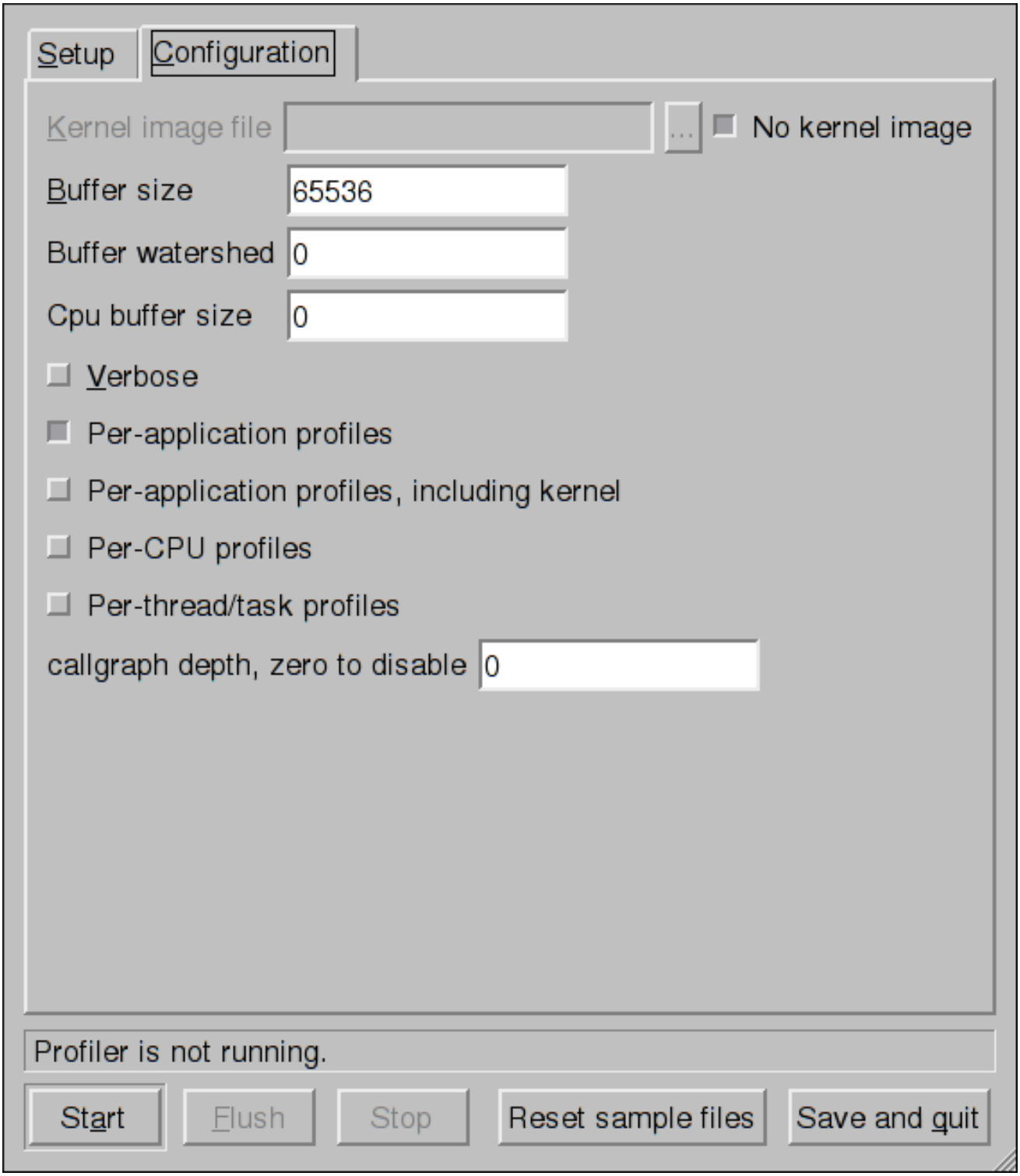


図23.2 OProfile の設定

Verbose オプションが選択されていると、**oprofiled** デーモンログはより詳細な情報を含みます。

Per-application profiles が選択されていると、OProfile はライブラリ用にアプリケーション単位のプロファイルを生成します。これは、**opcontrol --separate=library** コマンドと同じことになります。**Per-application profiles, including kernel** が選択されていると、「カーネルとユーザースペースのプロファイルの分離」 で説明してあるように OProfile はカーネルとカーネルモジュール用にアプリケーション単位のプロファイルを生成します。これは **opcontrol --separate=kernel** コマンドと同じことになります。

「データの分析」 で説明してあるように、データがサンプルファイルに書き込まれるように強制するには、**Flush** ボタンをクリックします。これは、**opcontrol --dump** コマンドと同じことになります。

グラフィカルインターフェイスから OProfile を開始するには、**Start** をクリックします。プロファイラーを停止するには、**Stop** をクリックします。アプリケーションの終了は OProfile がサンプル作成を停止することにはなりません。

23.10. OPROFILE と SYSTEMTAP

SystemTap は、追跡と探索のツールとしてユーザーがオペレーティングシステムの動作を詳細に渡って学習し監視できるようにしてくれます。これは **netstat**、**ps**、**top**、及び **iostat** などのツールの出力ににた情報を提供します。しかし SystemTap は、収集した情報に対してより多くのフィルタリングと分析オプションを提供するように設計されています。

OProfile の使用が、コードの特定エリアでプロセッサが時間を費やす場所と理由についてデータを収集するケースで推薦されますが、OProfile はプロセッサが遊休を続ける理由の発見にはあまり役立ちません。

コードの特定の場所を計測する際には SystemTap の使用が適しているでしょう。SystemTap を使用すると計測を停止したり再開したりせずにコード計測を実行できます。これは特にカーネルとデーモンの計測で役に立ちます。

SystemTap に関する詳細については、「役立つ Web サイト」 を参照して関連した SystemTap のドキュメントをご覧下さい。

23.11. その他のリソース

この章では、OProfile の設定と使用法についてのみ強調しています。より詳しい情報については以下のリソースを参照して下さい。

23.11.1. インストール済みドキュメント

- **/usr/share/doc/oprofile-*バージョン*/oprofile.html** — 『OProfile マニュアル』
- **oprofile** man ページ — **opcontrol**, **opreport**, **opannotate**, 及び**ophelp** を説明しています

23.11.2. 役立つ Web サイト

- http://oprofile.sourceforge.net/ — 最新のドキュメント、メーリングリスト、IRC チャンネル、その他を含んでいます。
- SystemTap Beginners Guide — Red Hat Enterprise Linux の異なるサブシステムを監視するための SystemTap の使用法に関する基本を詳細に説明しています。

パート VII. カーネル、モジュール、及びドライバーの設定

ここでは、管理者がカーネルをカスタマイズする場合に役立つ様々なツールを取り上げます。

第24章 手動のカーネルアップグレード

Red Hat Enterprise Linux カーネルは、サポートされているハードウェアとの整合性と互換性を確実にするために Red Hat Enterprise Linux のカーネルチームによりカスタムメイド作製されたものです。Red Hat がリリースする前に、カーネルは厳格な品質保証テスト群をパスしなければなりません。

Red Hat Enterprise Linux カーネルは RPM 形式でパッケージされていますので、**Yum** または **PackageKit** パッケージマネージャを使用して簡単にアップグレードも検証もできます。**PackageKit** は自動的に Red Hat Network サーバーにクエリを出して、カーネルパッケージを含む利用可能な更新があるパッケージをお知らせします。

そのため、この章は **yum** の代わりに **rpm** コマンドを使用して手動でカーネルパッケージを更新する必要のあるユーザーに***のみ*** 役立つと言えます。

警告

できる限り、**Yum** または **PackageKit** パッケージマネージャを使用して新しいカーネルをインストールして下さい。理由はこれらのツールは、常に新しいカーネルを **インストール** するからです。他の方法で現行バージョンを入れ替えると（失敗した場合）システムが起動できなくなる可能性が出てきます。

警告

カスタムカーネルの構築は、Red Hat Global Services Support チームではサポートされていません。そのため、このマニュアルではその記述もありません。

Yum を使ったカーネルパッケージのインストールについては、「パッケージの更新」 を参照してください。Red Hat Network についての情報は、*5章 システム登録およびサブスクリプションの管理* を参照してください。

24.1. カーネルパッケージの概要

Red Hat Enterprise Linux には、以下のカーネルパッケージが含まれています:

- kernel — シングル、マルチコア、及びマルチプロセッサシステム用のカーネルを含んでいます。
- kernel-debug — カーネル診断のために有効になっている数多くのデバッグオプションを持つカーネルを含んでいます。パフォーマンス低下の犠牲があります。
- kernel-devel — kernel パッケージに対して、モジュールを構築するのに十分なカーネルヘッダと makefiles を含んでいます。
- kernel-debug-devel — カーネル診断のために有効になっている数多くのデバッグオプションを持つ開発バージョンのカーネルを含んでいます。パフォーマンス低下の犠牲があります。

- kernel-doc — カーネルソースからのドキュメントファイルです。同梱で配布される Linux カーネルとデバイスドライバーの様々な部分がこれらのファイルで文書化されています。このパッケージをインストールすると、ロード時に Linux カーネルモジュールに渡すことの出来るオプションへの参照が提供されます。

  デフォルトでは、これらのファイルは **/usr/share/doc/kernel-doc-<*kernel_version*>/** ディレクトリに配置されています。

- kernel-headers — Linux カーネルと、ユーザースペースライブラリ及びプログラムとの間のインターフェイスを指定する C ヘッダファイルを含んでいます。このヘッダファイルはほとんどの標準プログラムを構築するのに必要な構造と定数を定義します。

- kernel-firmware — 各種デバイスが動作するために必要となるファームウェアファイルを含んでいます。

- perf — このパッケージは各カーネルイメージサブパッケージ内で配布される **perf** ツール用の支援スクリプトとドキュメントを含んでいます。

24.2. アップグレードへの準備

カーネルをアップグレードする前に、予防的な前準備手順の実行をお薦めします。

まず、システム用に機能するブートメディアがあることを確認します。ブートローダーが新しいカーネルをブートするように正しく設定されていない場合、このメディアを使って Red Hat Enterprise Linux をブートすることができます。

USB メディアは多くの場合、*ペンドライブ*とか、*サムディスク*とか、*キー* などと呼ばれるフラッシュデバイスの形式で、あるいは、外部接続のハードディスクとして提供されています。このタイプのほとんどすべてが **VFAT** ファイルシステムとしてフォーマットされています。但し、**ext2**、**ext3**、または **VFAT** としてフォーマットされているメディア上でブート可能な USB メディアを作成することができます。

ディストリビューションのイメージファイル、または最低限ブートメディア (minimal boot media) イメージを USB メディアに転送することができます。デバイスには十分な空き領域があることを確認して下さい。およそ **4 GB** がディストリビューション DVD イメージ用に必要で、およそ **700 MB** がディストリビューション CD イメージ用に、そしておよそ **10 MB** が最低限ブートメディアイメージ用に必要です。

Red Hat Enterprise Linux のインストール DVD、またはインストール CD-ROM#1 からの **boot.iso**ファイルのコピーと、およそ **16 MB** の空き領域を持つ **VFAT** ファイルシステムでフォーマットした USB ストレージデバイスが必要になります。以下の手順は、コピー先のファイルと同じパス名でない限りは USB ストレージデバイス上の既存のファイルに影響を与えません。USB ブートメディアを作成するには、root ユーザーとして以下のコマンドを実行します。

1. USB ストレージデバイス上に **SYSLINUX** ブートローダーをインストール

   ```
   ~]# syslinux /dev/sdX1
   ```

   ... ここで *sdX* はデバイス名です。

2. **boot.iso** と USB ストレージデバイス用にマウントポイントを作成します:

   ```
   ~]# mkdir /mnt/isoboot /mnt/diskboot
   ```

3. **boot.iso** をマウントします:

```
~]# mount -o loop boot.iso /mnt/isoboot
```

4. USB ストレージデバイスをマウントします:

```
~]# mount /dev/<sdX1> /mnt/diskboot
```

5. **boot.iso** から USB ストレージデバイスへ **ISOLINUX** ファイルをコピーします:

```
~]# cp /mnt/isoboot/isolinux/* /mnt/diskboot
```

6. **boot.iso** からの **isolinux.cfg** ファイルを USB デバイス用の **syslinux.cfg** ファイルとして使用します:

```
~]# grep -v local /mnt/isoboot/isolinux/isolinux.cfg >
/mnt/diskboot/syslinux.cfg
```

7. **boot.iso** と USB ストレージデバイスをアンマウントします:

```
~]# umount /mnt/isoboot /mnt/diskboot
```

8. このブートメディアでマシンを再起動して、ブート出来ることを確認してから他の操作に移ります。

別の方法としては、フロッピィディスクを持つシステム上でブートディスクを作成するには、mkbootdisk パッケージをインストールして、root として**mkbootdisk** コマンドを実行します。このパッケージをインストールした後にその使用法について、**man mkbootdisk** man ページを参照して下さい。

どのカーネルがインストールされているかを判定するには、シェルプロンプトでコマンド **yum list installed "kernel-*"** を実行します。その出力はシステムのアーキテクチャに応じて、以下のパッケージのすべてか、または一部を示すでしょう。バージョン番号も変化するかも知れません:

```
~]# yum list installed "kernel-*"
kernel.x86_64                 2.6.32-17.el6           @rhel-x86_64-
server-6
kernel-doc.noarch             2.6.32-17.el6           @rhel-x86_64-
server-6
kernel-firmware.noarch        2.6.32-17.el6           @rhel-x86_64-
server-6
kernel-headers.x86_64         2.6.32-17.el6           @rhel-x86_64-
server-6
```

この出力から、カーネルのアップグレードの為にダウンロードすべきパッケージを判断します。シングルプロセッサのシステムには、必要なパッケージは kernel パッケージのみです。異なるパッケージの説明には、「カーネルパッケージの概要」 を参照してください。

24.3. アップグレードされたカーネルのダウンロード

システム用にアップグレードされたカーネルが利用可能かを判定する手段は数種類あります。

- セキュリティ Errata — セキュリティ問題を修復するカーネルを含む、セキュリティ Errata についての詳細には、http://www.redhat.com/security/updates/ を参照して下さい。

- Red Hat Network 経由で — カーネル RPM パッケージをダウンロードしてインストールします。Red Hat Network は、最新のカーネルをダウンロードして、システム上のカーネルをアップグレードし、必要であれば初期 RAM ディスクイメージを作成してから新しいカーネルをブートするためにブートローダーを設定することができます。詳細情報には、http://www.redhat.com/docs/manuals/RHNetwork/ を参照して下さい。

カーネルのダウンロードとインストールに Red Hat Network が使用される場合は、「初期 RAM ディスクイメージの検証」 と「ブートローダーの確証」 にある案内に従います。但しカーネルをデフォルトでブートするように **変更しない** で下さい。Red Hat Network が自動的に最新バージョンをデフォルトのカーネルに変更します。手動でカーネルをインストールするには、「アップグレードの実行」 へ進みます。

24.4. アップグレードの実行

必要なパッケージをすべて取り込んだ後は、既存カーネルをアップグレードします。

重要

新しいカーネルで問題がある可能性を考慮し、古いカーネルの維持を強くお薦めします。

シェルプロンプトで、カーネル RPM パッケージを含んでいるディレクトリに移動します。**rpm** コマンドに **-i** 引数を使用して古いカーネルを残します。**-U** オプションは現在インストールしてあるカーネルを上書きして、ブートローダー問題を起こしますので、これは **使用しないでください**。例えば:

```
~]# rpm -ivh kernel-<kernel_version>.<arch>.rpm
```

次のステップとして初期 RAM ディスクイメージが作成されているかどうかを検証します。詳細には、「初期 RAM ディスクイメージの検証」 を参照して下さい。

24.5. 初期 RAM ディスクイメージの検証

初期 RAM ディスクイメージの仕事は、IDE、SCSI、または RAID などのブロックデバイスモジュールをプレロードすることです。そうすることで、それらのモジュールが通常配備されている root ファイルシステムがアクセス可能になりマウントできるようになります。Red Hat Enterprise Linux 6 システム上では、**Yum**、**PackageKit**、**RPM** パッケージマネージャのいずれかを使用してカーネルがインストールされる時は常に **Dracut** ユーティリティがインストールスクリプトによって呼び出されて、*initramfs* (初期 RAM ディスクイメージ) を作成します。

IBM eServer System i (参照「IBM eServer System i 上の初期 RAM ディスクイメージとカーネルの確証」) 以外のすべてのアーキテクチャ上では、**dracut** コマンドの実行により **initramfs** を作成できます。しかし、通常は手動で **initramfs** を作成する必要はありません。このステップは、カーネルとその関連パッケージが Red Hatで配布された RPM パッケージからインストールされているか、またはアップグレードされている場合には自動的に実行されます。

現在のカーネルバージョンに該当する **initramfs** が存在していること、及びそれが **grub.conf** 設定ファイル内で正しく指定されているかを検証するには、以下の手順に従います:

手順24.1 初期 RAM ディスクイメージの検証

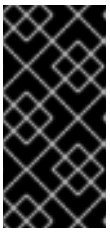
1. root として、**/boot/** ディレクトリのコンテンツを一覧表示して、カーネル (**vmlinuz-*<kernel_version>***) と最新のバージョン番号を持つ **initramfs-*<kernel_version>*** を見つけます。

例24.1 カーネルと initramfs バージョンの一致を確認

```
~]# ls /boot/
config-2.6.32-17.el6.x86_64 lost+found
config-2.6.32-19.el6.x86_64 symvers-2.6.32-17.el6.x86_64.gz
config-2.6.32-22.el6.x86_64 symvers-2.6.32-19.el6.x86_64.gz
efi symvers-2.6.32-22.el6.x86_64.gz
grub System.map-2.6.32-17.el6.x86_64
initramfs-2.6.32-17.el6.x86_64.img System.map-2.6.32-19.el6.x86_64
initramfs-2.6.32-19.el6.x86_64.img System.map-2.6.32-22.el6.x86_64
initramfs-2.6.32-22.el6.x86_64.img vmlinuz-2.6.32-17.el6.x86_64
initrd-2.6.32-17.el6.x86_64kdump.img vmlinuz-2.6.32-19.el6.x86_64
initrd-2.6.32-19.el6.x86_64kdump.img vmlinuz-2.6.32-22.el6.x86_64
initrd-2.6.32-22.el6.x86_64kdump.img
```

例24.1「カーネルと initramfs バージョンの一致を確認」 は以下を示します:

- ３つのカーネルがインストールされています (または、より正確に言うと、３つのカーネルファイルが,**/boot/** にあります)。
- 最新のカーネルは **vmlinuz-2.6.32-22.el6.x86_64** で、そして
- そのカーネルバージョンに一致する **initramfs** ファイルである **initramfs-2.6.32-22.el6.x86_64kdump.img** も存在します。



重要

/boot/ ディレクトリ内で、いくつかの **initrd-<*version*>kdump.img** ファイルを見つけることができるでしょう。これらは、カーネルデバッグ目的で **Kdump** メカニズムによって作成された特別なファイルです。そしてシステムのブートには使用されないため無視しても安全です。

2. (オプション) 使用している **initramfs-<*kernel_version*>** ファイルが、**/boot/** にある最新カーネルのバージョンと一致しない場合、または他の特定状況では、**Dracut** ユーティリティを使用して **initramfs** ファイルを生成する必要があるかも知れません。root としてオプション無しで **dracut** を呼び出すと、それが **/boot/** ディレクトリ内にある最新のカーネル用に **initramfs** ファイルを生成するようになります:

```
~]# dracut
```

既存の **initramfs** を上書きする (例えば、これまでの **initramfs** が破損しているなど) ために **dracut** を使用する場合は、**--force** オプションを使用しなければなりません。そうしないと **dracut** は既存の **initramfs** ファイルの上書きを拒否します:

```
~]# dracut
Will not override existing initramfs (/boot/initramfs-2.6.32-
22.el6.x86_64.img) without --force
```

現在のディレクトリに initramfs を作成するには、 **dracut <*initramfs_name*> <*kernel_version*>** を呼び出します:

```
~]# dracut "initramfs-$(uname -r).img" $(uname -r)
```

プレロードする特定のカーネルモジュールを指定する必要がある場合は、それらのモジュール名 (**.ko** などのファイル名の接尾辞を差し引いたもの) を **/etc/dracut.conf** 設定ファイルにある **add_dracutmodules="<*module*> [<*more_modules*>]"** 指示文の括弧内に追加します。dracut で作成された**initramfs** イメージファイルの内容は、**lsinitrd <*initramfs_file*>** コマンドを使用すると、一覧表示できます:

```
~]# lsinitrd initramfs-2.6.32-22.el6.x86_64.img
initramfs-2.6.32-22.el6.x86_64.img:
========================================================================
====
dracut-004-17.el6
========================================================================
====
drwxr-xr-x  23 root     root            0 May  3 22:34 .
drwxr-xr-x   2 root     root            0 May  3 22:33 proc
-rwxr-xr-x   1 root     root         7575 Mar 25 19:53 init
drwxr-xr-x   7 root     root            0 May  3 22:34 etc
drwxr-xr-x   2 root     root            0 May  3 22:34
etc/modprobe.d
[出力は省略されています]
```

オプションと使用法についての詳細情報には、**man dracut** と **man dracut.conf** を参照して下さい。

3. **/boot/grub/** ディレクトリ内の **grub.conf** 設定ファイルを検査して、ブートするカーネルバージョン用に **initrd initramfs-<*kernel_version*>.img** が存在することを確認します。詳細は、「ブートローダーの確認」 でご覧下さい。

IBM eServer System i 上の初期 RAM ディスクイメージとカーネルの確認

IBM eServer System i のマシン上では、初期 RAM ディスクとカーネルファイルは 1 つのファイルに統合してあり、これは **addRamDisk** コマンドで作成されます。カーネルとその関連パッケージがインストールされているか、または Red Hat で配布されている RPM パッケージでアップグレードされている場合は、このステップは自動的に実行されます。その場合、手動で実行する必要はありません。これが作成されていることを確認するには、コマンド **ls -l /boot/** を使用して**/boot/vmlinitrd-<*kernel_version*>** ファイルが存在することを確認します (**<*kernel_version*>** はインストールしたばかりのカーネルバージョンと一致する必要があります)。

24.6. ブートローダーの確認

rpm を使用してカーネルをインストールすると、カーネルパッケージはブートローダー設定ファイル内にその新しいカーネル用のエントリーを作成します。しかし、**rpm** はその新しいカーネルがデフォルトカーネルとしてブートするようには設定**しません。** **rpm** で新しいカーネルをインストールした時には手動で実行する必要があります。

rpm で新しいカーネルをインストールした後には、常にブートローダー設定ファイルをダブルチェックして設定が正しいことを確認するようにお薦めします。そうしないと、システムは Red Hat Enterprise Linux を正しくブートできないかも知れません。ブートしない場合、先に作成したブートメディアでシステムをブートしてブートローダーを再設定します。

以下の表で、使用アーキテクチャを見付けるとシステムが使用しているブートローダーを判定することができます。見付けたら、「参照 (Refer to)」リンクをクリックして使用システム用の正しい案内にジャンプします。

表24.1 アーキテクチャ別のブートローダー

アーキテクチャ	ブートローダー	参照
x86	GRUB	「GRUB ブートローダーの設定」
AMD AMD64 または Intel 64	GRUB	「GRUB ブートローダーの設定」
IBM eServer System i	OS/400	「OS/400 ブートローダーの設定」
IBM eServer System p	YABOOT	「YABOOT ブートローダーの設定」
IBM System z	z/IPL	

24.6.1. GRUB ブートローダーの設定

GRUB の設定ファイルである **/boot/grub/grub.conf** には、**default**、**timeout**、**splashimage**、**hiddenmenu**、などの指示文を持ついくつかの行が含まれています (最後の指示文は引数を持ちません)。ファイルの残りの部分は 4 行の *stanzas (スタンザ)* を含んでおり、それぞれがインストール済みのカーネルを 1 つ参照します。これらのスタンザはいつも **title** エントリーで始り、その後に関連した **root**、**kernel** 及び **initrd** の指示文が常にインデント付きになります。各スタンザが **title** で始り、それが同じスタンザの **kernel /vmlinuz-*<version_number>*** 行にあるバージョン番号と一致するバージョン番号 (括弧内) を含んでいることを確認します。

例24.2 /boot/grub/grub.conf

```
# grub.conf generated by anaconda
[comments omitted]
default=1
timeout=0
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu

title Red Hat Enterprise Linux (2.6.32-22.el6.x86_64)
        root (hd0,0)
        kernel /vmlinuz-2.6.32-22.el6.x86_64 ro
root=/dev/mapper/vg_vm6b-lv_root rd_LVM_LV=vg_vm6b/lv_root rd_NO_LUKS
rd_NO_MD rd_NO_DM LANG=en_US.UTF-8 SYSFONT=latarcyrheb-sun16
KEYBOARDTYPE=pc KEYTABLE=us rhgb quiet crashkernel=auto
        initrd /initramfs-2.6.32-22.el6.x86_64.img

title Red Hat Enterprise Linux (2.6.32-19.el6.x86_64)
        root (hd0,0)
        kernel /vmlinuz-2.6.32-19.el6.x86_64 ro
root=/dev/mapper/vg_vm6b-lv_root rd_LVM_LV=vg_vm6b/lv_root rd_NO_LUKS
rd_NO_MD rd_NO_DM LANG=en_US.UTF-8 SYSFONT=latarcyrheb-sun16
KEYBOARDTYPE=pc KEYTABLE=us rhgb quiet crashkernel=auto
        initrd /initramfs-2.6.32-19.el6.x86_64.img

title Red Hat Enterprise Linux 6 (2.6.32-17.el6.x86_64)
```

```
        root (hd0,0)
        kernel /vmlinuz-2.6.32-17.el6.x86_64 ro
root=/dev/mapper/vg_vm6b-lv_root rd_LVM_LV=vg_vm6b/lv_root rd_NO_LUKS
rd_NO_MD rd_NO_DM LANG=en_US.UTF-8 SYSFONT=latarcyrheb-sun16
KEYBOARDTYPE=pc KEYTABLE=us rhgb quiet
        initrd /initramfs-2.6.32-17.el6.x86_64.img
```

別の **/boot/** パーティションが作成されている場合、カーネルへのパスと **initramfs** イメージへのパスは **/boot/** に対して相対的になります。これが上記の 例24.2「/boot/grub/grub.conf」 内のケースです。そのため、最初のカーネルスタンザ内の **initrd /initramfs-2.6.32-22.el6.x86_64.img** の行は、root ファイルがマウントされている時には **initramfs** イメージが実際に **/boot/initramfs-2.6.32-22.el6.x86_64.img** にあることを意味します。そして、**grub.conf** の各スタンザ内のカーネルパス (例えば: **kernel /vmlinuz-2.6.32-22.el6.x86_64**) にも同じことが言えます。

注記

grub.conf 内のカーネルブートスタンザでは、**initrd** 指示文は同じカーネルバージョンに該当する **initramfs** ファイルの位置 (別のパーティションの場合、**/boot/** ディレクトリに対して相対的) を指している必要があります。この指示文は、初期 RAM ディスクイメージを作成したツール、**mkinitrd** が **initrd** と呼ばれるファイルを作成したために **initrd** と呼ばれます。そのため、**grub.conf** 指示文は、他のツールとの互換性を維持するために **initrd** であり続けます。初期 RAM ディスクイメージを作成するための **dracut** ユーティリティを使用したシステムのファイル名命名の慣習は: **initramfs-<*kernel_version*>.img** となります。

Dracut は Red Hat Enterprise Linux 6 内で使用できる新しいユーティリティであり、**mkinitrd** と比較してかなり向上しています。**Dracut** の使用法に関する詳細には、「初期 RAM ディスクイメージの検証」 を参照して下さい。

kernel /vmlinuz-<*kernel_version*> の行に提示されているカーネルバージョン番号が、各スタンザの **initrd /initramfs-<*kernel_version*>.img** 行に提示された **initramfs** イメージのバージョン番号に一致することを確認する必要があります。詳細には、手順24.1「初期 RAM ディスクイメージの検証」 を参照して下さい。

default= 指示文は GRUB に対してどのカーネルを**デフォルトで**ブートすべきかを伝えます。**grub.conf** 内の各**title** はブート可能なカーネルを示します。GRUB は、**0** から開始して、ブート可能なカーネルを示す **title** スタンザをカウントします。例24.2「/boot/grub/grub.conf」 内では、**default=1** の行は GRUB がデフォルトでブートすることを表し、これは、**2 つめ** のカーネルエントリである **title Red Hat Enterprise Linux (2.6.32-19.el6.x86_64)** です。

例24.2「/boot/grub/grub.conf」 内でバージョン番号で比較する場合、GRUB はより古いカーネルをブートするために設定されています。**grub.conf** 内の **1番め** の **title** エントリである、より新しいカーネルをブートするためには、**default** 値を**0** に変更する必要があります。

rpm を使用して新しいカーネルをインストールした後には、**/boot/grub/grub.conf** が正しいことを確認して、**default=** 値を新しいカーネルに変更します (**0** からカウントすることを忘れないで下さい)。そしてコンピュータをその新しいカーネルでブートします。ブートプロセス出力を見て、ハードウェアが検出されていることを確認します。

GRUB がエラーを表示してデフォルトカーネルでブートが出来ない場合、代替の、即ち古いカーネルでブートして見ることが一番簡単です。ブート後に問題の修復をします。

重要

grub.conf 内の **timeout** 指示文を**0** にセットすると、システムの起動時に GRUB はブート可能なカーネルの一覧を表示しなくなります。起動時にこの一覧を表示するには、BIOS 情報が出現した直後にいずれかの英数文字キーを押し続けます。すると GRUB は GRUB メニューを表示します。

別の方法として、システムのブート用に以前に作成しているブートメディアを使用します。

24.6.2. OS/400 ブートローダーの設定

カーネルをアップグレードすると、**/boot/vmlinitrd-<*kernel-version*>** ファイルがインストールされます。しかし、新しいカーネルでシステムをブートするには、**dd** コマンドを使用して設定する必要があります。

1. root としてコマンド **cat /proc/iSeries/mf/side** を発行して、デフォルトのサイド (A、B、または C) を判定します。

2. root として、以下のコマンドを発行します。ここで、*<kernel-version>* は新しいカーネルのバージョンであり、*<side>* は先のコマンドからのサイドです:

```
dd if=/boot/vmlinitrd-<kernel-version>
of=/proc/iSeries/mf/<side>/vmlinux bs=8k
```

新しいカーネルのテストとして、コンピュータを再起動してハードウェアが正常に検出されたことを確認できるメッセージを見付けます。

24.6.3. YABOOT ブートローダーの設定

IBM eServer System p はブートローダーとして YABOOT を使用します。YABOOT は **/etc/aboot.conf** を設定ファイルとして使用します。このファイルが、インストールしたばかりの kernel パッケージと同じバージョンの **image** セクションを含んでいることを確認します。**initramfs** イメージにも同じチェックをします:

```
boot=/dev/sda1 init-message=Welcome to Red Hat Enterprise Linux! Hit <TAB>
for boot options
partition=2 timeout=30 install=/usr/lib/yaboot/yaboot delay=10 nonvram
image=/vmlinuz-2.6.32-17.EL
  label=old
  read-only
  initrd=/initramfs-2.6.32-17.EL.img
  append="root=LABEL=/"
image=/vmlinuz-2.6.32-19.EL
  label=linux
  read-only
  initrd=/initramfs-2.6.32-19.EL.img
  append="root=LABEL=/"
```

デフォルトは新しいカーネルにセットされていないことに注意して下さい。最初のイメージ内のカーネルがデフォルトでブートされます。ブートするカーネルを変更するには、イメージスタンザを一覧の最初になるように移動するか、または指示文 **default** を追加して、それを新しいカーネルを含んでいるイメージスタンザの **label** にセットします。

新しいカーネルのテストとして、コンピュータを再起動してハードウェアが正常に検出されたことを確認できるメッセージを見付けます。

第25章 カーネルモジュールでの作業

Linux カーネルはモジュール式であるため、動的にロードされた *カーネルモジュール* を使用することでその機能を拡張することができます。カーネルモジュールが提供するのは、以下のとおりです。

- 新しいハードウェアに対するサポートを強化するデバイスドライバー、または
- **btrfs** や **NFS** のようなファイルシステムへのサポート

ほとんどの場合、デフォルトのパラメーターは十分に機能しますが、モジュールはカーネルのように動作をカスタマイズするパラメーターを取ることができます。ユーザースペースツールは、実行中のカーネルに現在ロードされているモジュールを一覧表示することができます。また、利用可能なパラメーターに使用できる全モジュールやモジュール固有の情報をクエリできます。さらには、実行中のカーネルに対してモジュールを動的にロード/アンロード (削除) することも可能です。こうしたユーティリティの多くは、module-init-tools パッケージにより提供され、動作の実行時にモジュールの依存関係を考慮するため、手動による依存関係の追跡が必要になることはほとんどありません。

最近のシステムでは、必要な状況が発生すると様々な仕組みによりカーネルモジュールは自動的にロードされます。ただし、モジュールを手動でロード/アンロードしなければならない場合もあります。例としては、モジュールがオプションの機能を提供する場合に、どちらのモジュールも基本的な機能を提供しながらも、あるモジュールが別のモジュールより好まれる場合、あるいはモジュールが他の状況で不正に動作する場合などです。

本章では、以下について説明します。

- ユーザースペースの module-init-tools パッケージを使用して、カーネルモジュール及びその依存関係を表示/クエリ/ロード/アンロードする方法
- コマンドラインで動的かつ永続的にモジュールパラメーターを設定して、カーネルモジュールの動作をカスタマイズする方法
- ブート時にモジュールをロードする方法

注記

本章で説明するカーネルモジュールのユーティリティを使用するには、最初に root で以下を実行して、ご使用のシステムに module-init-tools パッケージがインストールされていることを確認します:

```
~]# yum install module-init-tools
```

Yum を使用したパッケージのインストールについての詳細は、「パッケージのインストール」 を参照して下さい。

25.1. 現在ロード済みモジュールの一覧表示

lsmod コマンドを実行することで、現在カーネルにロード済みの全カーネルモジュールを一覧表示できます:

```
~]$ lsmod
Module                  Size  Used by
xfs                   803635  1
exportfs                3424  1 xfs
vfat                    8216  1
fat                    43410  1 vfat
```

```
tun                    13014  2
fuse                   54749  2
ip6table_filter         2743  0
ip6_tables             16558  1 ip6table_filter
ebtable_nat             1895  0
ebtables               15186  1 ebtable_nat
ipt_MASQUERADE          2208  6
iptable_nat             5420  1
nf_nat                 19059  2 ipt_MASQUERADE,iptable_nat
rfcomm                 65122  4
ipv6                  267017  33
sco                    16204  2
bridge                 45753  0
stp                     1887  1 bridge
llc                     4557  2 bridge,stp
bnep                   15121  2
l2cap                  45185  16 rfcomm,bnep
cpufreq_ondemand        8420  2
acpi_cpufreq            7493  1
freq_table              3851  2 cpufreq_ondemand,acpi_cpufreq
usb_storage            44536  1
sha256_generic         10023  2
aes_x86_64              7654  5
aes_generic            27012  1 aes_x86_64
cbc                     2793  1
dm_crypt               10930  1
kvm_intel              40311  0
kvm                   253162  1 kvm_intel
[出力は省略されています]
```

lsmod 出力の各行の定義:

- メモリに現在ロード済みのカーネルモジュールの名前
- カーネルモジュールが使用するメモリ量
- モジュールとそれに依存する他のモジュールを使用するプロセスの合計に加えて、ある場合はそれらモジュールの名前が記載された一覧。この一覧を使用すると、アンロードしたいモジュールに依存するすべてのモジュールを最初にアンロードできます。詳細については、「モジュールのアンロード」 を参照して下さい。

最後に注目する点としては、**lsmod** 出力は **/proc/modules** 擬似ファイルの内容よりも大まかで、かなり読みやすいことです。

25.2. モジュール情報の表示

modinfo *<module_name>* コマンドを実行すると、カーネルモジュールに関する詳しい情報を表示できます。

注記

module-init-tools ユーティリティの 1 つに引数としてカーネルモジュールの名前を入力する場合、拡張子である **.ko** を名前の最後に付けないで下さい。カーネルモジュールの名前に拡張子は付きませんが、それらの対応するファイルには付いています。

例として、Intel PRO/1000 ネットワークドライバーである **e1000e** モジュールに関する情報を表示するには、以下を実行します:

例25.1 lsmod を使用したカーネルモジュール情報の一覧表示

```
~]# modinfo e1000e
filename:       /lib/modules/2.6.32-
71.el6.x86_64/kernel/drivers/net/e1000e/e1000e.ko
version:        1.2.7-k2
license:        GPL
description:    Intel(R) PRO/1000 Network Driver
author:         Intel Corporation, <linux.nics@intel.com>
srcversion:     93CB73D3995B501872B2982
alias:          pci:v00008086d00001503sv*sd*bc*sc*i*
alias:          pci:v00008086d00001502sv*sd*bc*sc*i*
[alias の数行は省略]
alias:          pci:v00008086d0000105Esv*sd*bc*sc*i*
depends:
vermagic:       2.6.32-71.el6.x86_64 SMP mod_unload modversions
parm:           copybreak:Maximum size of packet that is copied to a
new buffer on receive (uint)
parm:           TxIntDelay:Transmit Interrupt Delay (array of int)
parm:           TxAbsIntDelay:Transmit Absolute Interrupt Delay (array
of int)
parm:           RxIntDelay:Receive Interrupt Delay (array of int)
parm:           RxAbsIntDelay:Receive Absolute Interrupt Delay (array
of int)
parm:           InterruptThrottleRate:Interrupt Throttling Rate (array
of int)
parm:           IntMode:Interrupt Mode (array of int)
parm:           SmartPowerDownEnable:Enable PHY smart power down (array
of int)
parm:           KumeranLockLoss:Enable Kumeran lock loss workaround
(array of int)
parm:           WriteProtectNVM:Write-protect NVM [WARNING: disabling
this can lead to corrupted NVM] (array of int)
parm:           CrcStripping:Enable CRC Stripping, disable if your BMC
needs the CRC (array of int)
parm:           EEE:Enable/disable on parts that support the feature
(array of int)
```

以下は、**modinfo** 出力にある一部のフィールドの説明です。

filename

.ko カーネルオブジェクトファイルへの絶対パスです。**modinfo -n** は **filename** フィールドのみを表示するショートカットコマンドとして使用できます。

description

モジュールに関する簡単な説明です。**modinfo -d** は、詳細フィールドのみを表示するショートカットコマンドとして使用できます。

alias

alias フィールドは、モジュールにエイリアスがある回数だけ表示されます。ない場合は、完全に省略されます。

depends

このフィールドには、このモジュールが依存するすべてのモジュールのカンマで区切られた一覧が含まれます。

注記

モジュールに依存関係がない場合は、出力から **depends** フィールドは省略される場合があります。

parm

parm フィールドは、***parameter_name*:*description*** の形式で 1 つのモジュールパラメーターを表します。

- *parameter_name* は、コマンドラインまたは **/etc/modprobe.d/** ディレクトリの **.conf** ファイルのオプション行でモジュールパラメーターとして使用している場合に使用する構文と同一です。
- *description* は、パラメーターが使用する括弧内の値の種類 (int、unit、array of int など) に対する期待値と併せて、パラメーターが実行する内容の簡単な説明です。

-p オプションを使用すると、モジュールが対応するすべてのパラメーターを一覧表示できます。ただし、値のタイプを含む有用な情報は **modinfo -p** 出力から省略されるため、以下を実行した方がより役立ちます。

例25.2 モジュールパラメーターの一覧表示

```
~]# modinfo e1000e | grep "^parm" | sort
parm:           copybreak:Maximum size of packet that is copied to a
new buffer on receive (uint)
parm:           CrcStripping:Enable CRC Stripping, disable if your
BMC needs the CRC (array of int)
parm:           EEE:Enable/disable on parts that support the feature
(array of int)
parm:           InterruptThrottleRate:Interrupt Throttling Rate
(array of int)
parm:           IntMode:Interrupt Mode (array of int)
parm:           KumeranLockLoss:Enable Kumeran lock loss workaround
(array of int)
parm:           RxAbsIntDelay:Receive Absolute Interrupt Delay (array
of int)
parm:           RxIntDelay:Receive Interrupt Delay (array of int)
parm:           SmartPowerDownEnable:Enable PHY smart power down
(array of int)
parm:           TxAbsIntDelay:Transmit Absolute Interrupt Delay
(array of int)
parm:           TxIntDelay:Transmit Interrupt Delay (array of int)
parm:           WriteProtectNVM:Write-protect NVM [WARNING: disabling
this can lead to corrupted NVM] (array of int)
```

25.3. モジュールのロード

カーネルモジュールをロードするには、root で **modprobe *<module_name>*** を実行します。例えば、**wacom** モジュールをロードするには、以下を実行して下さい:

```
~]# modprobe wacom
```

デフォルトでは、**modprobe** は **/lib/modules/*<kernel_version>*/kernel/drivers/** よりモジュールのロードを試行します。このディレクトリ内には、各種モジュールにそれぞれのサブディレクトリがあります。例えば、ネットワークドライバーの場合は **net/**、SCSI インターフェースドライバーの場合は **scsi/** です。

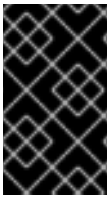
一部のモジュールは依存関係を持っています、それらの依存関係は、モジュールがロードされる前にロードする必要がある他のカーネルモジュールです。**modprobe** コマンドは、動作の実行時に常に依存関係を考慮に入れます。**modprobe** に特定のカーネルモジュールをロードするよう要求する場合、そのモジュールの依存関係があれば最初にそれらを調べ、カーネルにまだロードされていない場合はロードします。**modprobe** は、再帰的に依存関係を解決します。つまり、依存関係が持つすべての依存関係をロード、さらにはそれらの依存関係をロードと、必要に応じてロードしていきます。これにより、すべての依存関係が常に対応しているようにします。

-v (または **--verbose**) オプションを使用すると、**modprobe** が実行している内容の詳細を表示することができます。これには、モジュールの依存関係のロードが含まれる場合があります。以下は、**Fibre Channel over Ethernet** モジュールを詳細にロードする例です。

例25.3 モジュールの依存関係がロードされる場合の modprobe -v による表示

```
~]# modprobe -v fcoe
insmod /lib/modules/2.6.32-71.el6.x86_64/kernel/drivers/scsi/scsi_tgt.ko
insmod /lib/modules/2.6.32-
71.el6.x86_64/kernel/drivers/scsi/scsi_transport_fc.ko
insmod /lib/modules/2.6.32-
71.el6.x86_64/kernel/drivers/scsi/libfc/libfc.ko
insmod /lib/modules/2.6.32-
71.el6.x86_64/kernel/drivers/scsi/fcoe/libfcoe.ko
insmod /lib/modules/2.6.32-
71.el6.x86_64/kernel/drivers/scsi/fcoe/fcoe.ko
```

例25.3「モジュールの依存関係がロードされる場合の modprobe -v による表示」 は、**modprobe** が **fcoe** を最後にロードする前に、**scsi_tgt**、**scsi_transport_fc**、**libfc**、**libfcoe** のモジュールを依存関係としてロードしたことを示しています。また、**modprobe** がさらに 「プリミティブな」 **insmod** コマンドを使用して、実行中のカーネルにモジュールを挿入した点にも注目して下さい。



重要

insmod コマンドを使用してカーネルモジュールをロードすることもできますが、依存関係は解決しません。そのため、代わりに **modprobe** を **常に** 使用して、モジュールをロードすることをお勧めします。

25.4. モジュールのアンロード

カーネルモジュールをアンロードするには、root として **modprobe -r <*module_name*>** を実行します。例えば、カーネルに **wacom** モジュールがすでにロードされていると仮定した場合、そのモジュールをアンロードするには以下を実行します:

```
~]# modprobe -r wacom
```

ただし、プロセスが以下のいずれかを使用している場合は、このコマンドは失敗します。

- **wacom** モジュール
- **wacom** が直接依存するモジュール
- 依存関係ツリーにより **wacom** が直接依存するすべてのモジュール

特定のモジュールをアンロードできないようにしているモジュール名を取得するための、**lsmod** の使用に関する詳細は、「現在ロード済みモジュールの一覧表示」 を参照して下さい。

例として、**firewire_ohci** モジュールをアンロードしたい場合 (例えば、システムの安定性に影響を与えるバグがあると考えている場合) は、ターミナルセッションは以下のようになります:

```
~]# modinfo -F depends firewire_ohci
depends:        firewire-core
~]# modinfo -F depends firewire_core
depends:        crc-itu-t
~]# modinfo -F depends crc-itu-t
depends:
```

これで、ロードされた Firewire モジュールに対する依存関係ツリーが明確になりました (この例では分岐していません)。**firewire_ohci** は **firewire_core** に依存しており、**firewire_core** は **crc-itu-t** に依存しています。

firewire_ohci をアンロードするには、**modprobe -v -r <*module_name*>** コマンドを使用します。**-r** は **--remove** を、**-v** は **--verbose** をそれぞれ省略したものです。

```
~]# modprobe -r -v firewire_ohci
rmmod /lib/modules/2.6.32-71.el6.x86_64/kernel/drivers/firewire/firewire-
ohci.ko
rmmod /lib/modules/2.6.32-71.el6.x86_64/kernel/drivers/firewire/firewire-
core.ko
rmmod /lib/modules/2.6.32-71.el6.x86_64/kernel/lib/crc-itu-t.ko
```

アンロードされるモジュールに依存するプロセスがない場合、モジュールはロードされた時と逆の順番でアンロードされることをこの出力は示しています。

重要

rmmod コマンドを使用してカーネルモジュールをアンロードすることはできますが、その代わりに **modprobe -r** を使用することが推奨されます。

25.5. モジュールパラメーターの設定

カーネルのように、モジュールは動作を変更するパラメーターを取ることもできます。大半の場合、デフォルトのパラメーターは十分動作しますが、モジュールに対しカスタムパラメーターを設定すること

が必要な、あるいは望ましい場合があります。実行中のカーネルにすでにロードされたモジュールに対してパラメーターを動的に設定することはできないため、それらを設定する方法は 2 つあります。

1. パラメーターを設定したいモジュールが持つすべての依存関係のアンロード、**modprobe -r** を使用したモジュールのアンロード、そしてカスタマイズしたパラメーターの一覧と共に **modprobe** を使用したモジュールのロードを実行することができます。この方法がよく使用されるのは、モジュールに多くの依存関係がない場合、または様々なパラメーターの組み合わせを永続的にすることなくテスト目的で使用される場合です。これが本項で取り上げる方法です。

2. 別の方法として、**/etc/modprobe.d/** ディレクトリ内にある既存または新規作成のファイルに新しいパラメーターを一覧表示することもできます。モジュールがロードされるたびにモジュールパラメーターが設定されていることを確認することで、モジュールパラメーターを永続的にできます。例えば、毎回リブートした後や **modprobe** コマンドを使用した後などです。次の情報が前提条件ですが、この方法は 「永続的なモジュールのロード」 で説明します。

以下のコマンドライン形式を使用して、**modprobe** によりカスタムパラメーターでカーネルモジュールをロードできます:

例25.4 カーネルモジュールのロード時にオプションのパラメーターを指定

```
~]# modprobe <module_name> [parameter=value]
```

コマンドラインでカスタムパラメーターを使ってモジュールをロードする場合は、以下の点に注意して下さい。

- パラメーターと値を空白で区切ることで、複数入力が可能です。
- 引数としてカンマで区切られた値の一覧を必要とするモジュールパラメーターもあります。値の一覧を入力する場合、カンマの後に空白を **入力しないで下さい**。入力してしまうと、**modprobe** は空白に続く値を追加のパラメーターと間違って解釈することになります。
- 以下のいずれかの場合、**modprobe** コマンドは、成功すると何も表示せずにステータスは **0** という終了に変わります。
  - モジュールのロードが成功、**または**
  - モジュールが **すでに** カーネルにロード済み

  そのため、カスタムパラメーターを使ったモジュールのロード試行前に、モジュールがまだロードされていないことを確認する必要があります。**modprobe** コマンドは、モジュールの自動的な再ロードやすでにモジュールがロードされている旨の警告を行うことはありません。

カスタムパラメーターの設定及びカーネルモジュールのロードを行うにあたって推奨される手順は以下のとおりです。これは、例として Intel PRO/1000 ネットワークアダプター用のネットワークドライバーである **e1000e** モジュールを使用した手順について説明しています。

手順25.1 カスタムパラメーターによるカーネルモジュールのロード

1. 最初に、モジュールがカーネルにまだロードされていないことを確認します:

```
~]# lsmod |grep e1000e
~]#
```

出力は、モジュールがカーネルにすでにロードされていることを示しています。この場合は、続行前にまずそれをアンロードする必要があります。安全にアンロードする方法については、「モジュールのアンロード」 を参照して下さい。

2. モジュールをロードして、モジュール名の後にすべてのカスタムパラメーターを表示します。例えば、 Energy Efficient Ethernet (EEE) が有効な状態で、割り込みスロットル値をドライバーの第 1、第 2、第 3 インスタンスに対して毎秒 3000 割り込みに設定した Intel PRO/1000 ネットワークドライバーをロードしたい場合は、[6]root で以下を実行します:

```
~]# modprobe e1000e InterruptThrottleRate=3000,3000,3000 EEE=1
```

この例では、カンマで区切り空白を省くことにより、1 つのパラメーターに複数の値を渡すことを示しています。

25.6. 永続的なモジュールのロード

例25.1「lsmod を使用したカーネルモジュール情報の一覧表示」 のとおり、カーネルモジュールの多くはブート時に自動的にロードされます。**/etc/sysconfig/modules/** ディレクトリ内に新しい **<*file_name*>.modules** ファイルを作成することで、ロードする追加のモジュールを指定できます。*<file_name>* は、分かりやすければどのような名前でも構いません。**<*file_name*>.modules** ファイルは、システムのスタートアップスクリプトによりシェルスクリプトとして扱われ、1 行目は「感嘆符の行」 とも呼ばれる *interpreter directive (解釈実行プログラム指示文)* で始まる必要があります。

例25.5 *file_name*.modules ファイルの 1 行目

```
#!/bin/sh
```

また、**<*file_name*>.modules** ファイルは実行可能でなければなりません。実行可能なファイルに設定する方法は、以下のとおりです:

```
modules]# chmod +x <file_name>.modules
```

例えば、以下の **bluez-uinput.modules** スクリプトは **uinput** モジュールをロードします。

例25.6 /etc/sysconfig/modules/bluez-uinput.modules

```
#!/bin/sh

if [ ! -c /dev/input/uinput ] ; then
        exec /sbin/modprobe uinput >/dev/null 2>&1
fi
```

3 行目の **if** の条件付きステートメントにより、**/dev/input/uinput** ファイルが **存在していない** ようにします (**!** の記号は条件を無効にします)。その場合は、**exec /sbin/modprobe uinput** を呼び出すことで **uinput** モジュールをロードします。**uinput** モジュールは **/dev/input/uinput** ファイルを作成するため、そのファイルが存在するかどうか調べることは、**uinput** モジュールがカーネルにロードされているかどうかを確認することを意味します。

その行の最後にある **>/dev/null 2>&1** は、すべての出力を **/dev/null** にリダイレクトするため、**modprobe** コマンドは使用されません。

25.7. カーネルモジュール特有の機能

本項は、各種カーネルモジュールを使用してカーネル特有の機能を有効にする方法について説明します。

25.7.1. 複数のイーサネットカードの使用

単一のマシンで複数のイーサネットカードを使用することができます。それぞれカードには **alias** と、場合によっては各カードごとに **/etc/modprobe.d/** ディレクトリにあるユーザーが作成した ***<module_name>*.conf** ファイルに **options** の行が必要です。

複数のイーサネットカードの使用に関する追加情報は、http://www.redhat.com/mirrors/LDP/HOWTO/Ethernet-HOWTO.html にあるオンラインの 『Linux Ethernet-HOWTO』 を参照して下さい。

25.7.2. チャンネルボンディングの使用

Red Hat Enterprise Linux では、管理者はカーネルモジュールである **bonding** と *チャンネルボンディングインターフェース* と呼ばれる特殊なネットワークインターフェースを使用して、複数の NIC を単一のチャンネルにバインドすることが可能です。チャンネルボンディングにより、2 つもしくはそれ以上のネットワークインターフェースが 1 つとして機能できると同時に、帯域幅が増加し冗長性を実現します。

複数のネットワークインターフェースのチャンネルボンディングを行うには、管理者は以下の手順を実行する必要があります。

1. root として **/etc/modprobe.d/** ディレクトリ内に ***<bonding>*.conf** と呼ばれる新しいファイルを作成します。最後に **.conf** の拡張子が付く限りはどのような名前でも構いません。この新しいファイルに以下の行を挿入します:

   ```
   alias bond<N> bonding
   ```

   <N> を **0** などのインターフェースの番号に置き換えます。設定済みの各チャンネルボンディングインターフェースについては、新しい **/etc/modprobe.d/*<bonding>*.conf** ファイルに対応するエントリがなければなりません。

2. 「チャンネルボンディングインターフェース」 で説明のとおりに、チャンネルボンディングインターフェースを設定します。

3. パフォーマンスを強化するには、利用可能なモジュールオプションを調節して、最適な組み合わせを確認します。特に、**miimon** パラメーター、あるいは **arp_interval** 及び **arp_ip_target** パラメーターに注意して下さい。利用可能なオプションの一覧と使用しているボンディングされたインターフェースに最適なオプションを迅速に決定する方法については、「bonding モジュールの指示文」 を参照して下さい。

25.7.2.1. bonding モジュールの指示文

使用しているボンディングインターフェース設定ファイル (例えば **ifcfg-bond0**) の ***BONDING_OPTS="<bonding parameters>"*** 指示文にチャンネルボンディングのモジュールパラメーターを追加する前に、使用しているボンディングされたインターフェースにどれが最適かを検証するこ

とは好ましい方法です。ボンディングされたインターフェースのパラメーターは、**sysfs** ファイルシステム内のファイルを操作することで、bonding モジュールをアンロード (及び再ロード) することなく設定できます。

sysfs は仮想ファイルシステムであり、カーネルオブジェクトをディレクトリ、ファイル、シンボリックリンクとして表示します。**sysfs** を使用すると、カーネルオブジェクトの情報をクエリでき、通常のファイルシステムのコマンドを使用することでそうしたオブジェクトを操作することも可能です。**/etc/fstab** には、**sysfs** 仮想ファイルシステムの行が 1 行あり、**/sys/** ディレクトリ下にマウントされます。**/sys/class/net/** ディレクトリ下にあるファイルをやりとり、操作することですべてのボンディングインターフェースを動的に設定することができます。

使用しているボンディングインターフェースに最適なパラメーターを決定するためには、「チャンネルボンディングインターフェース」 にある手順に従って **ifcfg-bond0** などのチャンネルボンディングインターフェースのファイルを作成します。bond0 にボンディングされている各インターフェースの設定ファイルに ***SLAVE=yes*** 及び ***MASTER=bond0*** 指示文を挿入します。これが完了すると、パラメーターの確認に進むことができます。

最初に root で **ifconfig bond<*N*> up** を実行して、作成した bond を有効にします:

```
~]# ifconfig bond0 up
```

ifcfg-bond0 のボンディングインターフェースのファイルが正しく作成されたら、**ifconfig** (オプションなし) を実行した出力に **bond0** が表示されていることを確認できます:

```
~]# ifconfig
bond0     Link encap:Ethernet HWaddr 00:00:00:00:00:00
          UP BROADCAST RUNNING MASTER MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
eth0      Link encap:Ethernet  HWaddr 52:54:00:26:9E:F1
          inet addr:192.168.122.251  Bcast:192.168.122.255
Mask:255.255.255.0
          inet6 addr: fe80::5054:ff:fe26:9ef1/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:207 errors:0 dropped:0 overruns:0 frame:0
          TX packets:205 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:70374 (68.7 KiB)  TX bytes:25298 (24.7 KiB)
[出力は省略されています]
```

無効の bond も含めてすべての既存の bond を表示するには、以下を実行します:

```
~]# cat /sys/class/net/bonding_masters
bond0
```

それぞれの bond を個々に設定するには、**/sys/class/net/bond<*N*>/bonding/** ディレクトリにあるファイルを操作します。まずは、設定する bond を停止しなければなりません:

```
~]# ifconfig bond0 down
```

例として、bond0 の MII 監視を 1 秒間隔で有効にするには、(root で) 以下を実行します:

```
~]# echo 1000 > /sys/class/net/bond0/bonding/miimon
```

balance-alb モードに bond0 を設定するには、以下のいずれかを実行します:

```
~]# echo 6 > /sys/class/net/bond0/bonding/mode
```

あるいは、モードの名前を使用します:

```
~]# echo balance-alb > /sys/class/net/bond0/bonding/mode
```

bond にオプションを設定後、**ifconfig bond<*N*> up** を実行することで、bond を有効にしてテストします。オプションを変更する場合は、インターフェースを停止して、**sysfs** を使用してそのパラメーターを修正後、有効に戻して再確認します。

使用している bond に最適なパラメーターのセットを決定したら、設定しているボンディングインターフェースの **/etc/sysconfig/network-scripts/ifcfg-bond<*N*>** ファイルの ***BONDING_OPTS=*** 指示文に空白で区切った一覧としてそれらのパラメーターを追加します。bond が有効な場合 (例えば、***ONBOOT=yes*** 指示文が設定された場合にブートシーケンス中にシステムにより bond が有効な場合) はいつでも、***BONDING_OPTS*** で指定されるボンディングオプションはその bond に対して有効となります。ボンディングインターフェース (及び ***BONDING_OPTS***) の設定についての詳細は、「チャンネルボンディングインターフェース」 を参照して下さい。

以下の一覧には、一般的な多くのチャンネルボンディングパラメーターの名前が、それらの機能の詳細と併せて記載されています。詳細については、**modinfo bonding** 出力の各 **parm** の簡単な説明、または kernel-doc パッケージの **bonding.txt** ファイルにある包括的な説明を参照して下さい (「その他のリソース」 を参照)。

ボンディングインターフェースパラメーター

arp_interval=<*time_in_milliseconds*>

ARP 監視が発生する頻度を指定します (ミリ秒単位)。

重要

arp_interval 及び **arp_ip_target** の両パラメーター、あるいは **miimon** パラメーターを指定する必要があります。そうしないと、リンクが失敗した場合にネットワークパフォーマンスが低下する恐れがあります。

mode=0 または **mode=1** (2 つの負荷分散モード) でこの設定を使用する場合、ネットワークスイッチは NIC 全体に均等にパケットを分散するよう設定する必要があります。この方法の詳細については、**/usr/share/doc/kernel-doc-<*kernel_version*>/Documentation/networking/bonding.txt** を参照して下さい。

デフォルトでは値は **0** に設定されており、ARP 監視を無効にします。

arp_ip_target=<*ip_address*>[,<*ip_address_2*>,…<*ip_address_16*>]

arp_interval パラメーターが有効な場合の ARP 要求のターゲット IP アドレスを指定します。カンマで区切られた一覧で最大 16 個の IP アドレスを指定できます。

arp_validate=<*value*>

ARP プローブのソース/ディストリビューションを検証します。デフォルトは **none** です。他の有効な値は、**active**、**backup**、**all** です。

debug=<*number*>

デバッグメッセージを有効にします。指定可能な値は、以下のとおりです。

- **0** — デバッグメッセージは無効です。これがデフォルトです。
- **1** — デバッグメッセージは有効です。

downdelay=<*time_in_milliseconds*>

リンクを無効にする前に、リンクの失敗後に待機する時間を指定します (ミリ秒単位)。値は、**miimon** パラメーターで指定される値の倍数でなければなりません。デフォルトでは **0** に設定されており、待機時間を無効にします。

lacp_rate=<*value*>

リンクパートナーが 802.3ad モードで LACPDU パケットを送信するレートを指定します。指定可能な値は、以下のとおりです。

- **slow** または **0** — デフォルト設定です。パートナーが 30 秒ごとに LACPDU を送信するよう指定します。
- **fast** または **1** — パートナーが 1 秒ごとに LACPDU を送信するよう指定します。

miimon=<*time_in_milliseconds*>

MII リンク監視が発生する頻度を指定します (ミリ秒単位)。MII は NIC がアクティブであることを検証するために使用されるため、これは高可用性が必要な場合に役立ちます。root で以下のコマンドを入力して、特定の NIC のドライバーが MII ツールに対応していることを確認します:

```
~]# ethtool <interface_name> | grep "Link detected:"
```

このコマンドで、bond インターフェースではなく **eth0** のように *<interface_name>* をデバイスインターフェースの名前に置換します。MII が対応している場合は、コマンドは以下を返します:

```
Link detected: yes
```

高可用性のためにボンディングされたインターフェースを使用する場合、各 NIC のモジュールは MII に対応していなければなりません。値を **0** (デフォルト) に設定すると、この機能はオフになります。この設定を行う場合、このパラメーターの適切な開始ポイントは **100** です。

重要

arp_interval 及び **arp_ip_target** の両パラメーター、あるいは **miimon** パラメーターを指定する必要があります。そうしないと、リンクが失敗した場合にネットワークパフォーマンスが低下する恐れがあります。

mode=<*value*>

ボンディングポリシーを指定できます。*<value>* は以下のいずれかになります。

- **balance-rr** または **0** — 耐障害性とロードバランシングのためラウンドロビンポリシーを設定します。利用可能な第 1 のインターフェースからそれぞれのボンディングされたスレーブインターフェースで送受信が順次行われます。

- **active-backup** または **1** — 耐障害性のためアクティブなバックアップポリシーを設定します。利用可能な第 1 のボンディングされたスレーブインターフェースにより送受信が行われます。別のボンディングされたスレーブインターフェースは、アクティブなボンディングされたスレーブインターフェースが失敗した場合にのみ使用されます。

- **balance-xor** または **2** — 耐障害性とロードバランシングのため XOR (排他的論理和) ポリシーを設定します。この方法を使用すると、インターフェースによって受信要求の MAC アドレスとスレーブ NIC の 1 つの MAC アドレスが一致します。このリンクが確立すると、利用可能な第 1 のインターフェースから送信が順次行われます。

- **broadcast** または **3** — 耐障害性のためブロードキャストポリシーを設定します。すべての送信は、すべてのスレーブインターフェースで行われます。

- **802.3ad** または **4** — IEEE 802.3ad 動的リンクアグリゲーションのポリシーを設定します。同一の速度とデュプレックス設定を共有するアグリゲーショングループを作成します。アクティブなアグリゲーターのすべてのスレーブで送受信を行います。802.3ad に対応するスイッチが必要です。

- **balance-tlb** または **5** — 耐障害性とロードバランシングのため送信ロードバランシング (TLB) ポリシーを設定します。発信トラフィックは、各スレーブインターフェースの現在の負荷に従って分散されます。受信トラフィックは、現在のスレーブにより受信されます。受信しているスレーブが失敗すると、別のスレーブが失敗したスレーブの MAC アドレスを引き継ぎます。

- **balance-alb** または **6** — 耐障害性とロードバランシングのためアクティブロードバランシング (ALB) ポリシーを設定します。IPV4 トラフィック用の送受信ロードバランシングが含まれます。ARP ネゴシエーションにより、受信ロードバランシングが可能です。

num_unsol_na=<*number*>

フェイルオーバーイベント後に発行される未承諾の IPv6 近隣通知の数を指定します。1 つの未承諾 NA がフェイルオーバーの直後に発行されます。

有効な範囲は **0 - 255** です。デフォルト値は **1** です。このパラメーターは、アクティブなバックアップモードのみに適用します。

primary=<*interface_name*>

eth0 のようなプライマリデバイスのインターフェース名を指定します。**primary** デバイスは、使用される最初のボンディングインターフェースであり、失敗しない限りは破棄されません。この設定が特に役立つのは、ボンディングインターフェースの NIC の 1 つが高速なため、大規模な負荷に対応できる場合です。

この設定は、ボンディングインターフェースが **active-backup** モードの場合にのみ有効です。詳細については、 **/usr/share/doc/kernel-doc-<*kernel-version*>/Documentation/networking/bonding.txt** を参照して下さい。

primary_reselect=<*value*>

プライマリスレーブに対して再選択ポリシーを指定します。これは、アクティブなスレーブの失敗やプライマリスレーブの回復が発生した場合に、どのようにプライマリスレーブが選択されてアクティブなスレーブになるかという点に影響します。このパラメーターは、プライマリスレーブと他

のスレーブ間でのフリップフロップを防ぐように設計されています。指定可能な値は、以下のとおりです。

- **always** または **0** (デフォルト) — プライマリスレーブは有効になるといつでもアクティブなスレーブになります。
- **better** または **1** — プライマリスレーブの速度とデュプレックスが、現在のアクティブなスレーブの速度とデュプレックスと比べて良い場合は、プライマリスレーブは有効になるとアクティブなスレーブになります。
- **failure** または **2** — 現在のアクティブなスレーブが失敗してプライマリスレーブが有効になる場合のみ、プライマリスレーブはアクティブなスレーブになります。

primary_reselect の設定は、以下の 2 つの場合では無視されます。

- アクティブなスレーブがない場合は、回復する最初のスレーブがアクティブなスレーブになります。
- 初めにプライマリスレーブがスレーブにされた場合は、それは常にアクティブなスレーブになります。

sysfs により **primary_reselect** ポリシーを変更すると、新しいポリシーに従って直ちに最良のアクティブなスレーブを選択するようになります。これにより、状況によってはアクティブなスレーブに変更が生じる場合があります。

updelay=<*time_in_milliseconds*>

リンクを有効にする前の待機時間を指定します (ミリ秒単位)。値は、**miimon** パラメーターで指定される値の倍数でなければなりません。デフォルトでは、値は **0** に設定されており、待機時間を無効にします。

use_carrier=<*number*>

リンク状態を決定するために **miimon** が MII/ETHTOOL ioctls または **netif_carrier_ok()** を使用するかどうか指定します。**netif_carrier_ok()** 機能は、デバイスドライバーを使用して **netif_carrier_*on/off*** によりその状態を維持します。大半のデバイスドライバーはこの機能に対応しています。

MII/ETHROOL ioctls ツールは、カーネル内の非推奨の呼び出しシーケンスを活用します。ただし、これは使用しているデバイスドライバーが **netif_carrier_*on/off*** に対応しない場合でも設定可能です。

有効な値は次のとおりです。

- **1** — デフォルト設定です。**netif_carrier_ok()** の使用を有効にします。
- **0** — MII/ETHTOOL ioctls の使用を有効にします。

注記

リンクが有効であるべきではない時に有効であるとボンディングインターフェースが主張した場合、使用しているネットワークデバイスドライバーは **netif_carrier_*on/off*** に対応しない可能性があります。

xmit_hash_policy=<*value*>

balance-xor 及び **802.3ad** モードで、スレーブを選択する時に使用する送信ハッシュポリシーを選択します。指定可能な値は、以下のとおりです。

- **0** または **layer2** — デフォルト設定です。このパラメーターは、ハードウェア MAC アドレスの XOR を使用してハッシュを生成します。使用する公式は以下のとおりです:

  ```
  (<source_MAC_address> XOR <destination_MAC>) MODULO <slave_count>
  ```

  このアルゴリズムは、すべてのトラフィックを同じスレーブの特定のネットワークピアに割り振り、802.3ad に対応します。

- **1** または **layer3+4** — 上位レイヤープロトコルの情報を (利用可能な場合は) 使用して、ハッシュを生成します。これにより、特定のネットワークピアへのトラフィックが複数のスレーブに及ぶようにできますが、単一の接続では複数のスレーブに及びません。

  断片化された TCP 及び UDP パケットに使用される公式は、以下のとおりです:

  ```
  ((<source_port> XOR <dest_port>) XOR
    ((<source_IP> XOR <dest_IP>) AND 0xffff)
      MODULO <slave_count>
  ```

  断片化された TCP または UDP パケットと他の全 IP プロトコルトラフィックについては、送信元及び宛先ポート情報が省略されます。非 IP トラフィックに関しては、公式は **layer2** 送信ハッシュポリシーと同じです。

  このポリシーの目的は、特に PFC2 付きの Cisco スイッチや Foundry 及び IBM 製品など一部のスイッチの動作を真似ることです。

  このポリシーで使用されるアルゴリズムは、802.3ad に対応していません。

- **2** または **layer2+3** — layer2 及び layer3 プロトコル情報の組み合わせを使用して、ハッシュを生成します。

  ハードウェア MAC アドレス及び IP アドレスの XOR を使用して、ハッシュを生成します。公式は以下のとおりです:

  ```
  (((<source_IP> XOR <dest_IP>) AND 0xffff) XOR
    ( <source_MAC> XOR <destination_MAC> ))
      MODULO <slave_count>
  ```

  このアルゴリズムは、すべてのトラフィックを同じスレーブの特定のネットワークピアに割り振ります。非 IP トラフィックの場合、公式は layer2 送信ハッシュポリシーと同一です。

  このポリシーの目的は、特に layer3 ゲートウェイデバイスが大半の宛先に到達する必要がある環境において、layer2 単独の場合より分散されたトラフィックを提供することです。

  このアルゴリズムは、802.3ad に対応しています。

25.8. その他のリソース

カーネルモジュールとそれらのユーティリティの詳細については、以下のリソースを参照して下さい。

man ページ

man lsmod — **lsmod** コマンドの man ページです

- **man lsmod** — **lsmod** コマンドの man ページです。
- **man modinfo** — **modinfo** コマンドの man ページです。
- **man modprobe** — **modprobe** コマンドの man ページです。
- **man rmmod** — **rmmod** コマンドの man ページです。
- **man ethtool** — **ethtool** コマンドの man ページです。
- **man mii-tool** — **mii-tool** コマンドの man ページです。

インストールできる外部のドキュメント

- **/usr/share/doc/kernel-doc-<*kernel_version*>/Documentation/** — このディレクトリは kernel-doc パッケージにより提供されます。そこには、カーネル、カーネルモジュール及びそれぞれのパラメーターに関する情報が含まれています。カーネルのドキュメントにアクセスする前に、root で以下のコマンドを実行する必要があります:

```
~]# yum install kernel-doc
```

- Linux Loadable Kernel Module HOWTO — Linux Documentation Project の 『Linux Loadable Kernel Module HOWTO』 には、カーネルモジュールでの作業に関する詳しい情報が記載されています。

---

[6] 例が示唆している内容に関わらず、デフォルト設定で Energy Efficient Ethernet は **e1000e** ドライバーで有効です。

第26章 KDUMP クラッシュリカバリーサービス

kdump クラッシュダンピングメカニズムが有効だと、システムは別のカーネルのコンテキストからブートします。この第 2 のカーネルが確保するメモリーは少量で、システムがクラッシュした場合のコアダンプイメージのキャプチャーのみを目的としています。

コアダンプの分析ができると、システム障害の正確な原因究明に非常に役立つので、この機能を有効化が強く推奨されます。本章では、Red Hat Enterprise Linux における **kdump** サービスの設定/テスト/使用方法について説明します。また、生成されたコアダンプを **crash** デバッグユーティリティで分析する方法についても概説します。

26.1. KDUMP サービスのインストール

システム上で **kdump** サービスを使用するには、kexec-tools パッケージがインストールされていることを確認します。インストールするには、**root** として、以下をシェルプロンプトで入力します。

```
yum install kexec-tools
```

Red Hat Enterprise Linux における新規パッケージのインストール方法に関する詳しい説明は、「パッケージのインストール」 を参照して下さい。

26.2. KDUMP サービスの設定

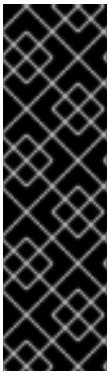
kdump サービスには以下の一般的な設定方法 3 つ があります。初回ブート時の設定、**カーネルダンプの設定** グラフィカルユーティリティを使用した設定、コマンドラインで手動での設定です。

重要

Intel IOMMU ドライバーの現行の実装における制限が原因で、**kdump** サービスによるコアダンプイメージのキャプチャーが時折妨げられることがあります。Intel アーキテクチャーで **kdump** を確実に使用するには、IOMMU サポートを無効にすることを推奨します。

26.2.1. 初回ブート時の kdump の設定

システムの初回ブート時には、**firstboot** アプリケーションが起動し、新たにインストールしたシステムの初期設定を通してユーザーをガイドします。**kdump** を設定するには、**Kdump** セクションに進み、以下の手順にしたがいます。



重要

システムに十分なメモリーがなければ、このオプションは使用できません。最小のメモリー要件については、『Red Hat Enterprise Linux Technology capabilities and limits』 comparison chart (バージョン比較) の 『Required minimums (最小メモリ容量)』 のセクションを参照して下さい。**kdump** クラッシュリカバリが有効化されている場合、最小のメモリー要件は、クラッシュリカバリに確保されているメモリー容量の分が増えます。この値はユーザーによって決定され、デフォルトでは 128 MB に物理メモリー 1 TB ごとに 64 MB を加えたものとなります (つまり、物理メモリーが 1 TB のシステムでは、合計 192 MB になります) 。

26.2.1.1. サービスの有効化

ブート時に **kdump** デーモンを起動可能にするには、**Enable kdump?** (kdump を有効にする) のチェックボックスにチェックマークを付けます。これにより、ランレベル **2**、**3**、**4**、**5** に対してサービスが有効となり、現行セッションで起動されます。同様に、このチェックボックスのチェックマークを外すと、サービスがすべてのランレベルに対して無効となり、直ちに停止されます。

26.2.1.2. メモリー使用量の設定

kdump カーネル用に確保するメモリー容量を設定するには、**Kdump メモリ** フィールドの横にある上向き/下向きの矢印をクリックして値を増減します。この値に応じて **使用可能なシステムメモリ** フィールドが変わり、システムが使用可能な残りのメモリ容量が表示される点に注意して下さい。

26.2.2. カーネルダンプ設定ユーティリティの使用

カーネルダンプの設定 ユーティリティを起動するには、パネルから **システム** → **管理** → **カーネルクラッシュダンプ** を選択するか、シェルプロンプトで **system-config-kdump** を入力します。すると、図26.1「基本設定」 に示したウィンドウが表示されます。

このユーティリティにより、**kdump** を設定したり、ブート時のサービス起動を有効化/無効化したりすることができます。設定が完了したら、**適用** をクリックして、変更を保存します。システムの再起動が要求され、認証済みでない場合には、スーパーユーザーのパスワードの入力を求めるプロンプトが表示されます。

重要

システムに十分なメモリーがなければ、このユーティリティは起動せず、エラーメッセージが表示されます。最小のメモリー要件については、『Red Hat Enterprise Linux Technology capabilities and limits』 comparison chart (バージョン比較) の 『Required minimums (最小メモリ容量)』 のセクションを参照して下さい。**kdump** クラッシュリカバリが有効化されている場合、最小のメモリー要件は、クラッシュリカバリに確保されているメモリー容量の分が増えます。この値はユーザーによって決定され、デフォルトでは 128 MB に物理メモリー 1 TB ごとに 64 MB を加えたものとなります (つまり、物理メモリーが 1 TB のシステムでは、合計 192 MB になります) 。

26.2.2.1. サービスの有効化

ブート時に **kdump** デーモンを起動するには、ツールバー上の **Enable (有効にする)** のボタンをクリックします。これにより、ランレベル **2**、**3**、**4**、**5** に対してサービスが有効となり、現行セッションで起動されます。同様に、**Disable (無効にする)** のボタンをクリックすると、サービスがすべてのランレベルに対して無効となり、直ちに停止されます。

ランレベルとサービス設定全般に関する詳細は 10章 *サービスとデーモン* を参照して下さい。

26.2.2.2. 基本設定タブ

基本設定 タブでは、**kdump** カーネル用に確保するメモリの量を設定することができます。この設定を行うには、**Manual kdump memory settings (手動の kdump メモリーセッティング)** のラジオボタンを選択し、**New kdump Memory (新規の kdump メモリー)** フィールドの横にある上向き/下向きの矢印ボタンをクリックして値を増減します。この値に応じて **Usable Memory (使用可能なメモリ)** フィールドが変わり、システムが使用可能な残りのメモリ容量が表示される点に注意して下さい。

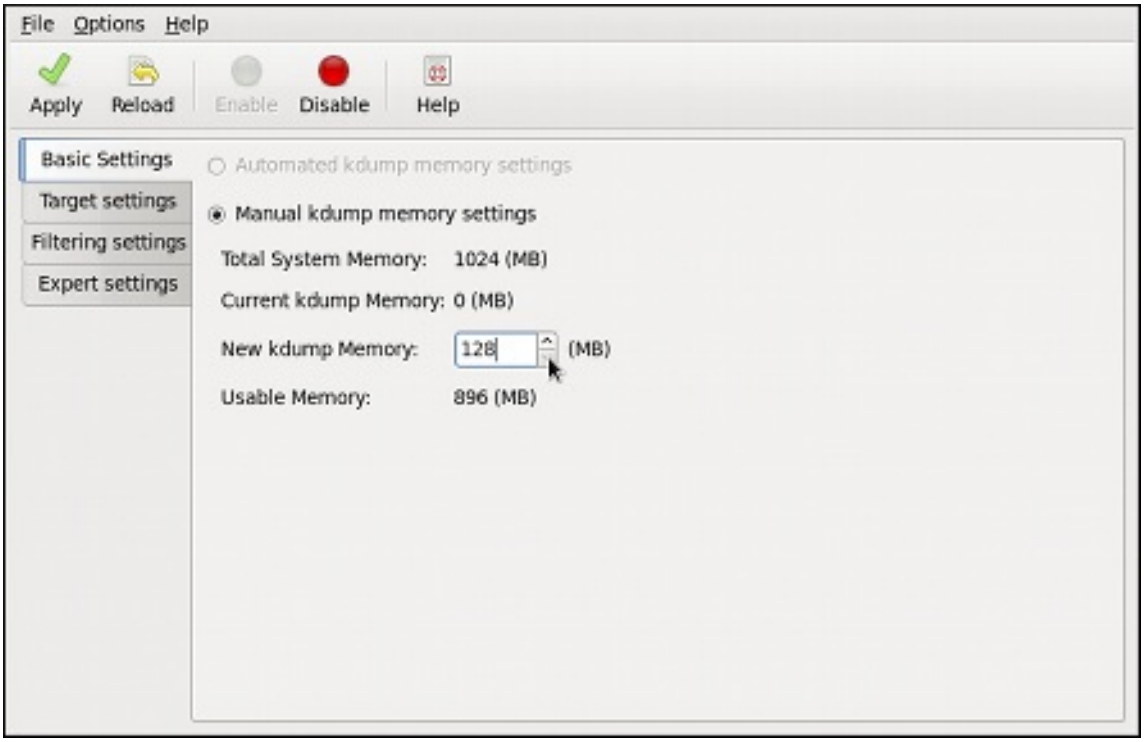


図26.1 基本設定

26.2.2.3. ターゲットセッティング タブ

Target Settings (ターゲットセッティング) タブでは、**vmcore** ダンプのターゲットロケーションを指定することができます。これは、ローカルシステムにファイルとして格納、デバイスに直接書き込み、NFS (Network File System) または SSH (Secure Shell) のプロトコルを使用したネットワーク上で送信のいずれかを選択することができます。

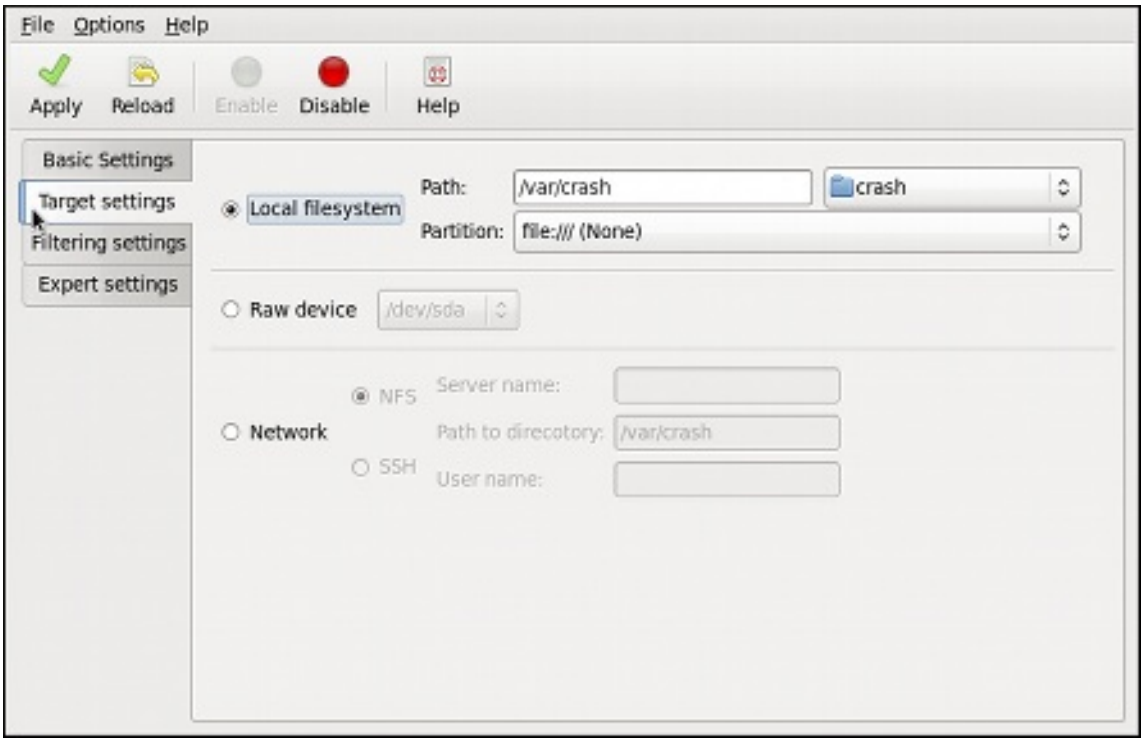


図26.2 ターゲットの設定

ダンプをローカルファイルシステムに保存するには、**Local filesystem (ローカルファイルシステム)** のラジオボタンを選択します。オプションとして、**Partition (パーティション)** プルダウンリ

ストから異なるパーティションを選択したり、**Path (パス)** プルダウンリストからターゲットディレクトリを選択して、設定をカスタマイズすることができます。

ダンプをデバイスに直接書き込むには、**Raw device (生デバイス)** のラジオボタンを選択し、横にあるプルダウンリストから希望のターゲットデバイスを選択します。

リモートマシン上にダンプを格納するには、**Network (ネットワーク)** のラジオボタンを選択します。NFS プロトコルを使用するには、**NFS** のラジオボタンを選択し、**Server name (サーバー名)** と **Path to directory (ディレクトリへのパス)** のフィールドに必要事項を記入します。SSH プロトコルを使用するには、**SSH** のラジオボタンを選択し、**サーバー名**、**ディレクトリへのパス**、及び **User name (ユーザー名)** のフィールドにリモートサーバーのアドレス、ターゲットディレクトリ、有効なリモートユーザー名をそれぞれ記入します。SSH サーバーの設定法とキーベースの認証の仕方についての詳細には、12章*OpenSSH* を参照して下さい。

現在サポートされているターゲットの完全な一覧は、表26.1「サポートされている kdump ターゲット」 を参照して下さい。

表26.1 サポートされている kdump ターゲット

タイプ	サポートされているターゲット	サポートされていないターゲット
生 (Raw) デバイス	ローカルで添付されたすべての生デバイスとパーティション	—
ローカルファイルシステム	直接接続されているディスクドライブ、ハードウェア RAID 論理ドライブ、LVM デバイス、および **mdraid** アレイ上にある **ext2**、**ext3**、**ext4**、**minix**、**btrfs**、**xfs** のファイルシステム	この表で明示的にサポート対象とされていないローカルファイルシステム。**auto** タイプ (自動ファイルシステム検出) を含む。
リモートディレクトリ	**IPv4** 上で **NFS** または **SSH** プロトコルを使用してアクセスされるリモートディレクトリ	**NFS** プロトコルを使用してアクセスされる **rootfs** ファイルシステム上のリモートディレクトリ
	iBFT (*iSCSI Boot Firmware Table*) が利用されていない場合に、ソフトウェアイニシエーター上で **iSCSI** プロトコルを使用してアクセスされるリモートディレクトリ	ソフトウェアイニシエーター上で **iBFT** を利用した **iSCSI** プロトコルを使用してアクセスされるリモートディレクトリ
	—	ハードウェアイニシエーター上で **iSCSI** プロトコルを使用してアクセスされるリモートディレクトリ
		IPv6 上でアクセスされるリモートディレクトリ
		SMB/**CIFS** プロトコルを使用してアクセスされるリモートディレクトリ

タイプ	サポートされているターゲット	サポートされていないターゲット
		FCoE (*Fibre Channel over Ethernet*) プロトコルを使用してアクセスされるリモートディレクトリ
		ワイヤレスネットワークインターフェイスを使用してアクセスされるリモートディレクトリ
		マルチパスベースのストレージ

26.2.2.4. フィルタリングセッティング タブ

Filtering Settings (フィルタリングセッティング) タブでは、**vmcore** ダンプのフィルターレベルを選択することができます。

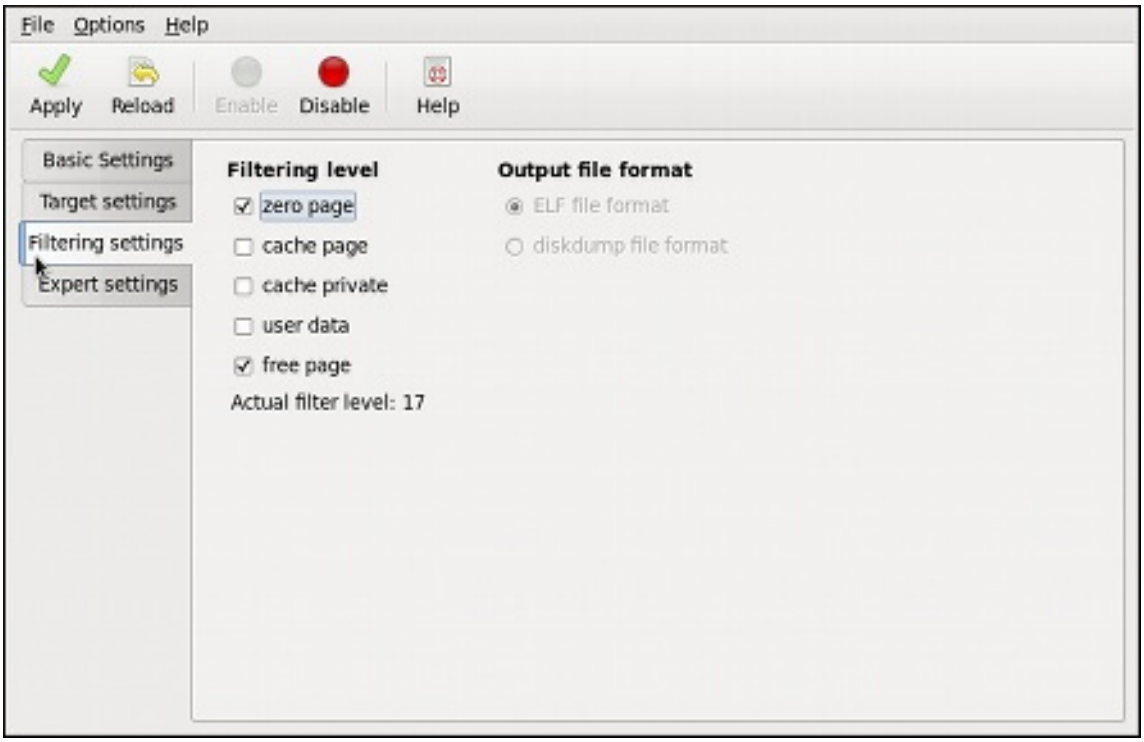


図26.3 フィルタリングセッティング

ダンプから **ゼロページ**、**キャッシュページ**、**キャッシュプライベート**、**ユーザーデータ**、または **フリーページ** を除外するには、該当するラベルの横にあるチェックボックスにチェックマークを付けます。

26.2.2.5. エキスパートセッティング タブ

Expert Settings (エキスパートセッティング) タブでは、使用するカーネルや初期 RAM ディスクを選択したり、カーネル及びコアコレクタープログラムに渡されるオプションをカスタマイズすることができます。

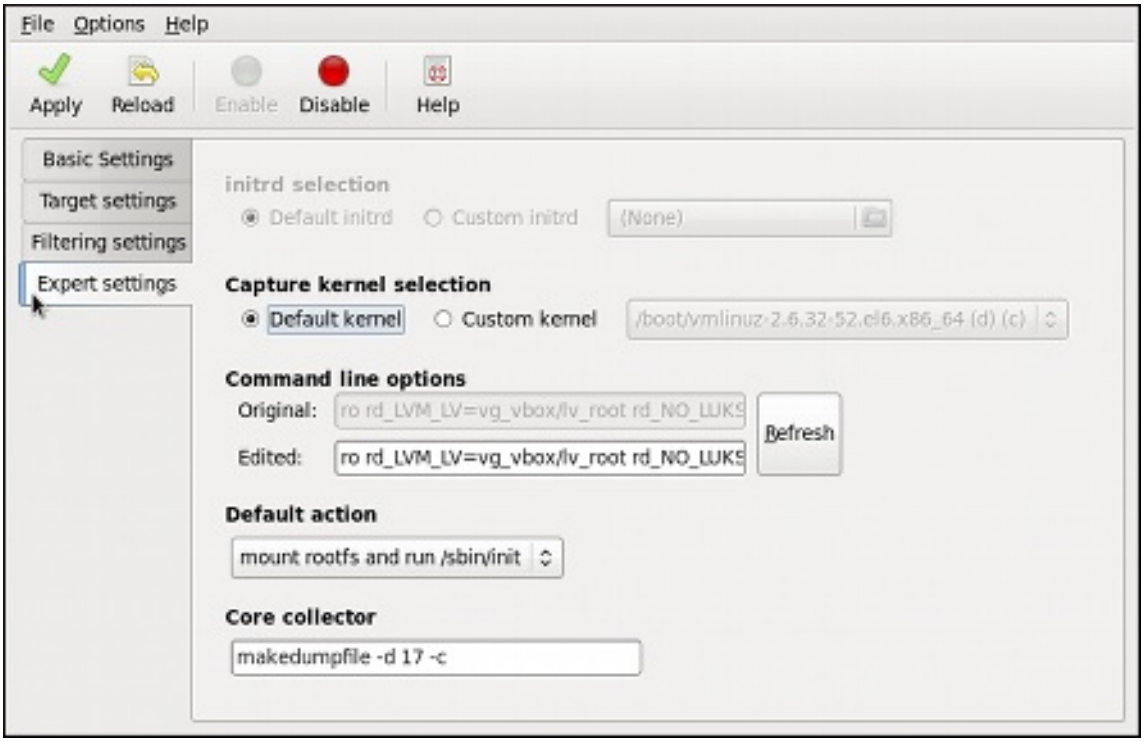


図26.4 エキスパートセッティング

異なる初期 RAM ディスクを使用するには、**Custom initrd (カスタム initrd)** のラジオボタンを選択して、その横にあるプルダウンリストから任意の RAM ディスクを選びます。

別のカーネルをキャプチャーするには、**Custom kernel (カスタムカーネル)** のラジオボタンを選択して、その右側にあるプルダウンリストから任意のカーネルイメージを選びます。

ブート時にカーネルに渡されるオプションのリストを調整するには、**Edited (編集済み)** のテキストフィールドの内容を編集します。**Refresh (更新)** ボタンをクリックすると、いつでも変更を元に戻すことができる点に注意して下さい。

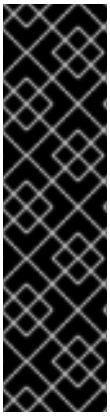
kdump がコアダンプの作成に失敗した際に実行する動作を選択するには、**デフォルトの動作** プルダウンリストから該当するオプションを選びます。利用可能なオプションには、**rootfs をマウントして /sbin/init を実行** (デフォルトの動作)、**reboot** (システムを再起動)、**shell** (ユーザーにインタラクティブなシェルプロンプトを提示)、**halt** (システムを停止)、および **poweroff** (システムの電源を切断) があります。

makedumpfile コアコレクターに渡されるオプションをカスタマイズするには、**Core collector (コアコレクタ)** のテキストフィールドを編集します。詳しくは、「コアコレクターの設定」 をご覧下さい。

26.2.3. コマンドライン上で kdump の設定

26.2.3.1. メモリー使用量の設定

kdump カーネル用に確保するメモリ容量を設定するには、**root** としてテキストエディターで **/boot/grub/grub.conf** ファイルを開き、例26.1「サンプルの /boot/grub/grub.conf ファイル」 に示したように、カーネルオプションのリストに **crashkernel=<*size*>M** (または、**crashkernel=auto**) パラメーターを追加します。



重要

システムに十分なメモリーがなければ、**kdump** クラッシュリカバリーサービスは動作しません。最小のメモリー要件については、『Red Hat Enterprise Linux Technology capabilities and limits』 comparison chart (バージョン比較) の 『Required minimums (最小メモリ容量)』 のセクションを参照して下さい。**kdump** クラッシュリカバリが有効化されている場合、最小のメモリー要件は、クラッシュリカバリに確保されているメモリー容量の分が増えます。この値はユーザーによって決定され、**crashkernel=auto** オプションが使用されていると、デフォルトでは 128 MB に物理メモリー 1 TB ごとに 64 MB を加えたものとなります (つまり、物理メモリーが 1 TB のシステムでは、合計 192 MB になります) 。

重要

Red Hat Enterprise Linux 6 では、**crashkernel=auto** オプションは、システムに 4 GB 以上の物理メモリーがある場合のみ、メモリーを確保します。

例26.1 サンプルの /boot/grub/grub.conf ファイル

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this
file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/sda3
#          initrd /initrd
#boot=/dev/sda
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Red Hat Enterprise Linux Server (2.6.32-220.el6.x86_64)
        root (hd0,0)
        kernel /vmlinuz-2.6.32-220.el6.x86_64 ro root=/dev/sda3
crashkernel=128M
        initrd /initramfs-2.6.32-220.el6.x86_64.img
```

26.2.3.2. ターゲットタイプの設定

カーネルクラッシュがキャプチャーされる場合、コアダンプは、ローカルファイルシステムにファイルとして格納されるか、デバイスに直接書き込まれるか、NFS (Network File System) または SSH (Secure Shell) のプロトコルを使用したネットワーク上で送信されるか、のいずれかの方法で処理が可能です。現時点で設定できるのは、これらのオプションのうちの一つのみである点に注意して下さい。デフォルトのオプションは、**vmcore** ファイルをローカルファイルシステムの **/var/crash/** ディレクトリ内に格納する方法です。このオプションを変更するには、**root** としてテキストエディターで **/etc/kdump.conf** 設定ファイルを開き、以下のようにオプションを編集します。

コアダンプを保存するローカルディレクトリを変更するには、**#path /var/crash** の行頭のハッシュサイン (「#」) を削除し、値を任意のディレクトリのパスに置き換えます。オプションとして、ファイルを異なるパーティションに書き込むには、**#ext4 /dev/sda3** の行についても、同じ手順に従って、

ファイルシステムタイプとデバイス (デバイス名、ファイルシステムラベル、UUID をすべてサポート) を適宜変更します。以下が例となります:

```
ext3 /dev/sda4
path /usr/local/cores
```

デバイスに直接ダンプディレクトリを書き込むには、**#raw /dev/sda5** の行頭のハッシュサイン (「#」) を削除して、値を任意のデバイス名に置き換えます。以下が例となります:

```
raw /dev/sdb1
```

NFS プロトコルを使用してリモートのマシンにダンプを保存するには、**#net my.server.com:/export/tmp** の行頭からハッシュサイン (「#」) を削除して、値を有効なホスト名とディレクトリパスに置き換えます。以下が例となります:

```
net penguin.example.com:/export/cores
```

SSH プロトコルを使用してリモートのマシンにダンプを格納するには、**#net user@my.server.com** の行頭のハッシュサイン (「#」) を削除し、値を有効なユーザー名とホスト名に置き換えます。以下が例となります:

```
net john@penguin.example.com
```

SSH サーバーの設定方法及びキーベース認証の設定方法については、12章*OpenSSH* を参照して下さい。

現在サポートされているターゲットの完全な一覧は、表26.1「サポートされている kdump ターゲット」 を参照して下さい。

26.2.3.3. コアコレクターの設定

kdump では、**vmcore** ダンプファイルのサイズを縮小する目的で、外部アプリケーション (コアコレクター) を指定してデータを圧縮することができます。又、オプションとして関連性のない情報をすべて除外することも可能です。現在、完全にサポートされている唯一のコアコレクターは **makedumpfile** です。

コアコレクターを有効にするには、**root** としてテキストエディターで **/etc/kdump.conf** 設定ファイルを開き、**#core_collector makedumpfile -c --message-level 1 -d 31** 行の先頭にあるハッシュサイン (「#」) を削除してから、以下のようにコマンドラインオプションを編集します。

ダンプファイルの圧縮を有効にするには、**-c** パラメーターを追加します。以下が例となります:

```
core_collector makedumpfile -c
```

ダンプから特定のページを削除するには、**-d *value*** パラメーターを追加します。ここで *value* は、表26.2「サポートされているフィルターレベル」 に記載したページの値の合計です。例えば、ゼロとフリーページの両方を削除するには、以下を使用します。

```
core_collector makedumpfile -d 17 -c
```

使用可能なオプションの完全な一覧は、**makedumpfile** の man ページを参照して下さい。

表26.2 サポートされているフィルターレベル

オプション	詳細
1	ゼロページ
2	キャッシュページ
4	キャッシュプライベート
8	ユーザーページ
16	フリーページ

26.2.3.4. デフォルトの動作の変更

デフォルトでは、**kdump** がコアダンプの作成に失敗すると、ルートのファイルシステムがマウントされ、**/sbin/init** が実行されます。この動作を変更するには、**root** としてテキストエディターで **/etc/kdump.conf** 設定ファイルを開き、**#default shell** の行頭のハッシュサイン (「#」) を削除してから、値を 表26.3「サポートされている動作」 に記載した任意の動作に置き換えます。

表26.3 サポートされている動作

オプション	詳細
reboot	システムを再起動します。その過程でコアは失われます。
halt	システムを停止します。
poweroff	システムの電源を切断します。
shell	initramfs 内で **msh** セッションを実行し、ユーザーがコアを手動で記録できるようにします。

例:

```
default halt
```

26.2.3.5. サービスの有効化

ブート時に **kdump** デーモンを起動するには、シェルプロンプトで **root** として以下を入力します。

```
chkconfig kdump on
```

これにより、ランレベル **2**、**3**、**4**、**5** に対してサービスが有効になります。同様に、**chkconfig kdump off** と入力すると、すべてのランレベルに対してサービスが無効となります。現行セッションでサービスを起動するには、**root** として以下のコマンドを使用します。

```
service kdump start
```

ランレベルとサービス設定全般に関する詳細は 10章 *サービスとデーモン* を参照して下さい。

26.2.4. 設定のテスト

警告

以下のコマンドはカーネルのクラッシュの原因となります。これらの手順に従う場合には、慎重に行い、実稼働用のマシンでは決して使用しないで下さい。

設定をテストするには、**kdump** を有効にした状態でシステムを再起動し、サービスが実行されていることを確認します (Red Hat Enterprise Linux におけるサービスの実行方法についての詳しい情報は、「サービスの実行」 を参照して下さい)。

```
~]# service kdump status
Kdump is operational
```

次に、シェルプロンプトで以下のコマンドを入力します。

```
echo 1 > /proc/sys/kernel/sysrq
echo c > /proc/sysrq-trigger
```

このコマンドにより、Linux カーネルは強制的にクラッシュして ***address-YYYY-MM-DD-HH:MM:SS*/vmcore** ファイルが設定で選択した場所にコピーされます (デフォルトでは **/var/crash/**)。

26.3. コアダンプの分析

システムクラッシュの原因を究明するには、GNU Debugger (GDB) と非常に良く似たインタラクティブなプロンプトを提供する **crash** ユーティリティを使用することができます。このユーティリティにより、実行中の Linux システムに加えて、**netdump**、**diskdump**、**xendump**、または **kdump** によって作成されたコアダンプをインタラクティブに分析することができます。

注記

vmcore ダンプファイルを分析するには、crash と kernel-debuginfo のパッケージがインストールされている必要があります。これらのパッケージをインストールするには、**root** としてシェルプロンプトで以下のように入力します。

```
yum install crash
debuginfo-install kernel
```

Red Hat Enterprise Linux における新規パッケージのインストール方法に関する詳しい説明は、「パッケージのインストール」 を参照して下さい。

26.3.1. crash ユーティリティの実行

ユーティリティを起動するには、シェルプロンプトで以下の形式のコマンドを入力します:

```
crash /var/crash/timestamp/vmcore
/usr/lib/debug/lib/modules/kernel/vmlinux
```

kernel のバージョンは、**kdump** によってキャプチャーされたものと同じでなければならない点に注意して下さい。現在実行中のカーネルを確認するには、**uname -r** のコマンドを使用します。

例26.2 crash ユーティリティの実行

```
~]# crash /usr/lib/debug/lib/modules/2.6.32-69.el6.i686/vmlinux \
/var/crash/127.0.0.1-2010-08-25-08:45:02/vmcore

crash 5.0.0-23.el6
Copyright (C) 2002-2010  Red Hat, Inc.
Copyright (C) 2004, 2005, 2006  IBM Corporation
Copyright (C) 1999-2006  Hewlett-Packard Co
Copyright (C) 2005, 2006  Fujitsu Limited
Copyright (C) 2006, 2007  VA Linux Systems Japan K.K.
Copyright (C) 2005  NEC Corporation
Copyright (C) 1999, 2002, 2007  Silicon Graphics, Inc.
Copyright (C) 1999, 2000, 2001, 2002  Mission Critical Linux, Inc.
This program is free software, covered by the GNU General Public
License,
and you are welcome to change it and/or distribute copies of it under
certain conditions.  Enter "help copying" to see the conditions.
This program has absolutely no warranty.  Enter "help warranty" for
details.

GNU gdb (GDB) 7.0
Copyright (C) 2009 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later
<http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show
copying"
and "show warranty" for details.
This GDB was configured as "i686-pc-linux-gnu"...

      KERNEL: /usr/lib/debug/lib/modules/2.6.32-69.el6.i686/vmlinux
    DUMPFILE: /var/crash/127.0.0.1-2010-08-25-08:45:02/vmcore  [PARTIAL
DUMP]
        CPUS: 4
        DATE: Wed Aug 25 08:44:47 2010
      UPTIME: 00:09:02
LOAD AVERAGE: 0.00, 0.01, 0.00
       TASKS: 140
    NODENAME: hp-dl320g5-02.lab.bos.redhat.com
     RELEASE: 2.6.32-69.el6.i686
     VERSION: #1 SMP Tue Aug 24 10:31:45 EDT 2010
     MACHINE: i686  (2394 Mhz)
      MEMORY: 8 GB
       PANIC: "Oops: 0002 [#1] SMP " (check log for details)
         PID: 5591
     COMMAND: "bash"
        TASK: f196d560  [THREAD_INFO: ef4da000]
         CPU: 2
```

```
        STATE: TASK_RUNNING (PANIC)

crash>
```

26.3.2. メッセージバッファーの表示

カーネルメッセージバッファーを表示するには、インタラクティブプロンプトで **log** コマンドを入力します。

例26.3 カーネルメッセージバッファーの表示

```
crash> log
... several lines omitted ...
EIP: 0060:[<c068124f>] EFLAGS: 00010096 CPU: 2
EIP is at sysrq_handle_crash+0xf/0x20
EAX: 00000063 EBX: 00000063 ECX: c09e1c8c EDX: 00000000
ESI: c0a09ca0 EDI: 00000286 EBP: 00000000 ESP: ef4dbf24
 DS: 007b ES: 007b FS: 00d8 GS: 00e0 SS: 0068
Process bash (pid: 5591, ti=ef4da000 task=f196d560 task.ti=ef4da000)
Stack:
 c068146b c0960891 c0968653 00000003 00000000 00000002 efade5c0 c06814d0
<0> fffffffb c068150f b7776000 f2600c40 c0569ec4 ef4dbf9c 00000002
b7776000
<0> efade5c0 00000002 b7776000 c0569e60 c051de50 ef4dbf9c f196d560
ef4dbfb4
Call Trace:
 [<c068146b>] ? __handle_sysrq+0xfb/0x160
 [<c06814d0>] ? write_sysrq_trigger+0x0/0x50
 [<c068150f>] ? write_sysrq_trigger+0x3f/0x50
 [<c0569ec4>] ? proc_reg_write+0x64/0xa0
 [<c0569e60>] ? proc_reg_write+0x0/0xa0
 [<c051de50>] ? vfs_write+0xa0/0x190
 [<c051e8d1>] ? sys_write+0x41/0x70
 [<c0409adc>] ? syscall_call+0x7/0xb
Code: a0 c0 01 0f b6 41 03 19 d2 f7 d2 83 e2 03 83 e0 cf c1 e2 04 09 d0
88 41 03 f3 c3 90 c7 05 c8 1b 9e c0 01 00 00 00 0f ae f8 89 f6 <c6> 05
00 00 00 00 01 c3 89 f6 8d bc 27 00 00 00 00 8d 50 d0 83
EIP: [<c068124f>] sysrq_handle_crash+0xf/0x20 SS:ESP 0068:ef4dbf24
CR2: 0000000000000000
```

このコマンドの使用方法についての詳しい情報を参照するには、**help log** と入力して下さい。

26.3.3. バックトレースの表示

カーネルスタックトレースを表示するには、インタラクティブプロンプトで **bt** コマンドを入力します。**bt *pid*** を使用すると、選択したプロセスのバックトレースを表示することができます。

例26.4 カーネルスタックトレースの表示

```
crash> bt
PID: 5591   TASK: f196d560  CPU: 2   COMMAND: "bash"
```

```
 #0 [ef4dbdcc] crash_kexec at c0494922
 #1 [ef4dbe20] oops_end at c080e402
 #2 [ef4dbe34] no_context at c043089d
 #3 [ef4dbe58] bad_area at c0430b26
 #4 [ef4dbe6c] do_page_fault at c080fb9b
 #5 [ef4dbee4] error_code (via page_fault) at c080d809
    EAX: 00000063  EBX: 00000063  ECX: c09e1c8c  EDX: 00000000  EBP:
00000000
    DS:  007b      ESI: c0a09ca0  ES:  007b      EDI: 00000286  GS:
00e0
    CS:  0060      EIP: c068124f  ERR: ffffffff  EFLAGS: 00010096
 #6 [ef4dbf18] sysrq_handle_crash at c068124f
 #7 [ef4dbf24] __handle_sysrq at c0681469
 #8 [ef4dbf48] write_sysrq_trigger at c068150a
 #9 [ef4dbf54] proc_reg_write at c0569ec2
#10 [ef4dbf74] vfs_write at c051de4e
#11 [ef4dbf94] sys_write at c051e8cc
#12 [ef4dbfb0] system_call at c0409ad5
    EAX: ffffffda  EBX: 00000001  ECX: b7776000  EDX: 00000002
    DS:  007b      ESI: 00000002  ES:  007b      EDI: b7776000
    SS:  007b      ESP: bfcb2088  EBP: bfcb20b4  GS:  0033
    CS:  0073      EIP: 00edc416  ERR: 00000004  EFLAGS: 00000246
```

このコマンドの使用方法についての詳しい情報を参照するには、**help bt** と入力して下さい。

26.3.4. プロセスステータスの表示

システム内のプロセスのステータスを表示するには、インタラクティブプロンプトで **ps** コマンドを入力します。**ps *pid*** を使用すると、選択したプロセスのステータスを表示することができます。

例26.5 システム内のプロセスのステータスの表示

```
crash> ps
   PID    PPID  CPU   TASK    ST  %MEM     VSZ    RSS  COMM
>     0      0   0  c09dc560  RU   0.0       0      0  [swapper]
>     0      0   1  f7072030  RU   0.0       0      0  [swapper]
      0      0   2  f70a3a90  RU   0.0       0      0  [swapper]
>     0      0   3  f70ac560  RU   0.0       0      0  [swapper]
      1      0   1  f705ba90  IN   0.0    2828   1424  init
... several lines omitted ...
   5566      1   1  f2592560  IN   0.0   12876    784  auditd
   5567      1   2  ef427560  IN   0.0   12876    784  auditd
   5587   5132   0  f196d030  IN   0.0   11064   3184  sshd
>  5591   5587   2  f196d560  RU   0.0    5084   1648  bash
```

このコマンドの使用方法についての詳しい情報を参照するには、**help ps** と入力して下さい。

26.3.5. 仮想メモリ情報の表示

基本的な仮想メモリ情報を表示するには、インタラクティブプロンプトで **vm** のコマンドを入力します。**vm *pid*** を使用すると、選択したプロセスに関する情報を表示することができます。

例26.6 現在のコンテキストの仮想メモリ情報の表示

```
crash> vm
PID: 5591   TASK: f196d560  CPU: 2   COMMAND: "bash"
   MM       PGD      RSS    TOTAL_VM
f19b5900  ef9c6000  1648k    5084k
  VMA       START      END    FLAGS  FILE
f1bb0310    242000    260000 8000875  /lib/ld-2.12.so
f26af0b8    260000    261000 8100871  /lib/ld-2.12.so
efbc275c    261000    262000 8100873  /lib/ld-2.12.so
efbc2a18    268000    3ed000 8000075  /lib/libc-2.12.so
efbc23d8    3ed000    3ee000 8000070  /lib/libc-2.12.so
efbc2888    3ee000    3f0000 8100071  /lib/libc-2.12.so
efbc2cd4    3f0000    3f1000 8100073  /lib/libc-2.12.so
efbc243c    3f1000    3f4000 100073
efbc28ec    3f6000    3f9000 8000075  /lib/libdl-2.12.so
efbc2568    3f9000    3fa000 8100071  /lib/libdl-2.12.so
efbc2f2c    3fa000    3fb000 8100073  /lib/libdl-2.12.so
f26af888    7e6000    7fc000 8000075  /lib/libtinfo.so.5.7
f26aff2c    7fc000    7ff000 8100073  /lib/libtinfo.so.5.7
efbc211c    d83000    d8f000 8000075  /lib/libnss_files-2.12.so
efbc2504    d8f000    d90000 8100071  /lib/libnss_files-2.12.so
efbc2950    d90000    d91000 8100073  /lib/libnss_files-2.12.so
f26afe00    edc000    edd000 4040075
f1bb0a18   8047000   8118000 8001875  /bin/bash
f1bb01e4   8118000   811d000 8101873  /bin/bash
f1bb0c70   811d000   8122000 100073
f26afae0   9fd9000   9ffa000 100073
... several lines omitted ...
```

このコマンドの使用方法についての詳しい情報を参照するには、**help vm** と入力してください。

26.3.6. オープンファイルの表示

オープンファイルに関する情報を表示するには、インタラクティブプロンプトで **files** のコマンドを入力します。**files *pid*** を使用すると、選択したプロセスによって開かれるファイルを表示することができます。

例26.7 現在のコンテキストのオープンファイルについての情報の表示

```
crash> files
PID: 5591   TASK: f196d560  CPU: 2   COMMAND: "bash"
ROOT: /    CWD: /root
 FD    FILE     DENTRY    INODE    TYPE  PATH
  0  f734f640  eedc2c6c  eecd6048  CHR   /pts/0
  1  efade5c0  eee14090  f00431d4  REG   /proc/sysrq-trigger
  2  f734f640  eedc2c6c  eecd6048  CHR   /pts/0
 10  f734f640  eedc2c6c  eecd6048  CHR   /pts/0
255  f734f640  eedc2c6c  eecd6048  CHR   /pts/0
```

このコマンドの使用方法に関する詳しい情報を参照するには、**help files** と入力して下さい。

26.3.7. ユーティリティの終了

インタラクティブプロンプトを終了して、**crash** を終了するには、**exit** 或いは **q** と入力します。

例26.8 crash ユーティリティの終了

```
crash> exit
~]#
```

26.4. その他のリソース

26.4.1. インストールされているドキュメント

- **kdump.conf**(5) — **/etc/kdump.conf** 設定ファイルの man ページです。使用可能なオプションについての全面的な説明が記載されています。
- **makedumpfile**(8) — **makedumpfile** コアコレクターの man ページです。
- **kexec**(8) — **kexec** の man ページです。
- **crash**(8) — **crash** ユーティリティの man ページです。
- **/usr/share/doc/kexec-tools-*version*/kexec-kdump-howto.txt** — **kdump** および **kexec** にインストールと使用方法に関する概要が記載されています。

26.4.2. 役立つ Web サイト

https://access.redhat.com/kb/docs/DOC-6039

kexec と **kdump** の設定に関する Red Hat のナレッジベース記事。

https://access.redhat.com/kb/docs/DOC-45183

サポートされている **kdump** ターゲットに関する Red Hat のナレッジベース記事。

http://people.redhat.com/anderson/

crash ユーティリティのホームページ。

付録A 一貫したネットワークデバイスの命名

Red Hat Enterprise Linux 6 は、ネットワークインターフェイス用に一貫したネットワークデバイス命名を提供します。この機能はインターフェイスの位置判定と区別が容易になるようにシステム上のネットワークインターフェイスの名前を制定します。

伝統的に Linux のネットワークインターフェイスは **eth[0123…]** などのように数値が付きました。しかし、これらの名前は必ずしもシャーシー上の実際のラベルと一致していません。複数のネットワークアダプターを持つ最近のサーバープラットフォームはこれらのインターフェイスについて非断定的な非直感的な命名に遭遇する可能性があります。この懸念はマザーボードに組み込まれたネットワークアダプター(*Lan-on-Motherboard*, 略語で *LOM*) と後付け (シングル及びマルチのポート) アダプターの両方に該当します。

新しい命名慣習では、組み込み型でも PCI スロットタイプでもその物理的位置に応じて名前をネットワークインターフェイスに割り当てます。この命名慣習に転向することにより、システム管理者はネットワークポートの物理的位置を想像したり、あるいは一貫した順序でそれらを命名し直すためにシステムを修正したりする必要は無くなります。

biosdevname プログラムで実装されているこの機能は、すべての組み込み型ネットワークインターフェイス、PCI カードネットワークインターフェイス、及び仮想機能ネットワークインターフェイスを、既存の **eth[0123…]** から 表A.1「新しい命名慣習」 に示してあるような新しい命名慣習へと変更するでしょう。

表A.1 新しい命名慣習

デバイス	旧式の名前	新しい名前
組み込み型ネットワークインターフェース (LOM)	**eth[0123…]**	**em[1234…]**[a]
PCI カードネットワークインターフェース	**eth[0123…]**	**p<*スロット*>p<*イーサネットポート*>**[b]
仮想機能	**eth[0123…]**	**p<*スロット*>p<*イーサネットポート*>_<*仮想インターフェイス*>**[c]

[a] 新しい数値は **1**から始まります。

[b] 例えば: **p3p4**

[c] 例えば: **p3p4_1**

システム管理者は **/etc/udev/rules.d/70-persistent-net.rules** にルールを書き込むことでデバイス名を希望の形式に変更することができます。この書き込みルールが物理位置命名慣習より優先されることになります。

A.1. 影響を受けるシステム

一貫したネットワークデバイス命名は、**Dell PowerEdge** のセット、**C Series**、及び **Precision Workstation** のシステム用にデフォルトで有効になっています。Dell システムに対するインパクトの詳細については https://access.redhat.com/kb/docs/DOC-47318 をご覧下さい。

他のシステムすべてに於いて、この命名方式はデフォルトで無効になっています。詳細については 「システムの要件」 及び 「この機能の有効化と無効化」 を参照して下さい。

システムのタイプに関係なく、Red Hat Enterprise Linux 5 ホストの下で実行している Red Hat Enterprise Linux 6 ゲストはデバイスの改名がありません。その理由は仮想マシンの BIOS が SMBIOS 情報を提供しないからです。Red Hat Enterprise Linux 6.0 から Red Hat Enterprise Linux 6.1 へのアップグレードは影響を受けませんので、旧式の **eth[0123…]** 命名慣習が継続して使用されます。

A.2. システムの要件

biosdevname プログラムはシステムの BIOS からの情報、特に SMBIOS 内に収納されている **type 9** (システムスロット) フィールドと **type 41** (オンボードデバイス拡張情報) フィールドからの情報を使用します。ステムの BIOS が SMBIOS のバージョン 2.6、またはそれ以降を持っていなければ、新しい命名慣習は使用されません。ほとんどの旧型のハードウェアは、必要な SMBIOS バージョンとフィールド情報を持つ BIOS を土台にしていないため、この機能をサポートしていません。BIOS あるいは SMBIOS バージョンの詳細についてはご使用のハードウェアの製造元にご連絡下さい。

この機能が効力を発揮するには、biosdevname パッケージがインストールされている必要があります。biosdevname パッケージは Red Hat Enterprise Linux 6 の **base** パッケージグループの一部です。**Minimal Install** 以外は、すべてのインストールオプションがこのパッケージを含んでいます。Red Hat Enterprise Linux 6.0 から RHEL 6.1.へのアップグレードにはインストールされていません。

A.3. この機能の有効化と無効化

デフォルトでこの機能を通常含んでいる Dell システム上で一貫したネットワークデバイス命名を無効にするには、インストール中とインストール後の両方で以下のオプションをブートコマンドラインに渡します:

```
biosdevname=0
```

最低限の要件 (「システムの要件」参照) を満たす他のタイプのシステム上でこの機能を有効にするには、インストール中とインストール後の両方で以下のオプションをブートコマンドラインに渡します:

```
biosdevname=1
```

システムが最低限要件を満たさない場合は、このオプションは無視されてシステムは伝統的なネットワークインターフェイス名形式でブートします。

biosdevname インストールオプションが指定されている場合、それはシステムの寿命期間中ブートオプションとして残る必要があります。

A.4. 管理者のための注記

多くのシステムカスタム化ファイルがネットワークインターフェイス名を含んでいる可能性がありますので、システムを旧式の慣習から新しい慣習へ移動する場合には、更新が必要となります。新しい命名慣習を使用する場合は、カスタム iptables ルール、スクリプト変更 irqbalance、及びその他の同様な設定ファイルのエリアでもネットワークインターフェイス名を更新する必要があります。また、インストール用にこの変更を有効にすると、**ksdevice** パラメータ経由でデバイス名を使用する既存のキックスタートファイルへの変更も必要になります。これらのキックスタートファイルはネットワークデバイスの MAC アドレス、またはネットワークデバイスの新しい名前を使用するために更新が必要になります。

Red Hat は、この機能についてインストール時点での選択肢として考慮されることをお薦めします。この機能の有効化/無効化をインストール後の作業とすることは、技術的には可能ですが、複雑化する可能

性があり推奨しかねます。希望されるシステム管理者は、最低限の要件を満足するシステム上で **/etc/udev/rules.d/70-persistent-net.rules** ファイルを削除して、すべての **/etc/sysconfig/network-scripts/ifcfg-*** ファイルの **HWADDR** の行を削除して下さい。そしてそれらの **ifcfg-***ファイルを改名してこの新しい命名慣習を使用するようにして下さい。新しい名前は再起動後に反映されます。ネットワークインターフェイスの名前を含む可能性のあるカスタムスクリプト、iptables ルール、及びサービス設定ファイルをすべて更新することを忘れないで下さい。

付録B RPM

RPM Package Manager (RPM パッケージマネージャ) (RPM) は、オープンパッケージングシステムで、Red Hat Enterprise Linux だけでなく他の Linux や UNIX システム上でも動作します。Red Hat, Inc. と Fedora Project では、他のベンダーに対し、自社製品に RPM を使用するよう奨励しています。RPM は *GPL* (*GNU General Public License*) 契約に基づいて配布されます。

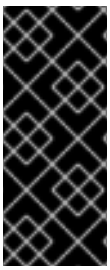
RPM パッケージマネージャは、**RPM 形式** と作動するようにビルトされたパッケージとともにのみ作動します。RPMは、プレインストールされた rpm パッケージとして提供されています。エンドユーザーは、RPM によりシステムを簡単に更新することができます。RPM パッケージのインストール、アンインストール、アップグレードは短いコマンドで実行できます。RPM はインストール済みのパッケージやそのファイル群に関するデータベースを維持しているため、システムで強力なクエリや検証を実行することができます。

RPM のパッケージ形式は、Red Hat Enterprise Linux 6 では改善されています。RPM パッケージは、XZ 無損失データ圧縮形式を使用して圧縮されます。これにはより強い圧縮と、圧縮解除時の CPU 使用低減という利点があり、パッケージの署名や検証において SHA-256 などの強力な複数ハッシュアルゴニズムをサポートします。

警告

大部分のパッケージ管理のタスクでは、**Yum** パッケージマネージャは、RPM と同じくらいかそれよりも卓越した機能とユーティリティを提供します。 **Yum** は、複雑なシステムの依存関係解決の実行と追跡をします。また、パッケージのインストールと削除に RPM も使用している場合はそれを指摘してシステムの整合性確認を強要します。これらの理由により、パッケージ管理のタスクを実行するにあたっては、可能な限り RPM の代わりに、**Yum** を使用することを強くお勧めします。6章*Yum* を参照してください。

グラフィカルインターフェースを選択したい場合は、**PackageKit** GUI アプリケーションを使用できます。これは、**Yum** をそのバックエンドとして使用するものであり、システムのパッケージを管理します。詳しくは 7章*PackageKit* を参照してください。



重要

パッケージのインストール時には、オペレーティングシステムとプロセッサのアーキテクチャに互換性があることを確かめてください。通常パッケージ名を確認すると分かります。次に示す例の多くは、AMD64/Intel 64 コンピュータのアーキテクチャ用にコンパイルされた RPM パッケージを示しています。そのため、RPM のファイル名は、`x86_64.rpm` で終わります。

アップグレード時に、RPM は設定ファイルを慎重に処理するためカスタム設定が失われることはありません。通常の `.tar.gz` ファイルでは、実現することができない特性です。

開発者はRPM を使用することで、ソフトウェアのソースコードを取り出し、エンドユーザー用にソースパッケージとバイナリパッケージにパッケージ化することができます。このプロセスは極めて単純で、1 つのファイルとユーザーが作成するオプションのパッチを基に実行されます。ビルドの指示に加えて、*純粋な* ソースとパッチの明確な説明により、ソフトウェアの新バージョンリリース時にパッケージの維持が容易になります。

注記

RPM はシステムに変更を加えるため、RPM パッケージのインストール、削除、アップグレードの実行時には root としてログインする必要があります。

B.1. RPM の設計目標

RPM の使用方法を理解するためには、RPM の設計目標を理解するとよいでしょう。

アップグレードの可能性

RPM を使用すれば、コンポーネントを完全に再インストールすることなく、個別にコンポーネントをアップグレードすることができます。Red Hat Enterprise Linux などの RPM に基づくオペレーティングシステムの新しいリリースを入手したときには、マシンにオペレーティングシステム自体を再インストールする必要がありません（他のパッケージングシステムに基づくオペレーティングシステムの場合はその必要があります）。RPM を使用すれば、インテリジェントで、完全自動化された適切なシステムのアップグレードを行うことができます。その上、パッケージに含まれる設定ファイルはアップグレード後にも保持されるので、カスタム設定が失われることはありません。システム上でのパッケージのインストールとアップグレードでは同じ RPM ファイルが使用されるため、パッケージをアップグレードするために特別なアップグレードファイルが必要になることはありません。

強力なクエリ

RPM では、強力なクエリのオプションが提供されています。データベース全体を通じてパッケージを検索したり、特定のファイル群のみを検索したりすることができます。あるファイルがどのパッケージに属し、どこから来たのかをも簡単に検索できます。RPM パッケージに含まれるファイルは圧縮アーカイブ形式であり、パッケージとその内容に関する有用な情報を含むカスタムバイナリヘッダが付いているため、個別のパッケージをすばやく簡単にクエリすることができます。

システムの検証

RPM のもう1つの強力な特徴は、パッケージの検証が可能であるということです。パッケージに関する重要なファイルを削除してしまったのではないかと心配になった場合には、パッケージを検証することができます。何か異常があれば通知されますので、その時点で必要に応じてパッケージを再インストールできます。再インストールを行っても、修正した設定ファイルは保持されます。

純粋なソース

極めて重要な設計目標は、ソフトウェアのオリジナル作者によって配布されたときのままの **純粋な** ソフトウェアソースをユーザーが利用できるようにすることでした。RPM を使用した場合、純粋なソースと適用済みのパッチ、完全なビルド指示が入手できます。様々な理由から、これは大きなメリットとなります。例えば、あるプログラムの新しいバージョンがリリースされた場合、それをコンパイルするために必ずしも最初から作業を始める必要はなくなります。パッチを見て、必要になる **かもしれない** 作業を確かめることができます。この技術を使えば、コンパイル済みのすべてのデフォルト設定と、ソフトウェアを適切に構築するために行われたすべての変更内容を容易に目に見える形にすることができます。

ソースを純粋な状態に保持するという目標は、開発者にとってのみ重要なことのように思われるかもしれませんが、結果として、エンドユーザーにより高品質のソフトウェアが提供されることにもなります。

B.2. RPM の使用法

RPMには、次の 5つの基本的作動モードがあります （パッケージの構築はカウントしません）: インストール、アンインストール、アップグレード、クエリ、検証の 5つです。このセクションでは、各モー

ドについて概説します。詳細な説明やオプションについては **rpm --help** または **man rpm** をご覧ください。RPM の詳細については 「その他のリソース」 も参照してください。

B.2.1. RPM パッケージの検索

RPM パッケージを使用する前に、それがどこにあるかを調べる必要があります。インターネットを検索すると、数多くの RPM レポジトリが見つかりますが、Red Hat 製の RPM パッケージは、以下の場所にあります:

- Red Hat Enterprise Linux インストールメディアには、多くのインストール可能な RPM があります。
- YUM パッケージマネージャで提供されている初期 RPM リポジトリ。公式の Red Hat Enterprise Linux パッケージリポジトリの使用法の詳細については 6章*Yum* を参照して下さい。
- Enterprise Linux 用エキストラパッケージは、Red Hat Enterprise Linux 用に高品質のアドオンパッケージを提供するためのコミュニティ有志の尽力によるものです。EPEL RPM パッケージの詳細については http://fedoraproject.org/wiki/EPEL を参照して下さい。
- Red Hat に関連しない非公式のサードパーティのリポジトリも RPM パッケージを提供します。

重要

Red Hat Enterprise Linux システムでの使用にサードパーティのリポジトリを考えている場合は、パッケージソースとしてリポジトリを追加する前に、パッケージの互換性に関するリポジトリのウェブサイトに注意を払ってください。代替のパッケージリポジトリは、Red Hat Enterprise Linux リポジトリにすでに含まれているパッケージなどと同じソフトウェアでも互換性のない異なるバージョンを提供することがあります。

- Red Hat のエラータページについては、http://www.redhat.com/apps/support/errata/ をご覧ください。

B.2.2. インストールとアップグレード

RPM パッケージは通常、**tree-1.5.3-2.el6.x86_64.rpm** のようなファイル名が付けられています。ファイル名は、パッケージ名（**tree**）、バージョン（**1.5.3**）、リリース（**2**）、オペレーティングシステムのメジャーバージョン（**el6**）、および CPU アーキテクチャ（**x86_64**）で構成されています。

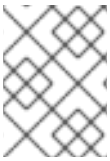
以下を実行するために、**rpm** の **-U** オプションを使用します:

- すでにシステム上にあるが古くなったパッケージを新しいバージョンにアップグレードする、または
- 古くなったバージョンがインストールされていない場合は、パッケージをインストールする

つまり、**rpm -U *<rpm_file>*** は、パッケージの必要に応じて **アップグレード** または **インストール** のいずれかの機能を実行できます。

tree-1.5.3-2.el6.x86_64.rpm パッケージが正しいディレクトリにあると仮定します。root としてログインし、**rpm** による tree パッケージをアップグレードまたはインストールするためにシェルプロンプトで以下のコマンドを入力します:

```
rpm -Uvh tree-1.5.3-2.el6.x86_64.rpm
```



注記

-v および **-h** のオプション（**-U** と組み合わされる）により、**rpm** は、詳細出力を表示し、ハッシュマークを使って進行メーターを表示します。

アップグレードおよび/またはインストールが正常に終了すると、以下のような出力が表示されます。

```
Preparing...                ########################################### 
[100%]
   1:tree                   ########################################### 
[100%]
```

警告

rpm は、パッケージをインストールするときに 2 つのオプションを提供します：前述の **-U** オプション（慣習的に **アップグレード** を意味します）と **インストール** を意味する **-i** オプションです。**-U** オプションには、インストールおよびアップグレード機能の両方が含まれるため、**kernel パッケージ以外の**すべてのパッケージに **rpm -Uvh** を使用することをお勧めします。

新しいカーネルパッケージはアップグレードするのではなく、単に **インストール** するために常に、**-i** オプションを使うことが推奨されます。カーネルパッケージをアップグレードする **-U** オプションを使用すると、前の（古い）カーネルパッケージが削除されるため、新しいカーネルに問題がある場合は、システムがブートできない状態になる可能性があります。そのため、新しいカーネルをインストールするには、**古い kernel パッケージを置換しない、 rpm -i *<kernel_package>*** コマンドを使用します。kernel パッケージのインストールについては、*24章 手動のカーネルアップグレード* を参照してください。

パッケージの署名は、パッケージのインストールまたはアップグレード時に自動的に確認されます。署名により、パッケージが認証機関によって署名されたことを確認できます。例えば、署名の検証が失敗すると、次のようなエラーメッセージが表示されます:

```
error: tree-1.5.3-2.el6.x86_64.rpm: Header V3 RSA/SHA256 signature: BAD,
key ID
d22e77f2
```

ヘッダのみの新しい署名なら、次のようなエラーメッセージが表示されます:

```
error: tree-1.5.3-2.el6.x86_64.rpm: Header V3 RSA/SHA256 signature: BAD,
key ID d22e77f2
```

署名を検証するための適切なキーをインストールしていない場合は、メッセージに **NOKEY** という単語が含まれます:

```
warning: tree-1.5.3-2.el6.x86_64.rpm: Header V3 RSA/SHA1 signature: NOKEY,
key ID 57bbccba
```

パッケージ署名の確認に関しては、「パッケージの署名確認」 を参照してください。

B.2.2.1. パッケージがすでにインストールされている場合

同一名で同一場合のパッケージがすでにインストールされていると、以下のような出力が表示されます。

```
Preparing...                ########################################### 
[100%]
   package tree-1.5.3-2.el6.x86_64 is already installed
```

どうしてもパッケージをインストールする必要がある場合は、**--replacepkgs** オプションを使用します。このオプションは RPM にエラーを無視するよう指示するものです:

```
rpm -Uvh --replacepkgs tree-1.5.3-2.el6.x86_64.rpm
```

このオプションは、RPM からインストールされたファイルが削除された場合や、RPM からオリジナルの設定ファイルをインストールしたい場合に便利です。

B.2.2.2. ファイルの競合

別のパッケージによってすでにインストールされたファイルを含むパッケージをインストールしようとすると、以下のメッセージが表示されます。

```
Preparing... ##################################################
 file /usr/bin/foobar from install of foo-1.0-1.el6.x86_64 conflicts
with file from package bar-3.1.1.el6.x86_64
```

--replacefiles オプションを使用して、RPM にこのエラーを無視するよう指示します。

```
rpm -Uvh --replacefiles foo-1.0-1.el6.x86_64.rpm
```

B.2.2.3. 未解決の依存

RPM パッケージは他のパッケージに依存することがあります。 つまり、正しく動作するために他のパッケージのインストールが必要なことがあります。未解決の依存関係を持つパッケージをインストールしようとすると、以下のような出力が表示されます。

```
error: Failed dependencies:
   bar.so.3()(64bit) is needed by foo-1.0-1.el6.x86_64
```

CD-ROM または DVD などの Red Hat Enterprise Linux インストールメディアからパッケージをインストールしている場合は、依存関係が利用可能なことがあります。Red Hat Enterprise Linux インストールメディアまたはアクティブな Red Hat Enterprise Linux ミラーで提案したパッケージを見つけて、次のようにコマンドに追加します。

```
rpm -Uvh foo-1.0-1.el6.x86_64.rpm    bar-3.1.1.el6.x86_64.rpm
```

両方のパッケージのインストールが正常に行われると、以下のような出力が表示されます。

```
Preparing...                ########################################### [
100%]
   1:foo                    ########################################### [
50%]
   2:bar                    ###########################################
[100%]
```

--whatprovides オプションを使うと、どのパッケージが必要なファイルを含んでいるか判断できます。

```
rpm -q --whatprovides "bar.so.3"
```

bar.so.3 を含むパッケージが RPM データベースにある場合は、そのパッケージ名が表示されます。

```
bar-3.1.1.el6.i586.rpm
```

警告

Failed dependencies エラーを出すパッケージを **rpm** は（**--nodeps** オプションを使用して）**強制的に** インストールできますが、これはお勧め **しません**。それを行うと、インストールしたパッケージの起動は通常失敗します。**rpm --nodeps** を使ってパッケージをインストールまたは削除することによって、アプリケーションが誤作動および/またはクラッシュすることにつながり、深刻なパッケージ管理の問題、場合によってはシステム障害を引き起こすことになります。そのため、警告に従うのが一番です。依存関係がどこにあるのかを明らかにすることは重要なため、パッケージマネージャは — **RPM**、**yum**、 または **PackageKit** であろうと — こうした警告を表示し、可能な修正方法を提案してくれます。**Yum** パッケージマネージャは、依存関係の解決を実行し、オンラインのリポジトリから依存関係をフェッチすることができますので、依存関係解決を考慮しないまま **rpm** で強制的に実行するよりも安全で、簡単でスマートな方法です。

B.2.3. 設定ファイルの変更

RPM は設定ファイルを使用してパッケージのインテリジェントなアップグレードを実行するため、以下のいずれかのメッセージが表示される場合があります。

```
saving /etc/foo.conf as /etc/foo.conf.rpmsave
```

このメッセージは、設定ファイルに加えた変更内容が、パッケージ内の新しい設定ファイルと **上位互換性** を持たない可能性があるため、RPM が元のファイルを保存してから新しいファイルをインストールしたことを意味しています。できる限り早期にこの 2つのファイルの違いを調査、解決することで、引き続きシステムが正しく動作することを確認する必要があります。

別の方法として、RPM はパッケージの **新しい** 設定ファイルを、例えば **foo.conf.rpmnew** として保存し、修正した設定ファイルをそのままにしておくこともできます。なお、通常古い設定ファイルから新しい設定ファイルへの変更と **diff** プログラムと組み合わせることによって、修正した設定ファイルと新しい設定ファイルとの競合を解決する必要があります。

古い バージョン番号が付いたパッケージでアップグレードしている場合（即ち、パッケージのより新しいバージョンが既にインストールされている場合）は、以下のような出力が表示されます。

```
package foo-2.0-1.el6.x86_64.rpm (which is newer than foo-1.0-1) is
already installed
```

それでも強制的に RPM にアップグレードさせるには、**--oldpackage** オプションを使用します。

```
rpm -Uvh --oldpackage foo-1.0-1.el6.x86_64.rpm
```

B.2.4. アンインストール

パッケージのアンインストールは、インストールと同様、簡単に実行できます。シェルプロンプトで以下の コマンドを入力します。

```
rpm -e foo
```

注記

アンインストール時に指定するのは、オリジナルのパッケージの **ファイル** の名前、**foo-1.0-1.el6.x86_64** ではなく、パッケージの **名前 foo** を使用することに注意してください。**rpm -e** コマンドとオリジナルのファイル名全体を使用してパッケージをアンインストールすると、パッケージ名のエラーが表示されます。

削除しようとしているパッケージに別のインストール済みのパッケージが依存している場合、アンインストール時に依存関係のエラーが発生することがあります。例えば、次のように表示されます。

```
rpm -e ghostscript
error: Failed dependencies:
        libgs.so.8()(64bit) is needed by (installed) libspectre-0.2.2-
3.el6.x86_64
        libgs.so.8()(64bit) is needed by (installed) foomatic-4.0.3-
1.el6.x86_64
        libijs-0.35.so()(64bit) is needed by (installed) gutenprint-5.2.4-
5.el6.x86_64
        ghostscript is needed by (installed) printer-filters-1.1-
4.el6.noarch
```

「未解決の依存」 の共有のオブジェクトライブラリ（***<library_name>*.so.*<number>*** ファイル）を検索する方法と類似の方法で、この厳密な構文を使って 64 ビットの共有オブジェクトライブラリを検索できます（ファイル名を引用することを忘れないでください）。

```
~]# rpm -q --whatprovides "libgs.so.8()(64bit)"
ghostscript-8.70-1.el6.x86_64
```

警告

Failed dependencies エラーを出すパッケージを **rpm** は（**--nodeps** オプションを使用して）**強制的に** インストールできますが、これはお勧め **しません**。それを行うと、インストールしたパッケージの起動は通常失敗します。**rpm --nodeps** を使ってパッケージをインストールまたは削除することによって、アプリケーションが誤作動および/またはクラッシュすることにつながり、深刻なパッケージ管理の問題、場合によってはシステム障害を引き起こすことになります。そのため、警告に従うのが一番です。依存関係がどこにあるのかを明らかにすることは重要なため、パッケージマネージャは — **RPM**、**yum**、 または **PackageKit** であろうと — こうした警告を表示し、可能な修正方法を提案してくれます。**Yum** パッケージマネージャは、依存関係の解決を実行し、オンラインのリポジトリから依存関係をフェッチすることができますので、依存関係解決を考慮しないまま **rpm** で強制的に実行するよりも安全で、簡単でスマートな方法です。

B.2.5. 最新の状態にする

パッケージを最新の状態にする作業は、既存パッケージだけがアップグレードされていることを除いては、アップグレードと似ています。シェルプロンプトで以下のコマンドを入力します。

```
rpm -Fvh foo-2.0-1.el6.x86_64.rpm
```

RPM の freshen オプションにより、コマンドラインで指定されたパッケージのバージョンと、すでにシステムにインストールされているパッケージのバージョンが照合されます。インストール済みのパッケージよりも新しいバージョンのパッケージに対して RPM の freshen オプションが実行されると、そのパッケージは新しいバージョンへアップグレードされます。ただし、同じ名前のインストール済みパッケージが存在しない場合は、RPM の freshen オプションによるパッケージのインストールは実行されません。これが、RPM のアップグレードのオプションと異なる点です。アップグレードでは、古いバージョンのパッケージがインストール済みであるかどうかに関係なくパッケージは **必ず** インストールされます。

freshen オプションは、単一のパッケージまたはパッケージグループに対しても使用できます。多数の異なるパッケージをダウンロードした後で、システムにインストール済みのパッケージをただアップグレードしたい場合に、freshen が役に立ちます。そのため、RPM を使う前にダウンロードしたグループ中から不要なパッケージを削除する必要はありません。

この場合、以下のコマンドを ***.rpm** グロブ (glob) を使って発行します:

```
rpm -Fvh *.rpm
```

RPM はすでにインストールされているパッケージのみ自動的にアップグレードします。

B.2.6. クエリ

RPM のデーターベースにはシステムにインストール済みのすべての RPM パッケージに関する情報が保存されています。それらは、**/var/lib/rpm/** ディレクトリ内にあり、どのパッケージがインストール済みで、どのパッケージがどのバージョンなのかをクエリしたり、またインストール以降に他で使用しているケースでパッケージ内のファイルに加えられたすべての変更を算出するときに使用します。

このデーターベースにクエリを行うには、**-q** オプションを使用します。**rpm -q *package name*** コマ

ンドは、パッケージ名、バージョン、インストール済みパッケージのリリース番号 *<package_name>* を表示します。例えば、**rpm -q tree** を使ってインストール済みのパッケージ **tree** にクエリを行うと、次のような出力が表示されることがあります。

```
tree-1.5.2.2-4.el6.x86_64
```

また、以下の **パッケージセレクションのオプション（Package Selection Options）** （これは RPM man ページの小見出しです。詳しくは **man rpm** をご覧ください）を使用して、さらに検索条件を絞ったり、クエリを修正したりすることもできます:

- **-a** — すべての現在インストール済みのパッケージのクエリを行う。
- **-f *<file_name>*** — パッケージが所有する RPM データーベースにクエリを行う ***<file_name>*** ファイルの絶対パスを指定する（例えば、**rpm -qf ls** の代わりに **rpm -qf /bin/ls** ）。
- **-p *<package_file>*** — アンインストール済みのパッケージ ***<package_file>*** のクエリを行う。

クエリを行なったパッケージに関してどの情報を表示するか指定する方法は多数あります。検索を行う情報の種類を選択するには、以下のオプションを使用します。これらのオプションを **パッケージクエリのオプション（Package Query Options）** と呼びます。

- **-i** で、 パッケージ名、詳細、リリース、サイズ、ビルドした日、インストールした日、ベンダーなど多様なパッケージ情報を表示します。
- **-l** で、パッケージに含まれるファイルの一覧を表示します。
- **-s** で、パッケージに含まれるすべてのファイルの状態を表示します。
- **-d** で、パッケージ内のドキュメンテーション（man ページ、info ページ、README など）とマークが付けられたファイルの一覧を表示します。
- **-c** で、設定ファイルとしてマークが付けられたファイルの一覧が表示されます。これはパッケージをシステムに適合させ、カスタマイズするために、ユーザーがインストール後に編集するファイルです（例えば、**sendmail.cf**、**passwd**、**inittab** など）。

ファイルの一覧を表示するオプションについては、コマンドに **-v** を追加して、その一覧を使い慣れた **ls -l** 形式で表示します。

B.2.7. 検証

パッケージの検証では、パッケージからインストールされたファイルに関する情報と、元のパッケージに含まれるファイルに関する情報が同一かどうか調べます。検証により、ファイルのサイズ、MD5 サム、権限、タイプ、所有者、ファイルのグループをはじめ、様々なことが比較されます。

コマンド **rpm -V** を実行すると、パッケージの検証が行われます。クエリの説明で示した **オプションの検証**を使用して、検証したいパッケージを指定することができます。簡単な使用方法としては、**rpm -V tree** というものがあります。これを実行すると、**tree** パッケージに含まれるすべてのファイルがそれがインストールされたときの状態と同じであるかどうかが検証されます。例えば次のようにします:

- 特定のファイルを含むパッケージを検証するには、以下のようにします:

```
rpm -Vf /usr/bin/tree
```

この例では、**/usr/bin/tree** がパッケージのクエリを行うために使用されるファイルの絶対パスです。

- システムのインストール済みのすべてのパッケージを検証するには以下のようにします（時間がかかります）:

```
rpm -Va
```

- インストール済みのパッケージと、RPM パッケージファイルとを検証するには以下のようにします:

```
rpm -Vp tree-1.5.3-2.el6.x86_64.rpm
```

RPM データベースが破損した疑いがある場合には、このコマンドが役立ちます。

すべてが正常に検証された場合は何も出力されません。何らかの矛盾が見つかった場合はその内容が表示されます。出力フォーマットは、8 個の文字列（ "**c**" は設定ファイルを示す）とファイル名です。8 個の各文字は、ファイルの 1 つの属性と RPM データベースに記録されたその属性の値とを比較した結果を示します。シングルピリオド（**.**）は、テストに合格したことを示します。以下の文字は、それぞれ何らかの矛盾が見つかったことを示しています:

- **5** — MD5 チェックサム
- **S** — ファイルサイズ
- **L** — シンボリックリンク
- **T** — ファイル修正時刻
- **D** — デバイス
- **U** — ユーザー
- **G** — グループ
- **M** — モード（権限とファイルタイプを含む）
- **?** — 読み込み不可ファイル（例：ファイルのアクセス権のエラー）

何かが出力された場合は、パッケージを削除するか、再インストールするか、または別の方法で問題を修正するかを熟慮の上判断してください。

B.3. パッケージの署名確認

パッケージが破損していないことや、改ざんされていないことを確認する場合には、シェルプロンプトで以下のコマンドを入力して md5sum のみを検証します (*<rpm_file>* は RPM パッケージのファイル名):

```
rpm -K --nosignature <rpm_file>
```

<*rpm_file*>: rsa sha1 (md5) pgp md5 OK という出力 (特に **OK** の部分) が表示されます。この短いメッセージは、ダウンロード中にファイルが破損していないことを示しています。より詳細なメッセージを表示するには、コマンドで **-K** を **-Kvv** に置き換えて下さい。

又、パッケージを作成した開発者についてはどの程度信頼できるでしょうか。パッケージが、開発者の GnuPG *キー* を使用して *署名されている* 場合には、その開発者が名乗った通りの人物であることがわかります。

RPM パッケージは、*GNU Privacy Guard* (GnuPG) を使用して署名することができます。これは、ダウンロードしたパッケージの信頼性の確認に役立ちます。

GnuPG は、セキュアな通信ツールで、電子プライバシープログラム PGP の暗号化技術に代わる完全かつ無償のツールです。GnuPG を使用して、ドキュメントの有効性の認証を行ったり、他の受信者とやり取りするデータを暗号化/復号化することができます。GnuPG は PGP5.*x* ファイルの復号化と検証が可能です。

GnuPG はインストール時にデフォルトでインストールされるので、即時に使用を開始して、Red Hat から受信するパッケージを検証することができます。この作業を行う前には、まず Red Hat の公開鍵をインポートする必要があります。

B.3.1. キーのインポート

Red Hat のパッケージを検証するには、Red Hat の GnuPG キーをインポートする必要があります。シェルプロンプトで以下のコマンドを実行します。

```
rpm --import /usr/share/rhn/RPM-GPG-KEY
```

RPM 検証用のインストール済みの全キーの一覧を表示するには、以下のコマンドを実行します:

```
rpm -qa gpg-pubkey*
```

Red Hat のキーの場合は、以下のような出力が表示されます:

```
gpg-pubkey-db42a60e-37ea5438
```

特定のキーの詳細を表示するには、**rpm -qi** の後に上記のコマンドの出力を続けて入力して下さい。

```
rpm -qi gpg-pubkey-db42a60e-37ea5438
```

B.3.2. パッケージの署名の検証

ビルダーの GnuPG キーをインポートした後に RPM ファイルの GnuPG 署名を確認するには、以下のコマンドを使用します RPM パッケージのファイル名に置き換えて下さい):

```
rpm -K <rpm-file>
```

正常に実行された場合には、**md5 gpg OK** というメッセージが表示されます。これは、パッケージの署名が検証され、破損していないことが確認されたので、安全にインストール/使用できることを意味します。

B.4. 実用的で一般的な RPM の使用例

RPM はシステムの管理や問題の診断と修正を行う上で便利なツールです。すべてのオプションを理解するには、いくつかの例を実際に見るのが最もよい方法です。

- 誤って何らかのファイルを削除してしまったものの、何を削除したかが分からないとします。システム全体を検証して足りないものを調べるには、以下のコマンドを試すことができます。

```
rpm -Va
```

足りないファイルがあるか、壊れているファイルがあるように感じる場合は、おそらくパッケージを再インストールするか、いったんアンインストールして再インストールする必要があります。

- 所属先の分からないファイルを見つけたとします。そのファイルが含まれるパッケージを検索するには、以下のように入力します。

```
rpm -qf /usr/bin/ghostscript
```

出力は以下のようになります:

```
ghostscript-8.70-1.el6.x86_64
```

- 次のような方法で上記 2 つの例を組み合わせることもできます。例えば、**/usr/bin/paste** に問題があるとします。このプログラムが含まれるパッケージを検証しようにも **paste** がどのパッケージに含まれるか分かりません。この場合は、次のコマンドを入力します。

```
rpm -Vf /usr/bin/paste
```

該当するパッケージが検証されます。

- 特定のプログラムに関して詳しい情報が必要なら、次のコマンドを入力して、そのプログラムを含むパッケージに付随するドキュメントの場所を確認することができます。

```
rpm -qdf /usr/bin/free
```

出力は以下のようになります。

```
/usr/share/doc/procps-3.2.8/BUGS
/usr/share/doc/procps-3.2.8/FAQ
/usr/share/doc/procps-3.2.8/NEWS
/usr/share/doc/procps-3.2.8/TODO
/usr/share/man/man1/free.1.gz
/usr/share/man/man1/pgrep.1.gz
/usr/share/man/man1/pkill.1.gz
/usr/share/man/man1/pmap.1.gz
/usr/share/man/man1/ps.1.gz
/usr/share/man/man1/pwdx.1.gz
/usr/share/man/man1/skill.1.gz
/usr/share/man/man1/slabtop.1.gz
/usr/share/man/man1/snice.1.gz
/usr/share/man/man1/tload.1.gz
/usr/share/man/man1/top.1.gz
/usr/share/man/man1/uptime.1.gz
/usr/share/man/man1/w.1.gz
/usr/share/man/man1/watch.1.gz
/usr/share/man/man5/sysctl.conf.5.gz
/usr/share/man/man8/sysctl.8.gz
/usr/share/man/man8/vmstat.8.gz
```

- 新しい RPM が見つかったものの、それが何かが分からないとします。何らかの情報を検索するには、以下のコマンド を使用します。

```
rpm -qip crontabs-1.10-32.1.el6.noarch.rpm
```

出力は以下のようになります。

```
Name        : crontabs                      Relocations: (not
relocatable)
Version     : 1.10                               Vendor: Red Hat,
Inc.
Release     : 32.1.el6                       Build Date: Thu 03 Dec
2009 02:17:44 AM CET
Install Date: (not installed)                Build Host: js20-bc1-
11.build.redhat.com
Group       : System Environment/Base        Source RPM: crontabs-
1.10-32.1.el6.src.rpm
Size        : 2486                              License: Public
Domain and GPLv2
Signature   : RSA/8, Wed 24 Feb 2010 08:46:13 PM CET, Key ID
938a80caf21541eb
Packager    : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
Summary     : Root crontab files used to schedule the execution of
programs
Description :
The crontabs package contains root crontab files and directories.
You will need to install cron daemon to run the jobs from the
crontabs.
The cron daemon such as cronie or fcron checks the crontab files to
see when particular commands are scheduled to be executed.  If
commands
are scheduled, it executes them.
Crontabs handles a basic system function, so it should be installed
on
your system.
```

- RPM がインストールする **crontabs** ファイルを見ることができます。以下のように入力します。

```
rpm -qlp crontabs-1.10-32.1.el6.noarch.rpm
```

出力は以下のようになります。

```
/etc/cron.daily
/etc/cron.hourly
/etc/cron.monthly
/etc/cron.weekly
/etc/crontab
/usr/bin/run-parts
/usr/share/man/man4/crontabs.4.gz
```

以上、いくつか例を紹介しました。使用していくにつれて、RPM の用途がたくさんあることに気づくでしょう。

B.5. その他のリソース

RPM は、パッケージのクエリ、インストール、アップグレード、削除を実行するための多くのオプションと方法を備えた非常に複雑なユーティリティです。RPM の詳細については、以下のリソースを参照してください。

B.5.1. インストールされているドキュメント

- **`rpm --help`** — このコマンドを実行すると RPM のパラメータのクイックリファレンスが表示されます。
- **`man rpm`** — RPM man ページでは、**`rpm --help`** コマンドよりさらに詳しい RPM のパラメータに関する情報が提供されています。

B.5.2. 役立つ Web サイト

- RPM の Web サイト — http://www.rpm.org/
- RPM のメーリングリストを購読、アーカイブを閲覧するには、こちらをクリックしてください。— https://lists.rpm.org/mailman/listinfo/rpm-list

B.5.3. 関連資料

『Maximum RPM』 — http://www.redhat.com/docs/books/max-rpm/

『Maximum RPM』 は、HTML または PDF でオンラインまたはダウンロードして読むことができます。また、一般的な RPM 使用方法から RPM のビルド、rpmlib を使ってのプログラミングまですべてを網羅しています。

付録C X WINDOW SYSTEM

Red Hat Enterprise Linux の中核はカーネルですが、多くのユーザーにとってオペレーティングシステムの顔となるのは、別名 *X* と呼ばれる *X Window System* によって提供されるグラフィカル環境です。

1984 年 6 月の X Window System のリリース以前に存在していた環境を含め、UNIX 業界では他のウィンドウ環境が存在していました。それでもなお、X は長年に渡って Red Hat Enterprise Linux など大半の UNIX 系オペレーティングシステムの既定のグラフィカル環境としてあり続けています。

Red Hat Enterprise Linux のグラフィカル環境は、X Window System と関連技術の開発と戦略を管理するために創設されたオープンソース組織である *X.Org Foundation* により提供されています。X.Org は、世界中の数百人の開発者が関与している大規模で急速に発展しているプロジェクトです。これは、各種のハードウェアデバイスとアーキテクチャーを幅広くサポートし、数え切れないほどのオペレーティングシステムとプラットフォーム上で機能します。

X Window System は、クライアントサーバーアーキテクチャーを使用します。この主な目的は、広範囲なコンピュータとグラフィックマシンで稼働するネットワーク透過なウィンドウシステムを提供することです。*X サーバー* (**Xorg** バイナリ) は、ネットワークまたはローカルのループバックインターフェースを介した *X クライアント* アプリケーションからの接続を待機します。X サーバーは、ビデオカード、モニター、キーボード、マウスなどのハードウェアと通信します。X クライアントアプリケーションは、ユーザースペースにあり、ユーザー用に *グラフィカルユーザーインターフェース* (*GUI*) を作成してユーザーの要求を X サーバーに渡します。

C.1. X サーバー

Red Hat Enterprise Linux 6 は、ビデオドライバー、EXA、特に以前のリリースに加えてプラットフォームサポートの拡張機能を含む X サーバーバージョンを使用します。さらに、このリリースには X サーバー向けの自動設定機能だけでなく、大半のマウスとキーボードなどカーネルが把握している全入力デバイスに対応する汎用入力ドライバーである、**evdev** も含まれています。

X11R7.1 は、X Window System のモジュール化を特別に活用するための最初のリリースでした。このリリースは、X を論理的に明確なモジュールに分割したため、オープンソースの開発者はこれまでより簡単にシステムにコードを書き込むことができます。

現行のリリースでは、すべてのライブラリ、ヘッダー、バイナリは **/usr/** ディレクトリ下にあります。X クライアント及びサーバーアプリケーション用の設定ファイルは、**/etc/X11/** ディレクトリにあります。これには、X サーバーや X ディスプレイマネージャーの他、多くのベースコンポーネント用の設定ファイルが含まれます。

新しい Fontconfig ベースのフォントアーキテクチャーの設定ファイルは、引き続き **/etc/fonts/fonts.conf** にあります。フォントの設定/追加方法の詳細については、「フォント」を参照して下さい。

X サーバーは、幅広いハードウェアで高度なタスクを実行するため、それが機能するハードウェアの詳しい情報が必要となります。X サーバーは、それが実行する大半のハードウェアを自動的に検出し、それに応じて自己設定することができます。別の方法として、設定ファイルでハードウェアを手動で指定することも可能です。

X パッケージがインストール用に選択されている限り、Red Hat Enterprise Linux システムのインストーラーである Anaconda は、X のインストールと設定を自動的に行います。X サーバーにより管理されているモニター、ビデオカードまたは他のデバイスに変更が加えられると、ほとんどの場合 X はそうした変更を自動検出して、再設定します。稀に手動で X を再設定しなければならない場合もあります。

C.2. デスクトップ環境とウィンドウマネージャー

X サーバーが実行すると、X クライアントアプリケーションはそれに接続し、ユーザー用に GUI を作成することができます。Red Hat Enterprise Linux には、ベーシックな *Tab Window Manager* (twm) から大半の Red Hat Enterprise Linux ユーザーが精通している高度に開発されたインタラクティブなデスクトップ環境 (*GNOME* や *KDE*) まで、利用可能な GUI が各種あります。

より総合的な GUI である後者のデスクトップ環境を作成するには、X クライアントアプリケーションの 2 つのメインクラスである *ウィンドウマネージャー* と *デスクトップ環境* を X サーバーに接続する必要があります。

C.2.1. デスクトップ環境

デスクトップ環境は、様々な X クライアントと統合して、一般的なグラフィカルユーザー環境と開発プラットフォームを作成します。

デスクトップ環境に備わっている高度な機能により、X クライアントと他の実行中のプロセスとの通信が可能になります。また一方で、ドラッグアンドドロップの操作など高度なタスクを実行するために、書き込まれたすべてのアプリケーションがデスクトップ環境で機能できるようにします。

Red Hat Enterprise Linux では 2 つのデスクトップ環境を採用しています。

- **GNOME** — GTK+ 2 グラフィカルツールキットを基にした Red Hat Enterprise Linux 向けのデフォルトのデスクトップ環境です。
- **KDE** — Qt 4 グラフィカルツールキットを基にしたもう 1 つのデスクトップ環境です。

GNOME と KDE どちらにも、ワードプロセッサー、スプレッドシート、Web ブラウザーなどの高度な生産性アプリケーションが備わっており、GUI の外観をカスタマイズするツールも提供します。さらに、GTK+ 2 と Qt ライブラリの両方が存在している場合は、KDE アプリケーションは GNOME で実行でき、その逆も同様です。

C.2.2. ウィンドウマネージャー

ウィンドウマネージャー は、デスクトップ環境の一部、または場合によってはスタンドアロンである X クライアントプログラムです。この主要な目的は、グラフィカルウィンドウを配置/サイズ変更/移動する方法を制御することです。また、ウィンドウマネージャーは、タイトルバー、ウィンドウフォーカスの動作、ユーザー指定のキー及びマウスボタンの組み合わせの制御も行います。

Red Hat Enterprise Linux リポジトリは、4 つの異なるウィンドウマネージャーを提供します。

`metacity`

Metacity のウィンドウマネージャーは、GNOME のデフォルトウィンドウマネージャーです。これは、カスタムのテーマをサポートするシンプルかつ効率的なウィンドウマネージャーです。このウィンドウマネージャーは、GNOME デスクトップがインストールされた場合に依存関係として自動的にプルされます。

`kwin`

KWin のウィンドウマネージャーは、KDE 用のデフォルトウィンドウマネージャーです。これは、カスタムのテーマをサポートする効率的なウィンドウマネージャーです。このウィンドウマネージャーは、KDE デスクトップがインストールされた場合に依存関係として自動的にプルされます。

`compiz`

Compiz のコンポジット型ウィンドウマネージャーは OpenGL をベースにしており、3D グラフィックスハードウェアを使用して、ウィンドウ管理のための高速な合成のデスクトップ効果を作成することができます。立体的なワークスペースなどの高度な機能は、ロード可能なプラグインとして実

装されています。このウィンドウマネージャーを実行するには、**compiz** パッケージをインストールする必要があります。

mwm

Motif Window Manager (**mwm**) は、簡易のスタンドアロンウィンドウマネージャーです。これはスタンドアロンとして設計されているため、GNOME や KDE との併用はお勧めしません。このウィンドウマネージャーを実行するには、**openmotif** パッケージをインストールする必要があります。

twm

最低限の機能を搭載した *Tab Window Manager* (**twm**) は、利用可能なウィンドウマネージャーの中でも最も基本的なツールを提供しており、スタンドアロンとしてまたはデスクトップ環境と共に使用することができます。このウィンドウマネージャーを実行するには、**xorg-x11-twm** パッケージをインストールする必要があります。

C.3. X サーバーの設定ファイル

X サーバーは、単一のバイナリ実行ファイル **/usr/bin/Xorg** であり、このファイルをポイントするシンボリックリンク **X** も提供されています。関連する設定ファイルは、**/etc/X11/** と **/usr/share/X11/** ディレクトリ内に格納されています。

X Window System は 2 種類の設定スキームに対応します。 **xorg.conf.d** ディレクトリの設定ファイルには、ベンダー及びディストリビューションより事前構成された設定が含まれています。これらのファイルは手動では編集しないことをお勧めします。一方で、**xorg.conf** ファイル内の設定はすべて手動で実行されますが、ほとんどの状況では必要ではありません。

注記

ディスプレイと周辺機器に必要なすべてのパラメーターは、インストール時に自動検出され、設定されます。X サーバー用の設定ファイルである **/etc/X11/xorg.conf** は、これまでのリリースでは必要でしたが、X Window System の現行のリリースには含まれていません。ただし、新しいハードウェアを設定するために手動でファイルを作成する場合や複数のビデオカードを備えた環境を設定する場合、デバッグする場合に引き続き有用です。

/usr/lib/xorg/modules/ (または **/usr/lib64/xorg/modules/**) ディレクトリには、ランタイム時に動的にロードできる X サーバーモジュールが含まれています。デフォルトでは、**/usr/lib/xorg/modules/** にある一部のモジュールのみ、X サーバーにより自動的にロードされます。

Red Hat Enterprise Linux 6 のインストール時に、X の設定ファイルは、インストールプロセス中に HAL (Hardware Abstraction Layer) 設定バックエンドにより収集されたシステムハードウェアに関する情報を使用して作成されます。X サーバーは起動すると常に、HAL に入力デバイスの一覧を要求して、それぞれ各ドライバーを使って追加します。新しい入力デバイスがプラグインされる場合や既存の入力デバイスが削除される場合、HAL は常にその変更を X サーバーに伝えます。この通知システムにより、**xorg.conf** ファイルで設定された **mouse**、**kbd** または **vmmouse** のドライバーを使用するデバイスは、デフォルトで X サーバーにより無視されます。詳細については、「**ServerFlags** セクション」を参照して下さい。追加設定は **/etc/X11/xorg.conf.d/** ディレクトリにあり、HAL によって取得した全設定を上書き/拡張することが可能です。

C.3.1. 設定の構造

X 設定ファイルの形式は、多くの様々なセクションで構成されており、システムハードウェアが持つ特有の側面に対応します。各セクションは **Section "*section-name*"** の行で始まります。"*section-name*" はセクションのタイトルで、**EndSection** の行で終わります。各セクションには、オプションの名前と 1 つ以上のオプション値を含む行があります。一部は、二重引用符 (**"**) で囲まれている場合があります。

/etc/X11/xorg.conf ファイル内の一部のオプションは、ブールスイッチを使用して機能をオン/オフします。使用可能な値は以下のとおりです。

- **1**、**on**、**true** または **yes** — オプションをオンにします。
- **0**、**off**、**false** または **no** — オプションをオフにします。

以下は、キーボード用の標準的な設定ファイルです。ハッシュ記号 (#) で始まる行は、X サーバーからは読み取り不可能ですが、人間が読み取れるコメントとして使用されます:

```
# This file is autogenerated by system-setup-keyboard. Any
# modifications will be lost.

Section "InputClass"
  Identifier  "system-setup-keyboard"
  MatchIsKeyboard "on"
  Option    "XkbModel"  "pc105"
  Option    "XkbLayout" "cz,us"
# Option    "XkbVariant"  "(null)"
  Option    "XkbOptions"
"terminate:ctrl_alt_bksp,grp:shifts_toggle,grp_led:scroll"
EndSection
```

C.3.2. xorg.conf.d ディレクトリ

X サーバーは 2 つの設定ディレクトリをサポートします。**/usr/share/X11/xorg.conf.d/** は、ベンダーやサードパーティのパッケージとは別個の設定ファイルを提供します。このディレクトリ内のファイルへの変更は、**/etc/X11/xorg.conf** ファイル内で指定された設定により上書きできます。**/etc/X11/xorg.conf.d/** ディレクトリは、ユーザー固有の設定を格納します。

設定ディレクトリにある **.conf** の接尾辞が付いたファイルは、起動時に X サーバーにより解析され、従来の **xorg.conf** 設定ファイルの一部のように扱われます。これらのファイルにはセクションを 1 つ以上含むことができます。セクションにあるオプションの詳細や設定ファイルの一般的なレイアウトについては、「**xorg.conf** ファイル」 または **xorg.conf(5)** の man ページを参照して下さい。X サーバーは、基本的に設定ファイルの集まりを最後の **xorg.conf** からのエントリを含む 1 つの大きなファイルとして扱います。カスタム設定を **/etc/xorg.conf** に含めて、ディストリビューションにより提供される設定スニペットにディレクトリを残しておくことが、ユーザーには推奨されます。

C.3.3. xorg.conf ファイル

これまでの X Window System のリリースでは、**/etc/X11/xorg.conf** ファイルは X の初期設定を格納するために使用されていました。X サーバーにより管理されているモニター、ビデオカードや他のデバイスが原因で変更が生じた場合、ファイルは手動で編集する必要がありました。Red Hat Enterprise Linux では、**/etc/X11/xorg.conf** ファイルを手動で作成/編集する必要はほとんどありません。ただし、通常とは異なるハードウェア設定のトラブルシューティングや調整を行う時のために、各種セクションと利用可能なオプションのパラメーターについて理解しておくことは有用です。

以下では、重要なセクションが標準的な **/etc/X11/xorg.conf** ファイルに表示される順で説明され

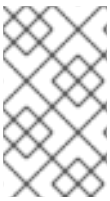
ています。X サーバーの設定ファイルについての詳しい情報は、**xorg.conf(5)** の man ページに記載されています。下記で説明されるほとんどの設定オプションは、一般的な設定状況では必要ないため、本項の大部分は上級ユーザー向けです。

C.3.3.1. InputClass セクション

InputClass は、ホットプラグデバイスをはじめ、単一のデバイスではなくデバイスのクラスに適用する新しいタイプの設定セクションです。**InputClass** セクションの範囲は、指定されるマッチにより制限されます。入力デバイスに適用するためには、すべてのマッチは以下の例のようにデバイスに適用する必要があります:

```
Section "InputClass"
   Identifier      "touchpad catchall"
   MatchIsTouchpad "on"
   Driver           "synaptics"
EndSection
```

このスニペットが **xorg.conf** ファイルか **xorg.conf.d** ディレクトリにある場合、システムにあるすべての touchpad は **synaptics** ドライバーに割り当てられます。



注記

xorg.conf.d ディレクトリ内の設定ファイルは英数字順で並び替えられているため、上記の例の **Driver** 設定は、以前に設定されているドライバーオプションを上書きします。汎用性が高いクラスほど、最初の方に表示されます。

match のオプションは、セクションが適用できるデバイスを指定します。デバイスがマッチするには、match の全オプションが対応しなければなりません。以下は、**InputClass** セクションでよく使用されるオプションです。

- **MatchIsPointer**、**MatchIsKeyboard**、**MatchIsTouchpad**、**MatchIsTouchscreen**、**MatchIsJoystick** — デバイスの種類を指定するブール値のオプションです。
- **MatchProduct "*product_name*"** — このオプションは、*product_name* サブ文字列がデバイスの製品名に存在する場合にマッチします。
- **MatchVendor "*vendor_name*"** — このオプションは、*vendor_name* サブ文字列がデバイスのベンダー名に存在する場合にマッチします。
- **MatchDevicePath "*/path/to/device*"** — このオプションは、デバイスのパスが "*/path/to/device*" テンプレートで与えられたパターンに対応する場合にデバイスにマッチします。例えば **/dev/input/event*** などです。詳細については **fnmatch(3)** の man ページを参照して下さい。
- **MatchTag "*tag_pattern*"** — このオプションは、HAL 設定バックエンドにより割り当てられた 1 つ以上のタグが *tag_pattern* パターンにマッチする場合にマッチします。

設定ファイルに、**InputClass** セクションが複数ある場合があります。これらのセクションはオプションで、自動的に追加される時に入力デバイスのクラスを設定するために使用されます。入力デバイスは、複数の **InputClass** セクションにマッチすることができます。こうしたセクションを編成する場合、重複が発生すると、各入力クラスは以前のクラスの設定を上書きできるため、汎用的なマッチを特定のマッチより優先させることが推奨されます。

C.3.3.2. InputDevice セクション

各 **InputDevice** セクションは、X サーバーに対し 1 つの入力デバイスを設定します。これまでは、通常システムにはキーボード用に 1 つ以上の **InputDevice** セクションがあり、大半のマウス設定は自動的に検出されていました。

Red Hat Enterprise Linux 6 では、大半の設定には **InputDevice** を設定する必要はありません。xorg-x11-drv-* 入力ドライバーパッケージは HAL による自動設定を提供します。キーボードとマウス用のデフォルトドライバーは **evdev** です。

以下の例は、キーボード用の標準的な **InputDevice** セクションです:

```
Section "InputDevice"
  Identifier "Keyboard0"
  Driver "kbd"
  Option "XkbModel" "pc105"
  Option "XkbLayout" "us"
EndSection
```

以下は、**InputDevice** セクションでよく使用されるエントリです。

- **Identifier** — この **InputDevice** セクションに一意の名前を指定します。このエントリは必須です。
- **Driver** — X がデバイス用にロードする必要があるデバイスドライバーの名前を指定します。**AutoAddDevices** オプションが有効な場合 (デフォルト設定) は、**Driver "mouse"** または **Driver "kbd"** があるすべての入力デバイスセクションは無視されます。これが必要な理由は、レガシーなマウス及びキーボードのドライバーと新しい **evdev** 汎用ドライバーとの間に競合があるためです。その代わりに、サーバーは全入力デバイスにバックエンドからの情報を使用します。**xorg.conf** にあるすべてのカスタム入力デバイス設定は、バックエンドへ移動することをお勧めします。ほとんどの場合、バックエンドは HAL となり、設定場所は **/etc/X11/xorg.conf.d** ディレクトリになります。
- **Option** — デバイスに関する必要なオプションを指定します。

  マウスを指定して、デバイス用のすべての自動検出値を上書きすることもできます。以下のオプションは、通常 **xorg.conf** ファイルにマウスを追加する場合に含まれています。

  - **Protocol** — **IMPS/2** など、マウスにより使用されるプロトコルを指定します。
  - **Device** — 物理デバイスの場所を指定します。
  - **Emulate3Buttons** — 両方のマウスボタンを同時に押した時に、2 ボタンマウスが 3 ボタンマウスのように機能するかどうかを指定します。

  このセクションに関する有効なオプションの全一覧については、**xorg.conf(5)** の man ページを参照して下さい。

C.3.3.3. ServerFlags セクション

オプションの **ServerFlags** セクションには、X サーバーのその他のグローバル設定が含まれています。このセクションの全設定は、**ServerLayout** セクションにあるオプションにより上書きできます (詳細は 「ServerLayout セクション」 を参照)。

ServerFlags セクション内の各エントリは、1 行に表示されます。また、**Option** という単語で始ま

り、二重引用符 (") で囲まれたオプションがその後に続きます。

以下は、**ServerFlags** セクションのサンプルです:

```
Section "ServerFlags"
  Option "DontZap" "true"
EndSection
```

最も役立つオプションの一部を以下に表示します。

- **"DontZap" "*boolean*"** — *<boolean>* の値が **true** に設定されている場合、X サーバーを直ちに終了させる **Ctrl+Alt+Backspace** キーの組み合わせは使用できなくなります。

注記

このオプションが有効であっても、X Keyboard Extension (XKB) マップでキーの組み合わせを使用前に設定する必要があります。マップにキーの組み合わせを追加する方法の 1 つとして、以下のコマンドを実行します:

```
setxkbmap -option "terminate:ctrl_alt_bksp"
```

- **"DontZoom" "*boolean*"** — *<boolean>* の値が **true** に設定されている場合、この設定では **Ctrl+Alt+Keypad-Plus** と **Ctrl+Alt+Keypad-Minus** キーの組み合わせを使用して、設定済みのビデオ解像度が変更しないようします。

- **"AutoAddDevices" "*boolean*"** — *<boolean>* の値が **false** に設定されている場合、サーバーは入力デバイスにホットプラグせずに、代わりに **xorg.conf** ファイルで設定済みのデバイスだけに依存します。入力デバイスの詳細については、「**InputDevice** セクション」 を参照して下さい。このオプションはデフォルトで有効であり、HAL (hardware abstraction layer) はデバイス検出のバックエンドとして使用されます。

C.3.3.4. ServerLayout セクション

ServerLayout セクションは、X サーバーにより制御されている入力出力デバイスをバインドします。このセクションは、最低限 1 つの入力デバイスと 1 つの出力デバイスを指定する必要があります。デフォルトでは、モニター (出力デバイス) とキーボード (入力デバイス) が指定されています。

以下の例は、標準的な **ServerLayout** セクションです:

```
Section "ServerLayout"
  Identifier "Default Layout"
  Screen 0 "Screen0" 0 0
  InputDevice "Mouse0" "CorePointer"
  InputDevice "Keyboard0" "CoreKeyboard"
EndSection
```

以下は、**ServerLayout** セクションでよく使用されるエントリです。

- **Identifier** — **ServerLayout** セクションに一意の名前を指定します。

- **Screen** — X サーバーと併用される **Screen** セクションの名前を指定します。**Screen** オプションは複数存在する場合があります。

以下の例は、標準的な **Screen** エントリです:

```
Screen 0 "Screen0" 0 0
```

この例の最初の番号である **Screen** エントリ (**0**) は、第 1 のモニターコネクターまたはビデオカードの ヘッド が、識別子 **"Screen0"** がある **Screen** セクションで指定されている設定を使用していることを示しています。

識別子 **"Screen0"** がある **Screen** セクションの例は、 「**Screen** セクション」 に記載されています。

ビデオカードにヘッドが複数ある場合、別の番号と別の **Screen** セクション識別子を持つ他の **Screen** エントリが必要です。

"Screen0" の右側の数字は、画面の左上隅に X と Y の絶対座標を渡します (デフォルトは **0 0**)。

- **InputDevice** — X サーバーにより使用される **InputDevice** セクションの名前を指定します。

  InputDevice エントリは 2 つ以上あることが推奨されます。1 つはデフォルトのマウス用、もう 1 つはデフォルトのキーボード用です。**CorePointer** と **CoreKeyboard** のオプションは、これらがプライマリのマウスとキーボードであることを示しています。**AutoAddDevices** オプションが有効な場合、このエントリは **ServerLayout** セクションで指定される必要はありません。**AutoAddDevices** オプションが無効な場合は、マウス、キーボードともに自動検出され、デフォルト値が有効です。

- **Option "*option-name*"** — このセクションに追加のパラメーターを指定するオプションのエントリです。ここに一覧表示されている全オプションは、**ServerFlags** セクションに表示されているものを上書きします。

  <option-name> を **xorg.conf(5)** の man ページにあるこのセクション用に表示されている有効なオプションと置換して下さい。

/etc/X11/xorg.conf ファイルには、複数の **ServerLayout** セクションを入れることができます。ただしデフォルトでは、サーバーはそれが見つける 1 番目のセクションのみを読み取ります。別の **ServerLayout** セクションがある場合は、**Xorg -layout <layoutname>** コマンドのように X セッションの開始時にコマンドラインの引数として指定することが可能です。

C.3.3.5. Files セクション

Files セクションは、フォントパスなど X サーバーに不可欠なサービス用のパスを設定します。通常こうしたパスは自動検出されるため、このセクションはオプションです。このセクションを使用して、自動検出された値を上書きできます。

以下の例は、標準的な **Files** セクションです:

```
Section "Files"
  RgbPath "/usr/share/X11/rgb.txt"
  FontPath "unix/:7100"
EndSection
```

以下は、**Files** セクションでよく使用されるエントリです。

- **ModulePath** — X サーバーモジュールを格納する別のディレクトリを指定するオプションのパラメーターです。

C.3.3.6. Monitor セクション

各 **Monitor** セクションでは、システムにより使用されるモニターのタイプを 1 つ設定します。現在、大半のモニターは自動検出されるため、このエントリはオプションです。

以下の例は、モニターの標準的な **Monitor** セクションです。

```
Section "Monitor"
  Identifier "Monitor0"
  VendorName "Monitor Vendor"
  ModelName "DDC Probed Monitor - ViewSonic G773-2"
  DisplaySize 320 240
  HorizSync 30.0 - 70.0
  VertRefresh 50.0 - 180.0
EndSection
```

以下は、**Monitor** セクションでよく使用されるエントリです。

- **Identifier** — この **Monitor** セクションに一意の名前を指定します。このエントリは必須です。
- **VendorName** — モニターのベンダーを指定するオプションのパラメーターです。
- **ModelName** — モニターのモデル名を指定するオプションのパラメーターです。
- **DisplaySize** — モニターの画像領域の物理サイズをミリメートル単位で指定するオプションのパラメーターです。
- **HorizSync** — モニターと互換性のある水平同期周波数の範囲を kHz 単位で指定します。これらの値により、X サーバーはモニター用の組み込み/指定された **Modeline** エントリの有効性を決定できます。
- **VertRefresh** — モニターに対応する垂直リフレッシュ周波数の範囲を kHz 単位で指定します。これらの値により、X サーバーはモニターに対する組み込み/指定された **Modeline** エントリの有効性を決定することができます。
- **Modeline** — 水平同期及び垂直リフレッシュ解像度である、特定の解像度を持つモニターに対して別途ビデオモードを指定するオプションのパラメーターです。**Modeline** エントリの詳しい説明については、**xorg.conf(5)** の man ページを参照して下さい。
- **Option "*option-name*"** — セクションに別途のパラメーターを指定するオプションのエントリです。*<option-name>* を **xorg.conf(5)** の man ページにあるこのセクション用に表示されている有効なオプションと置換して下さい。

C.3.3.7. Device セクション

各 **Device** セクションは、システム上の 1 つのビデオカードを設定します。1 つの **Device** セクションが最低限ですが、マシンにインストールされているそれぞれのビデオカードに対し追加のインスタンスが生成する場合があります。

以下の例は、ビデオカードの標準的な **Device** セクションです:

```
Section "Device"
  Identifier "Videocard0"
  Driver "mga"
  VendorName "Videocard vendor"
  BoardName "Matrox Millennium G200"
  VideoRam 8192
  Option "dpms"
EndSection
```

以下は、**Device** セクションでよく使用されるエントリです。

- **Identifier** — この **Device** セクションに一意の名前を指定します。このエントリは必須です。
- **Driver** — ビデオカードを使用するために X サーバーがロードする必要があるドライバーを指定します。ドライバーの一覧は **/usr/share/hwdata/videodrivers** にあり、hwdata パッケージでインストールされます。
- **VendorName** — ビデオカードのベンダーを指定するオプションのパラメーターです。
- **BoardName** — ビデオカードの名前を指定するオプションのパラメーターです。
- **VideoRam** — ビデオカードで利用可能な RAM の容量をキロバイト単位で指定するオプションのパラメーターです。この設定は、X サーバーがビデオ RAM の容量を検出するためにプローブできないビデオカードにのみ必要です。
- **BusID** — ビデオカードのバスの場所を指定するエントリです。ビデオカードが 1 つのみのシステム上では、**BusID** エントリはオプションであり、デフォルトの **/etc/X11/xorg.conf** ファイルに存在していない場合もあります。ただし、複数のビデオカードがあるシステムでは、**BusID** エントリは必須です。
- **Screen** — **Device** セクションで設定するモニターコネクターまたはビデオカードのヘッドを指定するオプションのエントリです。このオプションが有用なのは、複数のヘッドを持つビデオカードに対してのみです。

  複数のモニターが同じビデオカードの異なるヘッドに接続されている場合は、別々の **Device** セクションを終了して、それぞれのセクションに異なる **Screen** 値がなければなりません。

  Screen エントリの値は、整数でなければなりません。ビデオカードの第 1 ヘッドには **0** の値があります。それぞれ追加のヘッドの値は、この値を 1 つ増分させたものです。
- **Option "*option-name*"** — セクションに別途のパラメーターを指定するオプションのエントリです。*<option-name>* を **xorg.conf(5)** の man ページにあるこのセクション用に表示されている有効なオプションと置換して下さい。

  さらに一般的なオプションの 1 つは **"dpms"** (VESA 基準である Display Power Management Signaling) です。これにより、モニターのエネルギースター準拠の設定をアクティベートします。

C.3.3.8. Screen セクション

各 **Screen** セクションは、**Device** セクションと **Monitor** セクションをそれぞれ参照することによって、1 つのビデオカード (またはビデオカードヘッド) を 1 つのモニターにバインドします。1 つの **Screen** セクションが最低限ですが、マシンに存在する各ビデオカードとモニターの組み合わせに対し追加のインスタンスが生成される場合もあります。

以下の例は、標準的な **Screen** セクションです:

```
Section "Screen"
  Identifier "Screen0"
  Device "Videocard0"
  Monitor "Monitor0"
  DefaultDepth 16

  SubSection "Display"
    Depth 24
    Modes "1280x1024" "1280x960" "1152x864" "1024x768" "800x600" "640x480"
  EndSubSection

  SubSection "Display"
    Depth 16
    Modes "1152x864" "1024x768" "800x600" "640x480"
  EndSubSection
EndSection
```

以下は、**Screen** セクションでよく使用されるエントリです。

- **Identifier** — この **Screen** セクションに一意の名前を指定します。このエントリは必須です。
- **Device** — **Device** セクションに一意の名前を指定します。このエントリは必須です。
- **Monitor** — **Monitor** セクションに一意の名前を指定します。これは、特定の **Monitor** セクションが **xorg.conf** ファイルで定義されている場合にのみ必須です。通常、モニターは自動検出されます。
- **DefaultDepth** — デフォルトの色深度をビット単位で指定します。前の例では、**16** (何千もの色を提供) がデフォルトです。1 つの **DefaultDepth** エントリのみが許可されますが、Xorg コマンドラインのオプション **-depth *<n>*** により上書きされる場合があります。***<n>*** は指定する追加の深度です。
- **SubSection "Display"** — 特定の色深度で利用できる画面モードを指定します。**Screen** セクションは、複数の **Display** サブセクションを持つことができます。画面モードは自動検出されるためこれは完全にオプションです。

  通常このサブセクションは、自動検出されたモードを上書きします。
- **Option "*option-name*"** — セクションに別途のパラメーターを指定するオプションのエントリです。*<option-name>* を **xorg.conf(5)** の man ページにあるこのセクション用に表示されている有効なオプションと置換して下さい。

C.3.3.9. DRI セクション

オプションの **DRI** セクションは、*DRI (Direct Rendering Infrastructure: ダイレクトレンダリングインフラストラクチャー)* のパラメーターを指定します。DRI は、3D ソフトウェアアプリケーションが大半の最新ビデオハードウェアに組み込まれている 3D ハードウェアアクセラレーションの機能を活用できるようにするインターフェースです。加えて、ビデオカードドライバーによりサポートされている場合は、DRI はハードウェアアクセラレーションにより 2D 性能が向上します。

DRI の Group と Mode はデフォルト値に自動的に初期化されるため、このセクションはほとんど使用されません。別の Group や Mode が必要な場合は、このセクションを **xorg.conf** ファイルに追加すると、デフォルト値が上書きされます。

以下の例は、標準的な **DRI** セクションです:

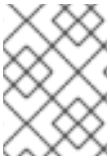
```
Section "DRI"
  Group 0
  Mode 0666
EndSection
```

それぞれビデオカードは様々な方法で DRI を使用するため、最初に http://dri.freedesktop.org/wiki/ を参照してから本セクションに追加するようにして下さい。

C.4. フォント

Red Hat Enterprise Linux は、*Fontconfig* サブシステムを使用して、X Window System でフォントの管理と表示を行います。*Fontconfig* サブシステムは、フォント管理を簡素化して、アンチエイリアシングなど高度な表示機能を提供します。このシステムは、**Qt 3** もしくは **GTK+ 2** グラフィカルツールキットまたはその最新版を使用してプログラムされたアプリケーション用に自動的に使用されます。

Fontconfig フォントサブシステムにより、アプリケーションは直接システムのフォントにアクセスし、*X FreeType インターフェースライブラリ* (*Xft*) やアンチエイリアシングなど高度な機能を持つ Fontconfig フォントをレンダリングする他のレンダリングメカニズムを使用することができます。グラフィカルなアプリケーションは、Fontconfig と共に Xft ライブラリを使用して画面にテキストを描画することができます。



注記

Fontconfig は **/etc/fonts/fonts.conf** 設定ファイルを使用します。このファイルを手動で編集することはお勧めしません。

警告

ユーザーがリモート X アプリケーションの実行を要求するシステムでは、**fonts** グループがインストールされている必要があります。このためには、インストーラーでグループを指定するか、インストール後に **yum groupinstall fonts** コマンドを実行します。

C.4.1. Fontconfig へのフォントの追加

Fontconfig のサブシステムに新しいフォントを追加するプロセスは簡単です。

1. 個々のユーザー用にフォントを追加するには、ユーザーのホームディレクトリの **.fonts/** ディレクトリに新しいフォントをコピーします。

   システム全体にフォントを追加するには、新しいフォントを **/usr/share/fonts/** ディレクトリにコピーします。ユーザーがインストールしたフォントとデフォルトのフォントを区別できるように、**local/** やそれに類似する新しいサブディレクトリを作成することは好ましい方

法です。

2. フォント情報のキャッシュを更新するには、root として **fc-cache** コマンドを実行します:

```
fc-cache <path-to-font-directory>
```

このコマンドでは、*<path-to-font-directory>* を新しいフォントを含むディレクトリ (**/usr/share/fonts/local/** または **/home/*<user>*/.fonts/**) と置換して下さい。

注記

個々のユーザーがフォントをインタラクティブにインストールするには、**Nautilus** アドレスバーに **fonts:///** を入力して、そこに新しいフォントファイルをドラッグしても可能です。

C.5. ランレベルと X

Red Hat Enterprise Linux インストーラーは、ほとんどの場合 *ランレベル 5* として知られるグラフィカルなログイン環境にブートするようマシンを設定します。ただし、*ランレベル 3* と呼ばれるテキストのみのマルチユーザーモードにブートして、そこから X セッションを始めることも可能です。

次項からは、X がランレベル 3 及び 5 で起動する方法を確認します。ランレベルの詳細については、「デフォルトのランレベルの設定」 を参照して下さい。

C.5.1. ランレベル 3

ランレベル 3 で実行している場合に X セッションを開始する最良の方法は、ログインして、**startx** と入力することです。**startx** コマンドは、**xinit** コマンドのフロントエンドで、X サーバー (**Xorg**) を起動して、それに X クライアントアプリケーションを接続します。ユーザーはすでにランレベル 3 でシステムにログインしているため、**startx** はディスプレイマネージャーの起動やユーザーの認証を行うことはありません。ディスプレイマネージャーの詳細については、「ランレベル 5」 を参照して下さい。

1. **startx** コマンドが実行されると、ユーザーのホームディレクトリにある **.xinitrc** ファイルを探して、デスクトップ環境、場合によっては実行する他の X クライアントアプリケーションも定義します。**.xinitrc** ファイルがない場合は、代わりにシステムデフォルトの **/etc/X11/xinit/xinitrc** ファイルを使用します。

2. その後、デフォルトの **xinitrc** スクリプトは、次のようなユーザー定義のファイルとデフォルトのシステムファイルを検索します。ユーザーのホームディレクトリにある **.Xresources**、**.Xmodmap**、**.Xkbmap**、及び **/etc/X11/** ディレクトリにある **Xresources**、**Xmodmap**、**Xkbmap** です。**Xmodmap** 及び **Xkbmap** ファイルが存在する場合、それらはキーボードを設定する **xmodmap** ユーティリティにより使用されます。**Xresources** ファイルは、特定の設定値をアプリケーションに割り当てるために読み取られます。

3. 上記のオプションの設定後、**xinitrc** スクリプトは **/etc/X11/xinit/xinitrc.d/** ディレクトリにあるすべてのスクリプトを実行します。このディレクトリにある重要なスクリプトは **xinput.sh** であり、デフォルトの言語などの設定を行います。

4. **xinitrc** スクリプトはユーザーのホームディレクトリにある **.Xclients** の実行を試行します。それがない場合は、**/etc/X11/xinit/Xclients** の実行を試みます。**Xclients** ファイルの目的は、デスクトップ環境、場合によっては簡易なウィンドウマネージャーのみを起動することです。ユーザーのホームディレクトリにある **.Xclients** スクリプトは **.Xclients-**

default ファイルにあるユーザー指定のデスクトップ環境を起動します。**.Xclients** がユーザーのホームディレクトリに存在していない場合は、標準の **/etc/X11/xinit/Xclients** スクリプトは、最初に GNOME、次に KDE、続いて **twm** と、異なるデスクトップ環境の起動を試みます。

ランレベル 3 で実行している場合、X セッションの終了後、ユーザーはテキストモードのユーザーセッションに戻されます。

C.5.2. ランレベル 5

ランレベル 5 でシステムをブートする場合、*ディスプレイマネージャー* と呼ばれる特殊な X クライアントアプリケーションが起動します。デスクトップ環境またはウィンドウマネージャーを起動する前に、ユーザーはディスプレイマネージャーを使用して認証される必要があります。

システムにインストールされているデスクトップ環境に応じて、ユーザー認証を実行できるディスプレイマネージャーは 3 種類あります。

- **GDM** (GNOME Display Manager) — Red Hat Enterprise Linux のデフォルトのディスプレイマネージャーです。**GNOME** により、ユーザーはシステムの言語設定/シャットダウン/再起動/ログインを行うことができます。
- **KDM** — KDE のディスプレイマネージャーであり、ユーザーはシステムのシャットダウン/再起動/ログインを行うことができます。
- **xdm** (X Window Display Manager) — ユーザーのシステムへのログインのみを可能にする非常に簡易なディスプレイマネージャーです。

ランレベル 5 でブートする場合、**/etc/X11/prefdm** スクリプトは、**/etc/sysconfig/desktop** ファイルを参照することで優先するディスプレイマネージャーを決定します。このファイルのオプション一覧は、次のファイルにあります:

```
/usr/share/doc/initscripts-<version-number>/sysconfig.txt
```

<version-number> は **initscripts** パッケージのバージョン番号です。

それぞれのディスプレイマネージャーは、**/etc/X11/xdm/Xsetup_0** ファイルを参照して、ログイン画面を設定します。ユーザーがシステムにログインすると、**/etc/X11/xdm/GiveConsole** スクリプトが実行され、コンソールのオーナーシップをユーザーに割り当てます。その後、**/etc/X11/xdm/Xsession** スクリプトが実行され、ランレベル 3 から X を起動する場合に **xinitrc** スクリプトで通常実行される多くのタスクを行うことができます。それには、システム及びユーザーリソースの設定だけでなく、**/etc/X11/xinit/xinitrc.d/** ディレクトリのスクリプトの実行も含まれます。

GNOME または **KDE** ディスプレイマネージャーを使用してユーザー認証を行う場合、**System (システム)** → **Preferences (設定)** → **More Preferences (他の個人設定)** → **Sessions (セッション)** の順に選択するとアクセスできる **Sessions (セッション)** のメニューアイテムから選択することで、ユーザーは使用したいデスクトップ環境を指定することができます。デスクトップ環境がディスプレイマネージャーで指定されていない場合は、**/etc/X11/xdm/Xsession** スクリプトはユーザーのホームディレクトリにある **.xsession** 及び **.Xclients** ファイルを確認して、ロードするデスクトップ環境を決定します。最後の手段として、**/etc/X11/xinit/Xclients** ファイルを使用して、ランレベル 3 の時と同様の方法で使用するデスクトップ環境またはウィンドウマネージャーを選択します。

ユーザーがデフォルトのディスプレイ (**:0**) で X セッションを終了してログアウトすると、**/etc/X11/xdm/TakeConsole** スクリプトが実行され、root ユーザーにコンソールのオーナーシップが再度割り当てられます。元のディスプレイマネージャーは、ユーザーがログインした後も引き

続き稼働し、新しいディスプレイマネージャーを起動することで制御します。これにより X サーバーが再起動して、新しいログインウィンドウが表示され、プロセス全体を最初から再度やり直します。

ユーザーはランレベル 5 の X からログアウト後、ディスプレイマネージャーに戻されます。

ディスプレイマネージャーがユーザー認証を管理する方法の詳細については、**/usr/share/doc/gdm-*<version-number>*/README** または **xdm** の man ページを参照して下さい。*<version-number>* は、インストールされている **gdm** パッケージのバージョン番号です。

C.6. その他のリソース

X サーバーやそれに接続するクライアント、様々なデスクトップ環境とウィンドウマネージャーに関する入手可能な詳しい情報は多数あります。

C.6.1. インストールされているドキュメント

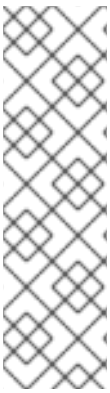
- **/usr/share/X11/doc/** — X Window System アーキテクチャーに関する詳しいドキュメント、さらには新しいユーザーとしての Xorg プロジェクトに関する追加情報の取得方法が記載されています。
- **/usr/share/doc/gdm-*<version-number>*/README** — ディスプレイマネージャーがユーザー認証を管理する方法が記載されています。
- **man xorg.conf** — **xorg.conf** 設定ファイル内の様々なセクションに関する意味と構文など、そのファイルに関する情報が含まれています。
- **man Xorg** — **Xorg** ディスプレイサーバーについて説明しています。

C.6.2. 役立つ Web サイト

- http://www.X.org/ — X.Org Foundation のホームページです。Red Hat Enterprise Linux に含まれている X Window System のメジャーリリースを作成して、必要なハードウェアの制御と GUI 環境の提供を行います。
- http://dri.sourceforge.net/ — DRI (Direct Rendering Infrastructure) プロジェクトのホームページです。DRI は、X のコアハードウェア 3D アクセラレーションコンポーネントです。
- http://www.gnome.org/ — GNOME プロジェクトのホームページです。
- http://www.kde.org/ — KDE デスクトップ環境のホームページです。

付録D SYSCONFIG ディレクトリ

本付録には、**/etc/sysconfig/** ディレクトリ内にあるファイルとディレクトリの機能や内容に関する概要が記載されています。こうしたファイルの多くには、非常に特定のまたは稀な状況でのみ使用される様々なオプションが含まれるため、本付録に全情報が記載されているわけではありません。



注記

/etc/sysconfig/ ディレクトリの実際のコンテンツは、マシンにインストールされているプログラムにより異なります。設定ファイルが所属するパッケージ名を知りたい場合は、シェルプロンプトで以下のコマンドを入力します:

```
~]$ yum provides /etc/sysconfig/filename
```

Red Hat Enterprise Linux で新しいパッケージをインストールする方法の詳細については、「パッケージのインストール」 を参照して下さい。

D.1. /ETC/SYSCONFIG/ ディレクトリのファイル

次項からは、通常 **/etc/sysconfig/** ディレクトリ内にあるファイルについて説明しています。

D.1.1. /etc/sysconfig/arpwatch

/etc/sysconfig/arpwatch ファイルは、ブート時に **arpwatch** デーモンに引数を渡すために使用されます。デフォルトでは、以下のオプションが含まれています。

OPTIONS=*value*

arpwatch デーモンに渡す追加のオプションです。例えば:

```
OPTIONS="-u arpwatch -e root -s 'root (Arpwatch)'"
```

D.1.2. /etc/sysconfig/authconfig

/etc/sysconfig/authconfig ファイルは、ホストで使用される認証を設定します。デフォルトでは、以下のオプションが含まれています。

USEMKHOMEDIR=*boolean*

初回ログイン時にユーザー用のホームディレクトリの作成を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USEMKHOMEDIR=no
```

USEPAMACCESS=*boolean*

PAM 認証を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USEPAMACCESS=no
```

USESSSDAUTH=*boolean*

SSSD 認証を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USESSSDAUTH=no
```

USESHADOW=*boolean*

シャドウパスワードを有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USESHADOW=yes
```

USEWINBIND=*boolean*

ユーザーアカウント設定用に Winbind の使用を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USEWINBIND=no
```

USEDB=*boolean*

FAS 認証を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USEDB=no
```

USEFPRINTD=*boolean*

指紋認証を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USEFPRINTD=yes
```

FORCESMARTCARD=*boolean*

スマートカード認証の強制を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
FORCESMARTCARD=no
```

PASSWDALGORITHM=*value*

パスワードアルゴリズムです。指定可能な *value* は **bigcrypt**、**descrypt**、**md5**、**sha256**、**sha512** です。例えば:

```
PASSWDALGORITHM=sha512
```

USELDAPAUTH=*boolean*

LDAP 認証を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USELDAPAUTH=no
```

USELOCAUTHORIZE=*boolean*

ローカルユーザー用のローカル認証を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USELOCAUTHORIZE=yes
```

USECRACKLIB=*boolean*

CrackLib の使用を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USECRACKLIB=yes
```

USEWINBINDAUTH=*boolean*

Winbind 認証を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USEWINBINDAUTH=no
```

USESMARTCARD=*boolean*

スマートカード認証を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USESMARTCARD=no
```

USELDAP=*boolean*

ユーザーアカウント設定に LDAP の使用を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USELDAP=no
```

USENIS=*boolean*

ユーザーアカウント設定に NIS の使用を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USENIS=no
```

USEKERBEROS=*boolean*

Kerberos 認証を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USEKERBEROS=no
```

USESYSNETAUTH=*boolean*

ネットワークサービスを使ったシステムアカウントの認証を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USESYSNETAUTH=no
```

USESMBAUTH=*boolean*

SMB 認証を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USESMBAUTH=no
```

USESSSD=*boolean*

ユーザー情報取得で SSSD の使用を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USESSSD=no
```

USEHESIOD=*boolean*

Hesoid ネームサービスの使用を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USEHESIOD=no
```

このトピックについての詳しい情報は、11章*認証の設定* を参照して下さい。

D.1.3. /etc/sysconfig/autofs

/etc/sysconfig/autofs ファイルは、デバイスの自動マウント用のカスタムオプションを定義します。このファイルはファイルシステムの使用時に自動的にそれらをマウントして、しばらく活動がないとアンマウントする自動マウントデーモンの動作を制御します。ファイルシステムには、ネットワークファイルシステム、CD-ROM ドライブ、ディスケット、及びその他のメディアが含まれます。

デフォルトでは、以下のようなオプションが含まれています。

MASTER_MAP_NAME=*value*

マスターマップのデフォルト名です。例えば:

```
MASTER_MAP_NAME="auto.master"
```

TIMEOUT=*value*

デフォルトのマウントタイムアウトです。例えば:

```
TIMEOUT=300
```

NEGATIVE_TIMEOUT=*value*

マウント試行が失敗した時のデフォルト設定の負のタイムアウトです。例えば:

```
NEGATIVE_TIMEOUT=60
```

MOUNT_WAIT=*value*

mount から応答を待つ時間です。例えば:

```
MOUNT_WAIT=-1
```

UMOUNT_WAIT=*value*

umount から応答を待つ時間です。例えば:

```
UMOUNT_WAIT=12
```

BROWSE_MODE=*boolean*

マップの閲覧を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
BROWSE_MODE="no"
```

MOUNT_NFS_DEFAULT_PROTOCOL=*value*

mount.nfs により使用されるデフォルトのプロトコルです。例えば:

```
MOUNT_NFS_DEFAULT_PROTOCOL=4
```

APPEND_OPTIONS=*boolean*

グローバルオプションの置換の代わりに追記を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
APPEND_OPTIONS="yes"
```

LOGGING=*value*

デフォルトのロギングレベルです。*value* は **none**、**verbose**、**debug** のいずれかでなければなりません。例えば:

```
LOGGING="none"
```

LDAP_URI=*value*

空白で区切られたサーバー URI の一覧です。形式は ***protocol*://*server*** です。例えば:

```
LDAP_URI="ldaps://ldap.example.com/"
```

LDAP_TIMEOUT=*value*

同期 API 呼び出しのタイムアウトです。例えば:

```
LDAP_TIMEOUT=-1
```

LDAP_NETWORK_TIMEOUT=*value*

ネットワーク応答のタイムアウトです。例えば:

```
LDAP_NETWORK_TIMEOUT=8
```

SEARCH_BASE=*value*

マップ検索用のベース識別名 (DN) です。例えば:

```
SEARCH_BASE=""
```

AUTH_CONF_FILE=*value*

SASL 認証設定ファイルがあるデフォルトの場所です。例えば:

```
AUTH_CONF_FILE="/etc/autofs_ldap_auth.conf"
```

MAP_HASH_TABLE_SIZE=*value*

マップキャッシュ用のハッシュテーブルサイズです。例えば:

```
MAP_HASH_TABLE_SIZE=1024
```

USE_MISC_DEVICE=*boolean*

autofs の他のデバイスの使用を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
USE_MISC_DEVICE="yes"
```

OPTIONS=*value*

LDAP デーモンに渡す追加のオプションです。例えば:

```
OPTIONS=""
```

D.1.4. /etc/sysconfig/clock

/etc/sysconfig/clock ファイルは、システムのハードウェアクロックで読み取られた値の解釈を制御します。このファイルは **日付/時刻のプロパティ** ツールで使用されるため、手動での編集はお勧めしません。デフォルトでは、以下のオプションが含まれています。

ZONE=*value*

/etc/localtime のコピー元である **/usr/share/zoneinfo** 下のタイムゾーンファイルです。例えば:

```
ZONE="Europe/Prague"
```

日付/時刻のプロパティ ツールとその使用方法の詳細については、「日付/時刻のプロパティのツール」を参照して下さい。

D.1.5. /etc/sysconfig/dhcpd

/etc/sysconfig/dhcpd ファイルは、ブート時に **dhcpd** デーモンに引数を渡すために使用されます。デフォルトでは、以下のオプションが含まれています。

DHCPDARGS=*value*

dhcpd デーモンに渡す追加のオプションです。例えば:

```
DHCPDARGS=
```

DHCP とその使用方法の詳細については、13章*DHCP サーバー* を参照して下さい。

D.1.6. /etc/sysconfig/firstboot

/etc/sysconfig/firstboot ファイルは、**firstboot** ユーティリティを実行するかどうかを定義します。デフォルトでは、以下のオプションが含まれています。

RUN_FIRSTBOOT=*boolean*

firstboot プログラムの実行を有効 (**YES**) /無効 (**NO**) にするためのブール値です。例えば:

```
RUN_FIRSTBOOT=NO
```

システムの初回ブート時に、**init** プログラムは **/etc/rc.d/init.d/firstboot** スクリプトを呼び出して、**/etc/sysconfig/firstboot** ファイルを探します。このファイルに **RUN_FIRSTBOOT=NO** オプションが含まれていない場合は、**firstboot** プログラムは実行され、ユーザーをシステムの初期設定へと導きます。

注記

次回のシステムブート時に **firstboot** プログラムを起動するには、**RUN_FIRSTBOOT** オプションの値を **YES** に変更して、シェルプロンプトで以下を入力します:

```
~]# chkconfig firstboot on
```

D.1.7. /etc/sysconfig/i18n

/etc/sysconfig/i18n 設定ファイルは、デフォルトの言語、サポートされている全言語、及びデフォルトのシステムフォントを定義します。デフォルトでは、以下のオプションが含まれています。

LANG=*value*

デフォルトの言語です。例えば:

```
LANG="en_US.UTF-8"
```

SUPPORTED=*value*

コロンで区切られたサポートされている言語の一覧です。例えば:

```
SUPPORTED="en_US.UTF-8:en_US:en"
```

SYSFONT=*value*

デフォルトのシステムフォントです。例えば:

```
SYSFONT="latarcyrheb-sun16"
```

D.1.8. /etc/sysconfig/init

/etc/sysconfig/init ファイルは、ブートプロセス中にシステムが表示され、機能する方法を制御します。デフォルトでは、以下のオプションが含まれています。

BOOTUP=*value*

ブートのスタイルです。値は **color** (標準色のブートディスプレイ)、**verbose** (より多くの情報を提供する旧型のディスプレイ)、あるいは ANSI 形式を使用しない新しいスタイルディスプレイ用の別の値でなければなりません。例えば:

```
BOOTUP=color
```

RES_COL=*value*

状態ラベルが始まるカラムの番号です。例えば:

```
RES_COL=60
```

MOVE_TO_COL=*value*

RES_COL で指定されたカラムへカーソルを移動するターミナルの配列です (上記参照)。例えば:

```
MOVE_TO_COL="echo -en \\033[${RES_COL}G"
```

SETCOLOR_SUCCESS=*value*

成功の色を設定するターミナルの配列です。例えば:

```
SETCOLOR_SUCCESS="echo -en \\033[0;32m"
```

SETCOLOR_FAILURE=*value*

失敗の色を設定するターミナルの配列です。例えば:

```
SETCOLOR_FAILURE="echo -en \\033[0;31m"
```

SETCOLOR_WARNING=*value*

警告の色を設定するターミナルの配列です。例えば:

```
SETCOLOR_WARNING="echo -en \\033[0;33m"
```

SETCOLOR_NORMAL=*value*

デフォルトの色を設定するターミナルの配列です。例えば:

```
SETCOLOR_NORMAL="echo -en \\033[0;39m"
```

LOGLEVEL=*value*

初期コンソールのログレベルです。*value* は、**1** (カーネルパニックのみ) から **8** (デバッグ情報などすべて) までの範囲内にする必要があります。例えば:

```
LOGLEVEL=3
```

PROMPT=*boolean*

ホットキーの対話型スタートアップを有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
PROMPT=yes
```

AUTOSWAP=*boolean*

スワップ署名を持つデバイスのプローブを有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
AUTOSWAP=no
```

ACTIVE_CONSOLES=*value*

アクティブなコンソールの一覧です。例えば:

```
ACTIVE_CONSOLES=/dev/tty[1-6]
```

SINGLE=*value*

シングルユーザーモードのタイプです。*value* は、**/sbin/sulogin** (ユーザーはログインするためのパスワードをプロンプトされる)、または **/sbin/sushell** (ユーザーは直接ログインする) でなければなりません。例えば:

```
SINGLE=/sbin/sushell
```

D.1.9. /etc/sysconfig/ip6tables-config

/etc/sysconfig/ip6tables-config ファイルは、ブート時や **ip6tables** サービスが起動する度に IPv6 パケットフィルタリングを設定するためにカーネルが使用する情報を格納します。ただし、**ip6tables** ルールを初めて使用される場合は変更しないことをお勧めします。デフォルトでは、以下のオプションが含まれています。

IP6TABLES_MODULES=*value*

ファイアウォールのルール適用後に、ロードされる空白で区切られたヘルパーの一覧です。例えば:

```
IP6TABLES_MODULES="ip_nat_ftp ip_nat_irc"
```

IP6TABLES_MODULES_UNLOAD=*boolean*

ファイアウォールが停止/再起動した時のモジュールのアンロードを有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
IP6TABLES_MODULES_UNLOAD="yes"
```

IP6TABLES_SAVE_ON_STOP=*boolean*

ファイアウォールの停止時に現在のファイアウォールルールの保存を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
IP6TABLES_SAVE_ON_STOP="no"
```

IP6TABLES_SAVE_ON_RESTART=*boolean*

ファイアウォールの再起動時に現在のファイアウォールルールの保存を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
IP6TABLES_SAVE_ON_RESTART="no"
```

IP6TABLES_SAVE_COUNTER=*boolean*

ルールおよびチェーンカウンターの保存を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
IP6TABLES_SAVE_COUNTER="no"
```

IP6TABLES_STATUS_NUMERIC=*boolean*

状態出力において IP アドレスとポート番号の数式での表示を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
IP6TABLES_STATUS_NUMERIC="yes"
```

IP6TABLES_STATUS_VERBOSE=*boolean*

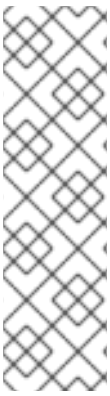
状態出力でのパケット数とバイト数に関する情報の表示を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
IP6TABLES_STATUS_VERBOSE="no"
```

IP6TABLES_STATUS_LINENUMBERS=*boolean*

状態出力での行数の表示を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
IP6TABLES_STATUS_LINENUMBERS="yes"
```



注記

ip6tables コマンドを使用して、手動でルールを作成できます。作成後は、シェルプロンプトで以下を入力します:

```
~]# service ip6tables save
```

これにより、**/etc/sysconfig/ip6tables** にルールが追加されます。このファイルが作成されると、そこに保存されたあらゆるファイアウォールのルールはシステムをリブートするかサービスを再起動することで保持されます。

D.1.10. /etc/sysconfig/keyboard

/etc/sysconfig/keyboard ファイルは、キーボードの動作を制御します。デフォルトでは、以下のオプションが含まれています。

KEYTABLE=*value*

キーテーブルファイルの名前です。キーテーブルとして使用可能なファイルは、**/lib/kbd/keymaps/i386/** ディレクトリで開始し、そこから様々なキーボードレイアウトへ分岐し、すべて ***value*.kmap.gz** と名前が付けられます。**KEYTABLE** 設定に一致する最初のファイル名が使用されます。例えば:

```
KEYTABLE="us"
```

MODEL=*value*

キーボードのモデルです。例えば:

```
MODEL="pc105+inet"
```

LAYOUT=*value*

キーボードのレイアウトです。例えば:

```
LAYOUT="us"
```

KEYBOARDTYPE=*value*

キーボードタイプです。指定可能な値は **pc** (PS/2 キーボード) か **sun** (Sun キーボード) です。例えば:

```
KEYBOARDTYPE="pc"
```

D.1.11. /etc/sysconfig/ldap

/etc/sysconfig/ldap ファイルには、LDAP サーバーの基本設定が含まれています。デフォルトでは、以下のオプションが含まれています。

SLAPD_OPTIONS=*value*

slapd デーモンに渡す追加のオプションです。例えば:

```
SLAPD_OPTIONS="-4"
```

SLURPD_OPTIONS=*value*

slurpd デーモンに渡す追加のオプションです。例えば:

```
SLURPD_OPTIONS=""
```

SLAPD_LDAP=*boolean*

TCP での LDAP の使用 (**ldap:///**) を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
SLAPD_LDAP="yes"
```

SLAPD_LDAPI=*boolean*

IPC での LDAP の使用 (**ldapi:///**) を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
SLAPD_LDAPI="no"
```

SLAPD_LDAPS=*boolean*

TLS での LDAP の使用 (**ldaps:///**) を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
SLAPD_LDAPS="no"
```

SLAPD_URLS=*value*

空白で区切られた URL の一覧です。例えば:

```
SLAPD_URLS="ldapi:///var/lib/ldap_root/ldapi ldapi:/// ldaps:///"
```

SLAPD_SHUTDOWN_TIMEOUT=*value*

slapd がシャットダウンするのを待機する時間です。例えば:

```
SLAPD_SHUTDOWN_TIMEOUT=3
```

SLAPD_ULIMIT_SETTINGS=*value*

slapd デーモンの開始前に **ulimit** に渡すパラメーターです。例えば:

```
SLAPD_ULIMIT_SETTINGS=""
```

LDAP とその設定の詳細については、「OpenLDAP」 を参照して下さい。

D.1.12. /etc/sysconfig/named

/etc/sysconfig/named ファイルは、ブート時に **named** デーモンに引数を渡すために使用されます。デフォルトでは、以下のオプションが含まれています。

ROOTDIR=*value*

named デーモンが実行する chroot 環境です。*value* はディレクトリのフルパスである必要があります。例えば:

```
ROOTDIR="/var/named/chroot"
```

chroot 環境は最初に設定しなければならない点に注意して下さい (詳細については、シェルプロンプトで **info chroot** と入力して下さい)。

OPTIONS=*value*

named に渡す追加のオプションです。例えば:

```
OPTIONS="-6"
```

-t オプションは使用しないことをお勧めします。代わりに、上記で説明した **ROOTDIR** を使用して下さい。

KEYTAB_FILE=*value*

キータブのファイル名です。例えば:

```
KEYTAB_FILE="/etc/named.keytab"
```

BIND DNS サーバーとその設定の詳細については、「BIND」 を参照して下さい。

D.1.13. /etc/sysconfig/network

/etc/sysconfig/network ファイルは、希望するネットワーク設定に関する情報を指定するために使用されます。デフォルトでは、以下のオプションが含まれています。

NETWORKING=*boolean*

ネットワーキングを有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
NETWORKING=yes
```

HOSTNAME=*value*

マシンのホスト名です。例えば:

```
HOSTNAME=penguin.example.com
```

GATEWAY=*value*

ネットワークのゲートウェイの IP アドレスです。例えば:

```
GATEWAY=192.168.1.0
```

警告

カスタムの init スクリプトを使用して、ネットワークを設定しないで下さい。ブート後にネットワークサービスを再起動すると、ネットワーク init スクリプトの外で実行されるネットワーク設定を行うカスタムの init スクリプトにより不測な事態が発生します。

D.1.14. /etc/sysconfig/ntpd

/etc/sysconfig/ntpd ファイルは、ブート時に **ntpd** デーモンに引数を渡すために使用されます。デフォルトでは、以下のオプションが含まれています。

OPTIONS=*value*

ntpd に渡す追加のオプションです。例えば:

```
OPTIONS="-u ntp:ntp -p /var/run/ntpd.pid -g"
```

ntpd デーモンを設定する方法の詳細については、「ネットワーク時刻プロトコルのプロパティ」 または 「ネットワーク時刻プロトコルの設定」 を参照して下さい。

D.1.15. /etc/sysconfig/quagga

/etc/sysconfig/quagga ファイルには、Quagga デーモンの基本設定が含まれています。デフォルトでは、以下のオプションが含まれています。

QCONFDIR=*value*

Quagga デーモンの設定ファイルがあるディレクトリです。例えば:

```
QCONFDIR="/etc/quagga"
```

BGPD_OPTS=*value*

bgpd デーモンに渡す追加のオプションです。例えば:

```
BGPD_OPTS="-A 127.0.0.1 -f ${QCONFDIR}/bgpd.conf"
```

OSPF6D_OPTS=*value*

ospf6d デーモンに渡す追加のオプションです。例えば:

```
OSPF6D_OPTS="-A ::1 -f ${QCONFDIR}/ospf6d.conf"
```

OSPFD_OPTS=*value*

ospfd デーモンに渡す追加のオプションです。例えば:

```
OSPFD_OPTS="-A 127.0.0.1 -f ${QCONFDIR}/ospfd.conf"
```

RIPD_OPTS=*value*

ripd デーモンに渡す追加のオプションです。例えば:

```
RIPD_OPTS="-A 127.0.0.1 -f ${QCONFDIR}/ripd.conf"
```

RIPNGD_OPTS=*value*

ripngd デーモンに渡す追加のオプションです。例えば:

```
RIPNGD_OPTS="-A ::1 -f ${QCONFDIR}/ripngd.conf"
```

ZEBRA_OPTS=*value*

zebra デーモンに渡す追加のオプションです。例えば:

```
ZEBRA_OPTS="-A 127.0.0.1 -f ${QCONFDIR}/zebra.conf"
```

ISISD_OPTS=*value*

isisd デーモンに渡す追加のオプションです。例えば:

```
ISISD_OPTS="-A ::1 -f ${QCONFDIR}/isisd.conf"
```

WATCH_OPTS=*value*

watchquagga デーモンに渡す追加のオプションです。例えば:

```
WATCH_OPTS="-Az -b_ -r/sbin/service_%s_restart -s/sbin/service_%s_start
-k/sbin/service_%s_stop"
```

WATCH_DAEMONS=*value*

空白で区切られたモニターされるデーモンの一覧です。例えば:

```
WATCH_DAEMONS="zebra bgpd ospfd ospf6d ripd ripngd"
```

D.1.16. /etc/sysconfig/radvd

/etc/sysconfig/radvd ファイルは、ブート時に **radvd** デーモンに引数を渡すために使用されます。デフォルトでは、以下のオプションが含まれています。

OPTIONS=*value*

radvd デーモンに渡す追加のオプションです。例えば:

```
OPTIONS="-u radvd"
```

D.1.17. /etc/sysconfig/samba

/etc/sysconfig/samba ファイルは、ブート時に Samba デーモンに引数を渡すために使用されます。デフォルトでは、以下のオプションが含まれています。

SMBDOPTIONS=*value*

smbd に渡す追加のオプションです。例えば:

```
SMBDOPTIONS="-D"
```

NMBDOPTIONS=*value*

nmbd に渡す追加のオプションです。例えば:

```
NMBDOPTIONS="-D"
```

WINBINDOPTIONS=*value*

winbindd に渡す追加のオプションです。例えば:

```
WINBINDOPTIONS=""
```

Samba とその設定の詳細については、「Samba」 を参照して下さい。

D.1.18. /etc/sysconfig/saslauthd

/etc/sysconfig/saslauthd ファイルは、ブート時にどの引数を SASL 認証サーバーである **saslauthd** にわたすかを制御するために使われます。デフォルトでは、以下のオプションを含んでいます。

SOCKETDIR=*value*

saslauthd のリスニングソケット用のディレクトリです。例えば：

```
SOCKETDIR=/var/run/saslauthd
```

MECH=*value*

ユーザーパスワードの確認に使用する認証メカニズムです。例えば:

```
MECH=pam
```

DAEMONOPTS=*value*

saslauthd サービスを開始する **/etc/rc.d/init.d/saslauthd** init スクリプトが使用する **daemon()** 機能にわたされるオプションです。例えば:

```
DAEMONOPTS="--user saslauth"
```

FLAGS=*value*

saslauthd サービスにわたす追加のオプションです。例えば:

```
FLAGS=
```

D.1.19. /etc/sysconfig/selinux

/etc/sysconfig/selinux ファイルには、SELinux の基本設定オプションが含まれています。これは、**/etc/selinux/config** へのシンボリックリンクであり、デフォルトでは以下のオプションが含まれています。

SELINUX=*value*

セキュリティポリシーです。*value* は、**enforcing** (セキュリティポリシーを常に強制)、**permissive** (ポリシーを強制する代わりに適切な警告を表示)、または **disabled** (ポリシーを使用しない) のいずれかが可能です。例えば:

```
SELINUX=enforcing
```

SELINUXTYPE=*value*

保護の種類です。*value* は、**targeted** (対象のプロセスを保護) または **mls** (マルチレベルセキュリティの保護) です。例えば:

```
SELINUXTYPE=targeted
```

D.1.20. /etc/sysconfig/sendmail

/etc/sysconfig/sendmail は、**Sendmail** アプリケーションのデフォルト値を設定するために使用されます。デフォルトでは、以下の値が含まれています。

DAEMON=*boolean*

デーモンとして **sendmail** の実行を有効 (**yes**) /無効 (**no**) にするためのブール値です。例えば:

```
DAEMON=yes
```

QUEUE=*value*

メッセージが処理される間隔です。例えば:

```
QUEUE=1h
```

Sendmail とその設定の詳細については、「Sendmail」 を参照して下さい。

D.1.21. /etc/sysconfig/spamassassin

/etc/sysconfig/spamassassin ファイルは、ブート時に **spamd** デーモン (**Spamassassin** のデーモン化されたバージョン) に引数を渡すために使用されます。デフォルトでは、以下のオプションが含まれています。

SPAMDOPTIONS=*value*

spamd デーモンに渡す追加のオプションです。例えば:

```
SPAMDOPTIONS="-d -c -m5 -H"
```

Spamassassin とその設定の詳細については、「スパムフィルター」 を参照して下さい。

D.1.22. /etc/sysconfig/squid

/etc/sysconfig/squid ファイルは、ブート時に **squid** デーモンに引数を渡すために使用されます。デフォルトでは、以下のオプションが含まれています。

SQUID_OPTS=*value*

squid デーモンに渡す追加のオプションです。例えば:

```
SQUID_OPTS=""
```

SQUID_SHUTDOWN_TIMEOUT=*value*

squid デーモンがシャットダウンするのを待機する時間です。例えば:

```
SQUID_SHUTDOWN_TIMEOUT=100
```

SQUID_CONF=*value*

デフォルトの設定ファイルです。例えば:

```
SQUID_CONF="/etc/squid/squid.conf"
```

D.1.23. /etc/sysconfig/system-config-users

/etc/sysconfig/system-config-users ファイルは、**ユーザー管理** ユーティリティの設定ファイルです。これを手動で編集することはお勧めしません。デフォルトでは、以下のオプションが含まれています。

FILTER=*boolean*

システムユーザーのフィルタリングを有効 (**true**) /無効 (**false**) にするためのブール値です。例えば:

```
FILTER=true
```

ASSIGN_HIGHEST_UID=*boolean*

新しく追加されたユーザーへの利用可能な最高値の UID の割り当てを有効 (**true**) /無効 (**false**) にするためのブール値です。例えば:

```
ASSIGN_HIGHEST_UID=true
```

ASSIGN_HIGHEST_GID=*boolean*

新しく追加されたグループへの利用可能な最高値の GID の割り当てを有効 (**true**) /無効 (**false**) にするためのブール値です。例えば:

```
ASSIGN_HIGHEST_GID=true
```

PREFER_SAME_UID_GID=*boolean*

新しく追加されたユーザーに同じ UID と GID の使用を可能な場合に有効 (**true**) /無効 (**false**) にするためのブール値です。例えば:

```
PREFER_SAME_UID_GID=true
```

ユーザー管理 とその使用方法の詳細については、「ユーザー管理ツールの使用」 を参照して下さい。

D.1.24. /etc/sysconfig/vncservers

/etc/sysconfig/vncservers ファイルは、*Virtual Network Computing* (*VNC*) サーバーが起動する方法を設定します。デフォルトでは、以下のオプションが含まれています。

VNCSERVERS=*value*

空白で区切られた ***display*:*username*** ペアの一覧です。例えば:

```
VNCSERVERS="2:myusername"
```

VNCSERVERARGS[*display*]=*value*

指定された *display* で実行する VNC サーバーに渡す追加の引数です。例えば:

```
VNCSERVERARGS[2]="-geometry 800x600 -nolisten tcp -localhost"
```

D.1.25. /etc/sysconfig/xinetd

/etc/sysconfig/xinetd ファイルは、ブート時に **xinetd** デーモンに引数を渡すために使用されます。デフォルトでは、以下のオプションが含まれています。

EXTRAOPTIONS=*value*

xinetd に渡す追加のオプションです。例えば:

```
EXTRAOPTIONS=""
```

XINETD_LANG=*value*

xinetd により開始されたすべてのサービスに渡されるロケール情報です。**xinetd** 環境からロケール情報を削除するには、空の文字列 (**""**) または **none** を使用することができます。例えば:

```
XINETD_LANG="en_US"
```

xinetd サービスの設定方法に関する詳細は、10章 *サービスとデーモン* を参照して下さい。

D.2. /ETC/SYSCONFIG/ ディレクトリ内のディレクトリ

以下のディレクトリは、通常 **/etc/sysconfig/** 内にあります。

/etc/sysconfig/cbq/

このディレクトリには、ネットワークインターフェース上で帯域幅を管理するにあたって *Class Based Queuing (クラスベースキューイング)* を行うために必要な設定ファイルが含まれています。CBQ は、IP アドレス、プロトコル、アプリケーションタイプの任意の組み合わせを基にして、ユーザートラフィックをクラスの階層に分けます。

/etc/sysconfig/networking/

このディレクトリは 廃止予定の **Network Administration Tool** (**system-config-network**) により使用され、そのコンテンツを手動で編集することはお勧めしません。グラフィカルな設定ツールを使用したネットワークインターフェースの設定についての詳細は、8章*NetworkManager* を参照して下さい。

/etc/sysconfig/network-scripts/

このディレクトリには、次のようなネットワーク関連の設定ファイルが含まれています。

- **eth0** イーサネットインターフェース用の **ifcfg-eth0** など、各設定済みのネットワークインターフェース用のネットワーク設定ファイル
- **ifup** や **ifdown** など、ネットワークインターフェースを有効/無効にするために使用されるスクリプト
- **ifup-isdn** や **ifdown-isdn** など、ISDN インターフェースを有効/無効にするために使用されるスクリプト
- 直接編集すべきではない各種の共有ネットワーク機能のスクリプト

/etc/sysconfig/network-scripts/ ディレクトリの詳細については、9章 *ネットワークインターフェース* を参照して下さい。

/etc/sysconfig/rhn/

このディレクトリには、Red Hat Network 用の設定ファイルと GPG キーが含まれています。このディレクトリ内のファイルは手動では編集しないことをお勧めします。Red Hat Network の詳細については、https://rhn.redhat.com/ にあるオンラインの Red Hat Network の Web サイトをご覧下さい。

D.3. その他のリソース

本章では、**/etc/sysconfig/** ディレクトリ内にあるファイルの紹介のみ行います。以下の資料に、さらに包括的な情報が含まれています。

D.3.1. インストールされているドキュメント

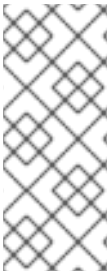
/usr/share/doc/initscripts-*version*/sysconfig.txt

/etc/sysconfig/ ディレクトリにある正式なファイルの一覧とそれらに利用可能な設定オプションが記載されています。

付録E PROC ファイルシステム

Linux カーネルには、コンピューター上の物理デバイスへのアクセスを制御し、それらのデバイスとプロセスがいつ、どのような方法で情報のやりとりを行うかをスケジュールするという、2 つの主要な機能があります。**/proc/** ディレクトリ (別名 **proc** ファイルシステム) には、カーネルの現在の状況を示す特殊なファイルの階層が含まれています。これにより、アプリケーション及びユーザーがシステムのカーネルビューを確認することができます

/proc/ ディレクトリには、システムのハードウェア及び実行中のプロセスに関する詳細情報が大量に格納されています。又、**/proc/** 内のファイルの一部をユーザーやアプリケーションが操作することによって、設定の変更をカーネルに伝達することもできます。



注記

2.6 カーネルの後続バージョンでは、**/proc/ide/** 及び **/proc/pci/** のディレクトリは廃止されています。**/proc/ide/** ファイルシステムに代わって **sysfs** 内のファイルが使用されるようになったので、PCI デバイ上の情報を取得するには、代わりに **lspci** を使用して下さい。**sysfs** または **lspci** に関する詳しい情報は、それぞれの **man** ページを参照して下さい。

E.1. 仮想ファイルシステム

Linux システムでは、すべてのデータが **ファイル** として保管されます。大半のユーザーは、主要な 2 つのファイルタイプ (テキストとバイナリ) について精通していますが、**/proc/** ディレクトリには、*仮想ファイル* と呼ばれる別のタイプのファイルも含まれています。このため、**/proc/** は、*仮想ファイルシステム* と呼ばれる場合もよくあります。

仮想ファイルには、固有の性質があります。大半の仮想ファイルは、サイズがゼロバイトで表示されていても、表示すると大量の情報が格納されている場合があります。又、仮想ファイルの時刻及び日付のスタンプの大半は、現在の時刻と日付を反映します。これは、常に更新されていることを示しています。

/proc/interrupts、**/proc/meminfo**、**/proc/mounts**、及び **/proc/partitions** などの仮想ファイルは、システムのハードウェアの最新状態のビューを提供します。**/proc/filesystems** ファイルや **/proc/sys/** ディレクトリなど、その他の仮想ファイルは、システムの設定情報及びインターフェースを提供します。

情報を体系化するために、同様のトピックに関する内容が記載されたファイルは、仮想ディレクトリ/サブディレクトリにグループ化されます。プロセスディレクトリには、システムで実行中の各プロセスに関する情報が格納されます。

E.1.1. 仮想ファイルの表示

/proc/ ファイル内の大半のファイルは、テキストファイルと同様に機能し、システム及びハードウェアの有用なデータを人間が読み取れるテキスト形式で保管します。このため、**cat**、**more**、**less** などを使用して内容を表示することができます。例えば、システムの CPU に関する情報を表示するには、**cat /proc/cpuinfo** を実行すると、以下のような出力が返されます:

```
processor : 0
vendor_id : AuthenticAMD
cpu family : 5
model  : 9
model name : AMD-K6(tm) 3D+
```

```
Processor stepping : 1 cpu
MHz  : 400.919
cache size : 256 KB
fdiv_bug : no
hlt_bug  : no
f00f_bug : no
coma_bug : no
fpu  : yes
fpu_exception : yes
cpuid level : 1
wp  : yes
flags  : fpu vme de pse tsc msr mce cx8 pge mmx syscall 3dnow k6_mtrr
bogomips : 799.53
```

/proc/ 内のファイルの一部には、人間には読み取れない情報が含まれています。このようなファイルから情報を取得するには、**lspci**、**apm**、**free**、**top** などのツールを使用して下さい。

注記

/proc/ ディレクトリ内の仮想ファイルには、root ユーザーしか読み取れないものがあります。

E.1.2. 仮想ファイルの変更

原則として、**/proc/** ディレクトリ内の仮想ファイルの大半は読み取り専用です。但し、一部は、カーネルの設定を調整するのに使用することができます。これは特に **/proc/sys/** サブディレクトリ内のファイルに該当します。

仮想ファイルの値を変更するには、以下のコマンドを使用します:

echo *value* > /proc/*file*

例えば、オンザフライでホスト名を変更するには、以下のコマンドを実行します:

echo *www.example.com* > /proc/sys/kernel/hostname

その他のファイルは、バイナリまたはブールスイッチとして機能します。**cat /proc/sys/net/ipv4/ip_forward** と入力すると、**0** (off または false) もしくは **1** (on または true) と返されます。**0** は、カーネルがネットワークパケットを転送していないことを示します。パケット転送をオンにするには、**echo 1 > /proc/sys/net/ipv4/ip_forward** を実行して下さい。

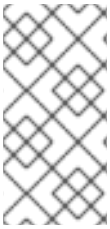
注記

/proc/sys/ サブディレクトリ内の設定を変更するためのもう 1 つのコマンドは、**/sbin/sysctl** です。このコマンドについての詳しい情報は、「sysctl コマンドの使用」 を参照して下さい。

/proc/sys/ サブディレクトリ内で使用可能な一部のカーネル設定ファイルの一覧は、「/proc/sys/」を参照して下さい。

E.2. PROC ファイルシステム内の最上位のファイル

以下は、**/proc/** ディレクトリの最上位にあるその他の有用な仮想ファイルの一覧です。



注記

本セクションに記載したファイルの内容は、大半の場合、お客様のマシン上にインストールされているファイルの内容とは異なります。これは、大部分の情報が、本ガイドを執筆する目的で使用している、Red Hat Enterprise Linux を実行中のハードウェア固有の情報であるためです。

E.2.1. /proc/buddyinfo

このファイルは、メモリ断片化問題の診断に主として使用されるファイルです。バディアルゴリズムを使用して、各カラムには、任意の時点で使用可能な、特定の順序 (特定のサイズ) のページ数が示されます。例えば、ゾーン *ダイレクトメモリアクセス* (DMA) では、$2^{(0*PAGE_SIZE)}$ メモリチャンクが 90 あります。又同様に、$2^{(1*PAGE_SIZE)}$ のメモリチャンクが 6 つ、$2^{(2*PAGE_SIZE)}$ のメモリチャンクが 2 つ使用可能です。

DMA の行はシステム上の最初の 16 MB、**HighMem** の行はシステム上の 4 GB を越える全メモリ、**Normal** の行はそれらの中間のサイズの全メモリを示しています。

/proc/buddyinfo の標準的な出力の例は以下の通りです:

```
Node 0, zone      DMA     90      6      2      1      1      ...
Node 0, zone   Normal   1650    310      5      0      0      ...
Node 0, zone  HighMem      2      0      0      1      1      ...
```

E.2.2. /proc/cmdline

このファイルには、カーネルの起動時に渡されるパラメーターが表示されます。**/proc/cmdline** ファイルのサンプルは以下のようになります:

```
ro root=/dev/VolGroup00/LogVol00 rhgb quiet 3
```

これは、カーネルが読み取り専用 (**ro**) でマウントされており、第 1 のボリュームグループ (**/dev/VolGroup00**) の第 1 の論理ボリューム (**LogVol00**) に配置されていることをいることを示しています。**LogVol00** は、非 LVM システム (Logical Volume Management) のディスクパーティションに相当します。これは、**/dev/VolGroup00** の概念が **/dev/hda1** と似ている一方で、はるかに拡張性が高いのと同様です。

Red Hat Enterprise Linux で使用する LVM についての詳しい情報は、http://www.tldp.org/HOWTO/LVM-HOWTO/index.html を参照して下さい。

次に、**rhgb** は、**rhgb** パッケージがインストール済みで、グラフィカルブートがサポートされていることを示します。但し、これは、デフォルトのランレベルが **id:5:initdefault:** に設定されていることが **/etc/inittab** に記載されていることを前提とします。

最後に、**quiet** は、すべての詳細なカーネルメッセージがブート時に抑制されていることを示しています。

E.2.3. /proc/cpuinfo

この仮想ファイルは、システムが使用するプロセッサーのタイプを特定します。**/proc/cpuinfo** の標準的な出力の例は以下の通りです:

```
processor : 0
```

```
vendor_id : GenuineIntel
cpu family : 15
model  : 2
model name : Intel(R) Xeon(TM) CPU 2.40GHz
stepping : 7 cpu
MHz  : 2392.371
cache size : 512 KB
physical id : 0
siblings : 2
runqueue : 0
fdiv_bug : no
hlt_bug  : no
f00f_bug : no
coma_bug : no
fpu  : yes
fpu_exception : yes
cpuid level : 2
wp  : yes
flags  : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca  cmov pat
pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm
bogomips : 4771.02
```

- **processor** — 各プロセッサーが識別番号で表示されます。プロセッサーを 1 基搭載しているシステムでは、**0** のみが表示されます。
- **cpu family** — システム内のプロセッサーを正式に識別します。Intel ベースのシステムの場合は、"86" の前に番号を配置し、値を特定します。これは、586、486、386 などの旧システムのアーキテクチャーを識別する場合に特に有用です。一部の RPM パッケージはこれらの特定のアーキテクチャー別にコンパイルされているので、この値は、ユーザーがインストールすべきパッケージを判断する際にも役立ちます。
- **model name** — プロセッサーの一般名がプロジェクト名と共に表示されます。
- **cpu MHz** — プロセッサーの正確な速度がメガヘルツ単位で小数点以下第 3 位まで表示されます。
- **cache size** — プロセッサーが使用可能な 2 次キャッシュメモリの容量が表示されます。
- **siblings** — ハイパースレッディングを採用したアーキテクチャーで同じ物理 CPU 上にある sibling CPU の数を表示します。
- **flags** — Floating Point Unit (FPU) の有無や MMX 指示の処理能力など、プロセッサーに関する数多くの異なる性質を定義します。

E.2.4. /proc/crypto

このファイルは、Linux カーネルが使用するインストール済みの全暗号化方式 (それぞれの追加詳細情報を含む) を一覧表示します。**/proc/crypto** のファイルのサンプルは以下のようになります:

```
name         : sha1
module       : kernel
type         : digest
blocksize    : 64
digestsize   : 20
name         : md5
```

```
module       : md5
type         : digest
blocksize    : 64
digestsize   : 16
```

E.2.5. /proc/devices

このファイルは、現在設定済みの様々なキャラクターデバイス及びブロックデバイスを表示します (但し、モジュールがロードされていないデバイスは除く)。このファイルの出力例は以下の通りです:

```
Character devices:
  1 mem
  4 /dev/vc/0
  4 tty
  4 ttyS
  5 /dev/tty
  5 /dev/console
  5 /dev/ptmx
  7 vcs
  10 misc
  13 input
  29 fb
  36 netlink
  128 ptm
  136 pts
  180 usb

Block devices:
  1 ramdisk
  3 ide0
  9 md
  22 ide1
  253 device-mapper
  254 mdp
```

/proc/devices の出力には、デバイスのメジャー番号及び名前が含まれ、**Character devices** (キャラクターデバイス) 及び **Block devices** (ブロックデバイス) の 2 つの主要なセクションに分かれます。

キャラクターデバイス は、*ブロックデバイス* と似ていますが、以下の 2 点が基本的な相違点となります:

1. キャラクターデバイスには、バッファリングは必要ありません。ブロックデバイスではバッファーが使用可能で、要求に対応する前に順序付けをすることができます。情報をデバイスに書き込む前に順序付けすることにより、より効率性の高い順序での配置が可能となるため、この機能は情報を保管するように設計されたデバイス (例: ハードディスク) に重要となります。

2. キャラクターデバイスは、サイズが事前設定されていないデータを送信します。ブロックデバイスは、デバイス別に設定済みの一定サイズのブロック単位で情報を送受信することができます。

デバイスについての詳しい情報は、システムにインストールされている以下のドキュメントを参照して下さい:

```
/usr/share/doc/kernel-doc-<kernel_version>/Documentation/devices.txt
```

E.2.6. /proc/dma

このファイルには、使用中の登録済み ISA DMA チャンネルの一覧が記載されています。**/proc/dma** ファイルのサンプルは以下ようになります:

```
 4: cascade
```

E.2.7. /proc/execdomains

このファイルは、Linux カーネルによって現在サポートされている *実行ドメイン* と、それらがサポートする様々なパーソナリティを一覧表示します。

```
0-0   Linux           [kernel]
```

実行ドメインは、オペレーティングシステムの「パーソナリティ」として考えて下さい。Linux では、Solaris、UnixWare、FreeBSD など、その他のバイナリ形式を使用することができるため、プログラマーは、タスクのパーソナリティを変更することにより、オペレーティングシステムがこれらのバイナリからのシステムコールを扱う方法を変更することができます。**PER_LINUX** 実行ドメイン以外では、異なるパーソナリティを動的ロード可能モジュールとして実装することができます。

E.2.8. /proc/fb

このファイルには、フレームバッファデバイスとフレームバッファデバイス番号、ならびにそれを制御するドライバーの一覧が記載されています。フレームバッファデバイスを搭載したシステムの **/proc/fb** の標準的な出力は以下の通りです:

```
0 VESA VGA
```

E.2.9. /proc/filesystems

このファイルは、カーネルが現在サポートしているファイルシステムタイプの一覧を表示します。一般的な **/proc/filesystems** ファイルの出力例は以下の通りです:

```
nodev   sysfs
nodev   rootfs
nodev   bdev
nodev   proc
nodev   sockfs
nodev   binfmt_misc
nodev   usbfs
nodev   usbdevfs
nodev   futexfs
nodev   tmpfs
nodev   pipefs
nodev   eventpollfs
nodev   devpts
 ext2
nodev   ramfs
nodev   hugetlbfs
 iso9660
nodev   mqueue
```

```
  ext3
nodev   rpc_pipefs
nodev   autofs
```

第 1 のカラムは、ファイルシステムがブロックデバイス上にマウントされているかどうかを示しています。**nodev** で始まる場合は、デバイスにはマウントされていません。第 2 のカラムには、サポートされているファイルシステムの名前が表示されます。

mount コマンドは、ファイルシステムが引数として指定されていない場合に、このファイルに記載されたファイルシステムを循環します。

E.2.10. /proc/interrupts

このファイルには、x86 アーキテクチャー上における、IRQ ごとの割り込み回数が記録されます。標準的な **/proc/interrupts** は、以下の通りです:

```
   CPU0
   0:   80448940          XT-PIC  timer
   1:     174412          XT-PIC  keyboard
   2:          0          XT-PIC  cascade
   8:          1          XT-PIC  rtc
  10:     410964          XT-PIC  eth0
  12:      60330          XT-PIC  PS/2 Mouse
  14:    1314121          XT-PIC  ide0
  15:    5195422          XT-PIC  ide1
NMI:          0
ERR:          0
```

マルチプロセッサーマシンの場合には、このファイルは若干異なります。

```
     CPU0       CPU1
   0: 1366814704          0          XT-PIC  timer
   1:        128        340    IO-APIC-edge  keyboard
   2:          0          0          XT-PIC  cascade
   8:          0          1    IO-APIC-edge  rtc
  12:       5323       5793    IO-APIC-edge  PS/2 Mouse
  13:          1          0          XT-PIC  fpu
  16:   11184294   15940594   IO-APIC-level  Intel EtherExpress Pro 10/100
Ethernet
  20:    8450043   11120093   IO-APIC-level  megaraid
  30:      10432      10722   IO-APIC-level  aic7xxx
  31:         23         22   IO-APIC-level  aic7xxx
NMI:          0
ERR:          0
```

第 1 のカラムには、IRQ 番号が表示されます。システム内の CPU 別にカラムが分かれており、各 CPU の IRQ ごとの割り込み回数が表示されます。次のカラムには割り込みのタイプ、最後のカラムには、その IRQ にあるデバイス名が表示されます。

このファイルに表示されている、アーキテクチャー固有の割り込みの各タイプは、それぞれ意味が異なります。x86 マシンの場合には、以下の値が一般的です:

- **XT-PIC** — これは、旧 AT コンピューターの割り込みです。
- **IO-APIC-edge** — この割り込みの電圧信号は低から高に推移し、**エッジ** が作成されます。こ

こで、割り込みが発生し、1 回だけ信号が送られます。この種の割り込みならびに **IO-APIC-level** 割り込みは、586 ファミリー以降のプロセッサーを搭載したシステムのみで発生します。

- **IO-APIC-level** — 電圧信号が高い時に、再度低くなるまで割り込みを生成します。

E.2.11. /proc/iomem

このファイルには、物理デバイス別のシステムメモリの現在のマップが表示されます:

```
00000000-0009fbff : System RAM
0009fc00-0009ffff : reserved
000a0000-000bffff : Video RAM area
000c0000-000c7fff : Video ROM
000f0000-000fffff : System ROM
00100000-07ffffff : System RAM
00100000-00291ba8 : Kernel code
00291ba9-002e09cb : Kernel data
e0000000-e3ffffff : VIA Technologies, Inc. VT82C597 [Apollo VP3] e4000000-
e7ffffff : PCI Bus #01
e4000000-e4003fff : Matrox Graphics, Inc. MGA G200 AGP
e5000000-e57fffff : Matrox Graphics, Inc. MGA G200 AGP
e8000000-e8ffffff : PCI Bus #01
e8000000-e8ffffff : Matrox Graphics, Inc. MGA G200 AGP
ea000000-ea00007f : Digital Equipment Corporation DECchip 21140
[FasterNet]
ea000000-ea00007f : tulip ffff0000-ffffffff : reserved
```

第 1 のカラムには、異なる各メモリタイプで使用するメモリレジスターが表示されます。第 2 のカラムには、それらのレジスター内にあるメモリの種類が示され、システム RAM 内のカーネルで使用されているメモリレジスターが表示されます。又、ネットワークインターフェースカードに複数のイーサネットポートがある場合には、各ポートに割り当てられたメモリレジスターが表示されます。

E.2.12. /proc/ioports

/proc/ioports の出力には、デバイスとの入出力通信に使用される、現在登録済みのポートリージョンの一覧が表示されます。このファイルは長くなる場合があります。以下は出力の一部です:

```
0000-001f : dma1
0020-003f : pic1
0040-005f : timer
0060-006f : keyboard
0070-007f : rtc
0080-008f : dma page reg
00a0-00bf : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
02f8-02ff : serial(auto)
0376-0376 : ide1
03c0-03df : vga+
03f6-03f6 : ide0
03f8-03ff : serial(auto)
0cf8-0cff : PCI conf1
```

```
d000-dfff : PCI Bus #01
e000-e00f : VIA Technologies, Inc. Bus Master IDE
e000-e007 : ide0
e008-e00f : ide1
e800-e87f : Digital Equipment Corporation DECchip 21140 [FasterNet]
e800-e87f : tulip
```

第 1 のカラムは、第 2 のカラムに記載されているデバイス用に確保されている I/O ポートアドレスの範囲を示しています。

E.2.13. /proc/kcore

このファイルは、システムの物理メモリを示し、コアファイル形式で保管されます。大半の **/proc/** ファイルとは異なり、**kcore** ではサイズが表示されます。この値はバイト単位で示され、使用中の物理メモリ (RAM) の容量に 4 KB を加算した値となります。

このファイルの内容は、**gdb** などのデバッガーによって検査されるように設計されており、人間には読み取れません。

警告

/proc/kcore 仮想ファイルは表示しないで下さい。このファイルの内容は、ターミナル上のテキスト出力を混乱させてしまいます。このファイルを誤って表示してしまった場合には、**Ctrl+C** を押してプロセスを停止した上で **reset** と入力し、コマンドラインプロンプトを元に戻します。

E.2.14. /proc/kmsg

このファイルは、カーネルが生成したメッセージを保持するのに使用されます。これらのメッセージは、**/sbin/klogd** や **/bin/dmesg** などの別のプログラムによってピックアップされます。

E.2.15. /proc/loadavg

このファイルは、CPU 及び IO の経時的な負荷平均ならびに **uptime** やその他のコマンドが使用する追加のデータを表示します。**/proc/loadavg** ファイルのサンプルは以下のようになります:

```
0.20 0.18 0.12 1/80 11206
```

最初の 3 つのカラムは、過去 1 分間、5 分間、15 分間の CPU 及び IO の使用率の測定値を示しています。第 4 のカラムは、現在実行中のプロセス数とプロセスの合計数、最後のカラムは、最後に使用されたプロセスの ID を示しています。

又、負荷平均は、実行する準備の整った (実行キューにあり、CPU 共有に向けて待機をしている) プロセスの数も示します。

E.2.16. /proc/locks

このファイルは、カーネルによって現在ロックされているファイルを表示します。このファイルの内容には、内部カーネルデバッグデータが含まれ、システムの使用法によって大幅に異なります。負荷の軽いシステムの **/proc/locks** ファイルのサンプルは以下のようになります:

```
1: POSIX  ADVISORY  WRITE 3568 fd:00:2531452 0 EOF
2: FLOCK  ADVISORY  WRITE 3517 fd:00:2531448 0 EOF
3: POSIX  ADVISORY  WRITE 3452 fd:00:2531442 0 EOF
4: POSIX  ADVISORY  WRITE 3443 fd:00:2531440 0 EOF
5: POSIX  ADVISORY  WRITE 3326 fd:00:2531430 0 EOF
6: POSIX  ADVISORY  WRITE 3175 fd:00:2531425 0 EOF
7: POSIX  ADVISORY  WRITE 3056 fd:00:2548663 0 EOF
```

ロックごとに行が分かれており、各行の先頭には一意の番号が付いています。第 2 のカラムは、使用しているロックのクラスを示し、**FLOCK** は **flock** システムコールからの旧式の UNIX ファイルロックを、 **POSIX** は **lockf** システムコールからのより新しい POSIX ロックを表しています。

第 3 のカラムは、**ADVISORY** または **MANDATORY** の 2 つの値が入ります。**ADVISORY** は、そのロックによって他のユーザーがデータにアクセスするのを防止するのではなく、他のユーザーがファイルをロックすることのみを防ぐことを意味します。**MANDATORY** は、ロックが保持されている間は、他のユーザーによるデータへのアクセスは一切許可されないことを意味します。第 4 のカラムには、ロックを保持するプロセスに、ファイルへの **READ** (読み取り) または **WRITE** (書き込み) アクセスが許可されているかどうかを示しています。第 5 のカラムは、ロックを保持するプロセスの ID です。第 6 のカラムは、ロックされているファイルの ID を ***MAJOR-DEVICE*:*MINOR-DEVICE*:*INODE-NUMBER*** の形式で示しています。第 7 と第 8 のカラムは、ロックされている領域の開始及び終了位置を示します。

E.2.17. /proc/mdstat

このファイルには、マルチディスクの RAID 構成についての最新の情報が含まれます。そのような構成がシステムには含まれていない場合には、**/proc/mdstat** の出力は以下のようになります:

```
Personalities :  read_ahead not set unused devices: <none>
```

このファイルは、ソフトウェア RAID または **md** デバイスがない場合には、上記に示した状態で維持されます。その場合は、 **/proc/mdstat** を表示して **md*X*** RAID デバイスの現在の状況を確認して下さい。

以下の **/proc/mdstat** ファイルには、**md0** が RAID 1 デバイスとして設定されたシステムで、現在ディスクを再同期中であることが表示されています:

```
Personalities : [linear] [raid1] read_ahead 1024 sectors
md0: active raid1 sda2[1] sdb2[0] 9940 blocks [2/2] [UU] resync=1%
finish=12.3min algorithm 2 [3/3] [UUU]
unused devices: <none>
```

E.2.18. /proc/meminfo

これは、システムの RAM の使用状況に関する重要な情報を大量にレポートするため、**/proc/** ディレクトリでより一般的に使用されるファイルの一つです

256 MB の RAM を搭載し、512 MB の swap 領域を持つシステムの **/proc/meminfo** 仮想ファイルのサンプルは以下のようになります:

```
MemTotal:       255908 kB
```

```
MemFree:         69936 kB
Buffers:         15812 kB
Cached:         115124 kB
SwapCached:          0 kB
Active:          92700 kB
Inactive:        63792 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:       255908 kB
LowFree:         69936 kB
SwapTotal:      524280 kB
SwapFree:       524280 kB
Dirty:               4 kB
Writeback:           0 kB
Mapped:          42236 kB
Slab:            25912 kB
Committed_AS:   118680 kB
PageTables:       1236 kB
VmallocTotal:  3874808 kB
VmallocUsed:      1416 kB
VmallocChunk:  3872908 kB
HugePages_Total:     0
HugePages_Free:      0
Hugepagesize:     4096 kB
```

これらの情報の大半は、**free**、**top**、及び **ps** のコマンドによって使用されます。実際、**free** コマンドの出力は、**/proc/meminfo** の内容及び構造と似ていますが、**/proc/meminfo** を直接確認すると、より詳しい情報が明らかになります:

- **MemTotal** — 物理 RAM 容量の合計 (キロバイト単位)。
- **MemFree** — システムが未使用の物理 RAM の容量 (キロバイト単位)。
- **Buffers** — ファイルバッファーに使用されている物理 RAM の容量 (キロバイト単位)。
- **Cached** — キャッシュメモリとして使用されている物理 RAM の容量 (キロバイト単位)。
- **SwapCached** — キャッシュメモリとして使用されている swap の容量 (キロバイト単位)。
- **Active** — アクティブに使用中のバッファーまたはページキャッシュ用メモリ容量の合計 (キロバイト単位)。これは、最近使用されたメモリで、通常、他の目的では再構築されません。
- **Inactive** — バッファーまたはページキャッシュ用の使用可能な空きメモリ容量の合計 (キロバイト単位)。これは、最近は使用されていないメモリで、他の目的で再構築することができます。
- **HighTotal** 及び **HighFree** — カーネル領域に直接マッピングされていない空きメモリ容量の合計 (キロバイト単位)。**HighTotal** 値は使用しているカーネルのタイプによって異なる場合があります。
- **LowTotal** 及び **LowFree** — カーネル領域に直接マッピングされている空きメモリ容量の合計 (キロバイト単位)。**LowTotal** 値は使用しているカーネルのタイプによって異なる場合があります。
- **SwapTotal** — 使用可能な swap の容量の合計 (キロバイト単位)。

- **SwapFree** — swap の空き容量の合計 (キロバイト単位)。
- **Dirty** — ディスクに書き戻しされるのを待っているメモリの合計容量 (キロバイト単位)。
- **Writeback** — ディスクに書き戻しされているアクティブなメモリの合計容量 (キロバイト単位)。
- **Mapped** — **mmap** コマンドを使用した、デバイス、ファイル、またはライブラリのマッピングに使用されているメモリの合計容量 (キロバイト単位)。
- **Slab** — カーネルが自ら使用する目的でデータ構造をキャッシュするのに使用しているメモリの合計容量 (キロバイト単位)。
- **Committed_AS** — ワークロードの完了が予想されるメモリの合計容量 (キロバイト単位)。この値は、最悪のシナリオの値を示しており、swap メモリも含みます。
- **PageTables** — 最も低いページテーブルレベル専用のメモリの合計容量 (キロバイト単位)。
- **VMallocTotal** — 割り当て済みの総仮想アドレス空間のメモリ容量の合計 (キロバイト単位)。
- **VMallocUsed** — 使用済み仮想アドレス空間のメモリ容量の合計 (キロバイト単位)。
- **VMallocChunk** — 使用可能な仮想アドレス空間の最大の連続メモリブロック (キロバイト単位)。
- **HugePages_Total** — システムのヒュージページの合計数。この値は、**Hugepagesize** を **/proc/sys/vm/hugetlb_pool** で指定されているヒュージページ用に確保されたメガバイトの値で除算することによって算出されます。**この統計は、x86、Itanium、及び AMD64 のアーキテクチャーのみで表示されます**
- **HugePages_Free** — システムで使用可能なヒュージページの合計数。**この統計は、x86、Itanium、及び AMD64 のアーキテクチャーのみで表示されます。**
- **Hugepagesize** — 各ヒュージページユニットのサイズ (キロバイト単位)。32 ビットアーキテクチャー用のユニプロセッサーカーネルの場合、この値のデフォルトは 4096 KB です。SMP、hugemem カーネル、及び AMD64 の場合は、デフォルトは 2048 KB です。又、Itanium アーキテクチャーの場合は、デフォルトは 262144 KB です。**この統計は、x86、Itanium、及び AMD64 のアーキテクチャーのみで表示されます。**

E.2.19. /proc/misc

このファイルは、その他のメジャーデバイス (デバイス番号 10) 上に登録されているその他のドライバーを一覧表示します:

```
63 device-mapper 175 agpgart 135 rtc 134 apm_bios
```

第 1 のカラムは、各デバイスのマイナー番号、第 2 のカラムは使用中のドライバーを示します。

E.2.20. /proc/modules

このファイルには、カーネルにロードされる全モジュールの一覧が表示されます。内容は設定とシステムの使用方法によって異なりますが、以下の **/proc/modules** ファイルの出力例と同じように構成されるはずです:

注記

この例は、読みやすい形式に再構成されています。この情報の大半は、**/sbin/lsmod** コマンドでも表示することができます。

```
nfs      170109  0 -          Live 0x129b0000
lockd    51593   1 nfs,       Live 0x128b0000
nls_utf8 1729    0 -          Live 0x12830000
vfat     12097   0 -          Live 0x12823000
fat      38881   1 vfat,      Live 0x1287b000
autofs4  20293   2 -          Live 0x1284f000
sunrpc   140453  3 nfs,lockd, Live 0x12954000
3c59x    33257   0 -          Live 0x12871000
uhci_hcd 28377   0 -          Live 0x12869000
md5      3777    1 -          Live 0x1282c000
ipv6     211845 16 -          Live 0x128de000
ext3     92585   2 -          Live 0x12886000
jbd      65625   1 ext3,      Live 0x12857000
dm_mod   46677   3 -          Live 0x12833000
```

第 1 のカラムには、モジュール名が表示されます。

第 2 のカラムには、モジュールのメモリ容量がバイト単位で表示されます。

第 3 のカラムには、現在ロードされているモジュールのインスタンス数が表示されます。ゼロの値は、アンロードされているモジュールを示します。

第 4 のカラムには、そのモジュールが機能するために依存する別のモジュールの有無と、それらの別のモジュールが表示されます。

第 5 のカラムには、ロード状態が表示されます: 取り得る値は、**Live**、**Loading**、**Unloading** のいずれかのみです。

第 6 のカラムには、ロードされているモジュールの現在のカーネルメモリオフセットが表示されます。この情報は、デバッグや **oprofile** などのプロファイリングツールに役立ちます。

E.2.21. /proc/mounts

このファイルは、システムが使用中の全マウントの一覧を提供します。

```
rootfs / rootfs rw 0 0
/proc /proc proc rw,nodiratime 0 0 none
/dev ramfs rw 0 0
/dev/mapper/VolGroup00-LogVol00 / ext3 rw 0 0
none /dev ramfs rw 0 0
/proc /proc proc rw,nodiratime 0 0
/sys /sys sysfs rw 0 0
none /dev/pts devpts rw 0 0
usbdevfs /proc/bus/usb usbdevfs rw 0 0
/dev/hda1 /boot ext3 rw 0 0
none /dev/shm tmpfs rw 0 0
none /proc/sys/fs/binfmt_misc binfmt_misc rw 0 0
sunrpc /var/lib/nfs/rpc_pipefs rpc_pipefs rw 0 0
```

上記の出力は、**/etc/mtab** の内容と似ていますが、**/proc/mounts** の方がより最新である点が異なります。

第 1 のカラムにはマウントされているデバイス、 第 2 のカラムにはマウントポイント、第 3 のカラムにはファイルシステムのタイプ、第 4 のカラムには読み取り専用 (**ro**) もしくは 読み取り/書き込み (**rw**) でマウントされているかが表示されます。第 5 及び第 6 のカラムには、**/etc/mtab** で使用されている形式に適合するように設計されたダミー値が表示されます。

E.2.22. /proc/mtrr

このファイルには、システムで現在使用中の Memory Type Range Register (MTRR) が記載されています。システムアーキテクチャーが MTRR をサポートしている場合には、**/proc/mtrr** ファイルは以下のように表示されます:

```
reg00: base=0x00000000 (   0MB), size= 256MB: write-back, count=1
reg01: base=0xe8000000 (3712MB), size=  32MB: write-combining, count=1
```

MTRR は、Intel P6 ファミリーのプロセッサー (Pentium II 以降) で使用され、メモリ範囲にアクセスするプロセッサーを制御します。 PCI または AGP バス上でビデオカードを使用する場合、**/proc/mtrr** ファイルを適切に設定していると、パフォーマンスを 150% 以上向上させることができます。

ほとんどの場合、この値はデフォルトで適切に設定されています。このファイルの手動設定に関する詳しい情報は、以下の場所にあるドキュメントを参照して下さい:

```
/usr/share/doc/kernel-doc-<kernel_version>/Documentation/<arch>/mtrr.txt
```

E.2.23. /proc/partitions

このファイルには、パーティションブロックの割り当て情報が記載されています。ベーシックなシステムの場合、このファイルの出力例は以下のようになります:

```
major minor  #blocks  name
  3     0   19531250 hda
  3     1     104391 hda1
  3     2   19422585 hda2
253     0   22708224 dm-0
253     1     524288 dm-1
```

上記の情報の大半は、ユーザーにとってはあまり重要ではありませんが、以下のカラムは例外です:

- **major** — このパーティションを使用するデバイスのメジャー番号。**/proc/partitions** の (**3**) は、**/proc/devices** のブロックデバイス **ide0** と一致します。
- **minor** — このパーティションを使用するデバイスのマイナー番号。これは、パーティションを異なる物理デバイスに分けるのに役立ち、そのパーティション名の末尾についた番号に関連付けします。
- **#blocks** — 特定のパーティションに含まれている物理ディスクブロックの番号を表示します。
- **name** — パーティション名。

E.2.24. /proc/slabinfo

このファイルは、*スラブレベル*におけるメモリ使用状況に関する詳しい情報を提供します。バージョン 2.2 以降の Linux カーネルは、*スラブプール* を使用してページレベル以上のメモリを管理します。一般的に使用されるオブジェクトには独自のスラブプールがあります。

詳細度の高い **/proc/slabinfo** ファイルを手動で解析する代わりに **/usr/bin/slabtop** プログラムを使用して、カーネルスラブキャッシュの情報をリアルタイムで表示することができます。 このプログラムは、カラムの並べ替えや画面のリフレッシュなどのカスタム設定が可能です。

/usr/bin/slabtop のサンプルスクリーンショットは、通常以下の例ようになります:

```
Active / Total Objects (% used)    : 133629 / 147300 (90.7%)
Active / Total Slabs (% used)      : 11492 / 11493 (100.0%)
Active / Total Caches (% used)     : 77 / 121 (63.6%)
Active / Total Size (% used)       : 41739.83K / 44081.89K (94.7%)
Minimum / Average / Maximum Object : 0.01K / 0.30K / 128.00K
OBJS   ACTIVE USE      OBJ   SIZE     SLABS OBJ/SLAB CACHE SIZE NAME
44814  43159  96%    0.62K   7469      6     29876K ext3_inode_cache
36900  34614  93%    0.05K    492     75      1968K buffer_head
35213  33124  94%    0.16K   1531     23      6124K dentry_cache
7364   6463  87%    0.27K    526     14      2104K radix_tree_node
2585   1781  68%    0.08K     55     47       220K vm_area_struct
2263   2116  93%    0.12K     73     31       292K size-128
1904   1125  59%    0.03K     16    119        64K size-32
1666    768  46%    0.03K     14    119        56K anon_vma
1512   1482  98%    0.44K    168      9       672K inode_cache
1464   1040  71%    0.06K     24     61        96K size-64
1320    820  62%    0.19K     66     20       264K filp
678    587  86%    0.02K      3    226        12K dm_io
678    587  86%    0.02K      3    226        12K dm_tio
576    574  99%    0.47K     72      8       288K proc_inode_cache
528    514  97%    0.50K     66      8       264K size-512
492    372  75%    0.09K     12     41        48K bio
465    314  67%    0.25K     31     15       124K size-256
452    331  73%    0.02K      2    226         8K biovec-1
420    420 100%    0.19K     21     20        84K skbuff_head_cache
305    256  83%    0.06K      5     61        20K biovec-4
290      4   1%    0.01K      1    290         4K revoke_table
264    264 100%    4.00K    264      1      1056K size-4096
260    256  98%    0.19K     13     20        52K biovec-16
260    256  98%    0.75K     52      5       208K biovec-64
```

/usr/bin/slabtop に含まれる、より一般的に使用されている **/proc/slabinfo** 内の統計には以下が含まれます:

- **OBJS** — 使用中 (割り当て済み) 及び使用していない予備を含むオブジェクトの合計数 (メモリブロック)。
- **ACTIVE** — 使用中 (割り当て済み) のオブジェクト (メモリブロック) 数。
- **USE** — アクティブな全オブジェクトの割合 ((ACTIVE/OBJS)(100))。
- **OBJ SIZE** — オブジェクトのサイズ。
- **SLABS** — スラブの合計数。
- **OBJ/SLAB** — 1 つのスラブに収まるオブジェクトの数。

- **CACHE SIZE** — スラブのキャッシュサイズ。
- **NAME** — スラブの名前。

/usr/bin/slabtop プログラムに関する詳しい情報は、**slabtop** の man ページを参照して下さい。

E.2.25. /proc/stat

このファイルには、前回の再起動以降のシステムに関する様々な異なる統計が記録されます。**/proc/stat** の内容は、長くなる場合があり、通常は以下の例のような内容で始まります:

```
cpu  259246 7001 60190 34250993 137517 772 0
cpu0 259246 7001 60190 34250993 137517 772 0
intr 354133732 347209999 2272 0 4 4 0 0 3 1 1249247 0 0 80143 0 422626
5169433
ctxt 12547729
btime 1093631447
processes 130523
procs_running 1
procs_blocked 0
preempt 5651840
cpu  209841 1554 21720 118519346 72939 154 27168
cpu0 42536 798 4841 14790880 14778 124 3117
cpu1 24184 569 3875 14794524 30209 29 3130
cpu2 28616 11 2182 14818198 4020 1 3493
cpu3 35350 6 2942 14811519 3045 0 3659
cpu4 18209 135 2263 14820076 12465 0 3373
cpu5 20795 35 1866 14825701 4508 0 3615
cpu6 21607 0 2201 14827053 2325 0 3334
cpu7 18544 0 1550 14831395 1589 0 3447
intr 15239682 14857833 6 0 6 6 0 5 0 1 0 0 0 29 0 2 0 0 0 0 0 0 0 94982 0
286812
ctxt 4209609
btime 1078711415
processes 21905
procs_running 1
procs_blocked 0
```

より一般的に使用される統計には、以下が含まれます:

- **cpu** — システムがユーザーモード、優先度の低い (nice) ユーザーモード、システムモード、アイドルタスク、I/O 待ち、IRQ (hardirq)、及び softirq に入っていた *jiffies* (x86 システムでは 1/100 秒) 数をそれぞれ測定します。IRQ (hardirq) はハードウェアイベントに対する直接応答です。IRQ は最小限の作業に対応し、「負担の重い」作業は softirq が実行するように待ち行列を作成します。softirq は、IRQ よりも低い優先度で実行されるので、割り込み頻度がより高くなる可能性があります。全 CPU の合計は最上部に表示され、その下に個別の CPU がそれぞれの統計とともに表示されます。以下の例は、マルチスレッディング対応の 4-way Intel Pentium Xeon の設定なので、4 基の物理プロセッサーと 4 基の仮想プロセッサーを合わせて合計 8 基のプロセッサーが表示されています。
- **page** — システムがディスクに書き込み/書き出ししているメモリページの数。
- **swap** — システムが取り込み/取り出ししている swap ページの数。
- **intr** — システムで発生した割り込み数。

- **btime** — 1970 年 1 月 1 日からの秒数で示された起動時間 (別称: *エポック*)。

E.2.26. /proc/swaps

このファイルは、swap 領域と使用率を測定します。swap パーティションが 1 つのみのシステムの場合には、**/proc/swaps** の出力は以下のようになります:

```
Filename                        Type        Size    Used    Priority
/dev/mapper/VolGroup00-LogVol01 partition   524280  0       -1
```

この情報の一部は **/proc/** ディレクトリ内のその他のファイルにも記載されていますが、**/proc/swap** は、各 swap ファイル名、swap 領域のタイプ、合計サイズ、使用中の容量 (キロバイト単位) のスナップショットを提供します。優先度のカラムは、複数の swap ファイルを使用している場合に役立ちます。swap ファイルは、優先度が低いほど、使用される可能性が高くなります。

E.2.27. /proc/sysrq-trigger

echo コマンドを使用して、このファイルに書き込むことにより、リモートの root ユーザーは、大半のシステム要求キーコマンドを、ローカルのターミナルで実行しているかのように、リモートで実行することができます。このファイルに値を **echo** するには、**/proc/sys/kernel/sysrq** を **0** 以外の値に設定する必要があります。システム要求キーに関する詳しい情報は、「/proc/sys/kernel/」 を参照して下さい。

このファイルは、書き込み可能ですが、root ユーザーでさえも読み取ることはできません。

E.2.28. /proc/uptime

このファイルには、前回の再起動から経過した時間に関する詳細情報が記載されています。**/proc/uptime** の出力は極めて少なくなります:

```
350735.47 234388.90
```

第 1 の値は、システムが起動してから経過した合計時間を示しています。 第 2 の値は、各コアがアイドル状態で経過した合計時間の秒数です。このため、マルチコア搭載のシステムでは、第 2 の値がシステムアップタイムの合計よりも大きくなる場合があります。

E.2.29. /proc/version

このファイルは、Linux カーネルのバージョン、カーネルのコンパイルに使用した **gcc**、及びカーネルのコンパイルの時刻を示します。又、カーネルコンパイラーのユーザー名も (括弧内に) 表示されます。

```
Linux version 2.6.8-1.523 (user@foo.redhat.com) (gcc version 3.4.1
20040714 \  (Red Hat Enterprise Linux 3.4.1-7)) #1 Mon Aug 16 13:27:03 EDT
2004
```

この情報は、ユーザーのログイン時に表示されるバージョンデータをはじめとする様々な目的で使用されます。

E.3. /PROC/ 内のディレクトリ

カーネルに関する情報の共通グループは、**/proc/** ディレクトリ内のディレクトリ及びサブディレクトリにグループ化されます。

E.3.1. プロセスディレクトリ

各 **/proc/** ディレクトリには、数値名が付いた数多くのディレクトリが格納されています。この一覧は以下のような内容です:

```
dr-xr-xr-x    3 root     root            0 Feb 13 01:28 1
dr-xr-xr-x    3 root     root            0 Feb 13 01:28 1010
dr-xr-xr-x    3 xfs      xfs             0 Feb 13 01:28 1087
dr-xr-xr-x    3 daemon   daemon          0 Feb 13 01:28 1123
dr-xr-xr-x    3 root     root            0 Feb 13 01:28 11307
dr-xr-xr-x    3 apache   apache          0 Feb 13 01:28 13660
dr-xr-xr-x    3 rpc      rpc             0 Feb 13 01:28 637
dr-xr-xr-x    3 rpcuser  rpcuser         0 Feb 13 01:28 666
```

これらのディレクトリは、プログラムのプロセス ID にちなんで名付けられ、そのプロセス固有の情報が含まれているため、*プロセスディレクトリ* と呼ばれています。各プロセスディレクトリのオーナーとグループは、そのプロセスを実行しているユーザーに設定されます。プロセスが終了すると、そのプロセスの **/proc/** プロセスディレクトリはなくなります。

各プロセスディレクトリには以下のファイルが格納されています:

- **cmdline** — プロセスを起動した際に発行されたコマンドが格納されています。
- **cwd** — そのプロセスの現在の作業ディレクトリへのシンボリックリンク。
- **environ** — そのプロセスの環境変数の一覧。環境変数はすべて大文字で、値は小文字で記載されます。
- **exe** — そのプロセスの実行可能ファイルへのシンボリックリンク。
- **fd** — 特定のプロセスの全ファイル記述子を格納するディレクトリ。これらは、番号付きリンクの形式で表示されます:

  ```
  total 0
  lrwx------    1 root     root           64 May  8 11:31 0 ->
  /dev/null
  lrwx------    1 root     root           64 May  8 11:31 1 ->
  /dev/null
  lrwx------    1 root     root           64 May  8 11:31 2 ->
  /dev/null
  lrwx------    1 root     root           64 May  8 11:31 3 ->
  /dev/ptmx
  lrwx------    1 root     root           64 May  8 11:31 4 ->
  socket:[7774817]
  lrwx------    1 root     root           64 May  8 11:31 5 ->
  /dev/ptmx
  lrwx------    1 root     root           64 May  8 11:31 6 ->
  socket:[7774829]
  lrwx------    1 root     root           64 May  8 11:31 7 ->
  /dev/ptmx
  ```

- **maps** — そのプロセスに関連した様々な実行可能ファイル及びライブラリファイルへのメモリマップの一覧。このファイルは、プロセスの複雑度によっては、かなり長くなる場合があります。**sshd** プロセスからの出力例は以下のような内容で始まります:

```
08048000-08086000 r-xp 00000000 03:03 391479     /usr/sbin/sshd
08086000-08088000 rw-p 0003e000 03:03 391479 /usr/sbin/sshd
08088000-08095000 rwxp 00000000 00:00 0
40000000-40013000 r-xp 0000000 03:03 293205 /lib/ld-2.2.5.so
40013000-40014000 rw-p 00013000 03:03 293205 /lib/ld-2.2.5.so
40031000-40038000 r-xp 00000000 03:03 293282 /lib/libpam.so.0.75
40038000-40039000 rw-p 00006000 03:03 293282 /lib/libpam.so.0.75
40039000-4003a000 rw-p 00000000 00:00 0
4003a000-4003c000 r-xp 00000000 03:03 293218 /lib/libdl-2.2.5.so
4003c000-4003d000 rw-p 00001000 03:03 293218 /lib/libdl-2.2.5.so
```

- **mem** — プロセスによって確保されているメモリ。このファイルは、ユーザーには読み取れません。
- **root** — そのプロセスの root ディレクトリへのリンク。
- **stat** — プロセスの状態。
- **statm** — プロセスが使用中のメモリの状態。**/proc/statm** ファイルのサンプルは以下のようになります:

  ```
  263 210 210 5 0 205 0
  ```

  7 つのカラムは、そのプロセスの異なるメモリ統計に関連しています。これらのカラムは、左から右の順に、以下のような使用メモリに関する情報をレポートしています:

  1. プログラムの合計サイズ (キロバイト単位)
  2. メモリポーションのサイズ (キロバイト単位)
  3. 共有されているページ数
  4. コードのページ数
  5. データ/スタックのページ数
  6. ライブラリページの数
  7. ダーティページの数

- **status** — **stat** や **statm** よりも読みやすい形式で記載されたプロセスの状態。**sshd** の出力例は以下の通りです:

  ```
  Name: sshd
  State: S (sleeping)
  Tgid: 797
  Pid: 797
  PPid: 1
  TracerPid: 0
  Uid: 0 0 0 0
  Gid: 0 0 0 0
  FDSize: 32
  Groups:
  VmSize:     3072 kB
  VmLck:         0 kB
  ```

```
VmRSS:      840 kB
VmData:      104 kB
VmStk:       12 kB
VmExe:      300 kB
VmLib:     2528 kB
SigPnd: 0000000000000000
SigBlk: 0000000000000000
SigIgn: 8000000000001000
SigCgt: 0000000000014005
CapInh: 0000000000000000
CapPrm: 00000000ffffffeff
CapEff: 00000000ffffffeff
```

この出力に記載されている情報には、プロセスの名前と ID、状態 (**S（スリープ中）**、**R（実行中）** など)、プロセスを実行しているユーザー/グループ ID、メモリの使用状況に関する詳細情報などが含まれます。

E.3.1.1. /proc/self/

/proc/self/ ディレクトリは、現在実行中のプロセスへのリンクです。これにより、プロセス ID を知らなくても、プロセスが自ら確認することができます。

シェル環境では、**/proc/self/** ディレクトリを表示すると、そのプロセスのプロセスディレクトリを表示するのと同じ内容が表示されます。

E.3.2. /proc/bus/

このディレクトリには、システム上で使用可能な様々なバスの固有の情報が格納されます。例えば、PCI バス及び USB バスを搭載した標準的なシステムでは、これらのバスに関する現在のデータは、**/proc/bus/** 内のサブディレクトリ内に同じ名前で格納されています (例: **/proc/bus/pci/**)。

/proc/bus/ 内に格納されるサブディレクトリ及びファイルは、システムに接続されているデバイスによって異なります。但し、バスタイプごとに少なくとも 1 つのディレクトリが必要です。これらのバスディレクトリの中には、通常、少なくとも 1 つの数値名 (例: **001**) が付いたサブディレクトリがあり、バイナリファイルが格納されます。

例えば、**/proc/bus/usb/** サブディレクトリには、USB バス上の様々なデバイスをトラッキングするファイルと、それらに必要なドライバーが格納されます。**/proc/bus/usb/** ディレクトリの一覧の例は以下の通りです:

```
total 0 dr-xr-xr-x    1 root     root            0 May  3 16:25 001
-r--r--r--    1 root     root            0 May  3 16:25 devices
-r--r--r--    1 root     root            0 May  3 16:25 drivers
```

The **/proc/bus/usb/001/** ディレクトリには、第 1 の USB バス上にある全デバイス及びマザーボード上の USB ルートハブを識別する **devices** ファイルが格納されます。

/proc/bus/usb/devices ファイルの例は以下の通りです:

```
T:  Bus=01 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=12  MxCh= 2
B:  Alloc=  0/900 us ( 0%), #Int=  0, #Iso=  0
D:  Ver= 1.00 Cls=09(hub  ) Sub=00 Prot=00 MxPS= 8 #Cfgs=  1
P:  Vendor=0000 ProdID=0000 Rev= 0.00
S:  Product=USB UHCI Root Hub
```

```
S:  SerialNumber=d400
C:* #Ifs= 1 Cfg#= 1 Atr=40 MxPwr=  0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub  ) Sub=00 Prot=00 Driver=hub
E:  Ad=81(I) Atr=03(Int.) MxPS=   8 Ivl=255ms
```

E.3.3. /proc/bus/pci

2.6 Linux カーネルの後続バージョンでは、**/proc/pci** ディレクトリは廃止され、**/proc/bus/pci** ディレクトリが採用されました。**cat /proc/bus/pci/devices** のコマンドを使用して、システム上に存在する PCI デバイスの一覧表示することができますが、出力は読みにくく、解釈が困難です。

人間が読める形式で PCI デバイスを一覧表示するには、以下のコマンドを実行して下さい:

```
~]# /sbin/lspci -vb
00:00.0 Host bridge: Intel Corporation 82X38/X48 Express DRAM Controller
        Subsystem: Hewlett-Packard Company Device 1308
        Flags: bus master, fast devsel, latency 0
        Capabilities: [e0] Vendor Specific Information <?>
        Kernel driver in use: x38_edac
        Kernel modules: x38_edac

00:01.0 PCI bridge: Intel Corporation 82X38/X48 Express Host-Primary PCI
Express Bridge (prog-if 00 [Normal decode])
        Flags: bus master, fast devsel, latency 0
        Bus: primary=00, secondary=01, subordinate=01, sec-latency=0
        I/O behind bridge: 00001000-00001fff
        Memory behind bridge: f0000000-f2ffffff
        Capabilities: [88] Subsystem: Hewlett-Packard Company Device 1308
        Capabilities: [80] Power Management version 3
        Capabilities: [90] MSI: Enable+ Count=1/1 Maskable- 64bit-
        Capabilities: [a0] Express Root Port (Slot+), MSI 00
        Capabilities: [100] Virtual Channel <?>
        Capabilities: [140] Root Complex Link <?>
        Kernel driver in use: pcieport
        Kernel modules: shpchp

00:1a.0 USB Controller: Intel Corporation 82801I (ICH9 Family) USB UHCI
Controller #4 (rev 02) (prog-if 00 [UHCI])
        Subsystem: Hewlett-Packard Company Device 1308
        Flags: bus master, medium devsel, latency 0, IRQ 5
        I/O ports at 2100
        Capabilities: [50] PCI Advanced Features
        Kernel driver in use: uhci_hcd
[出力は省略されています]
```

この出力は、カーネルではなく、PCI バス上のカードによって確認される IRQ 番号とアドレスのソートされた一覧です。この一覧は、デバイスの名前とバージョン以外に IRQ の詳細情報も提供するので、管理者は競合を迅速に探し出すことができます。

E.3.4. /proc/driver/

このディレクトリには、カーネルが使用中の特定のドライバーに関する情報が格納されます。

ここにある共通ファイルは **rtc** です。これは、システムの電源が切断されている間に時刻を管理するデバイスである、システムの *Real Time Clock (RTC)* 用のドライバーからの出力を提供します。**/proc/driver/rtc** の出力例は以下のようになります:

```
rtc_time        : 16:21:00
rtc_date        : 2004-08-31
rtc_epoch       : 1900
alarm           : 21:16:27
DST_enable      : no
BCD             : yes
24hr            : yes
square_wave     : no
alarm_IRQ       : no
update_IRQ      : no
periodic_IRQ    : no
periodic_freq   : 1024
batt_status     : okay
```

RTC についての詳しい情報は、システムにインストールされている以下のドキュメントを参照して下さい:

/usr/share/doc/kernel-doc-<*kernel_version*>/Documentation/rtc.txt

E.3.5. /proc/fs

このディレクトリには、どのファイルシステムがエクスポートされているかが表示されます。NFS サーバーを実行している場合は、**cat /proc/fs/nfsd/exports** のコマンドを入力すると、共有されているファイルシステムとそれらのファイルシステムに許可されているパーミッションが表示されます。NFS を使用したファイルシステムの共有に関する更なる情報は、『Storage Administration Guide』 (ストレージ管理ガイド) の 『Network File System (NFS)』 の章を参照して下さい。

E.3.6. /proc/irq/

このディレクトリは、IRQ を CPU アフィニティに設定するのに使用します。これにより、システムは、特定の IRQ を単一の CPU のみに接続することができます。又、特定の CPU が IRQ を一切処理しないようにすることもできます。

各 IRQ には、独自のディレクトリがあり、各 IRQ に個別の設定をすることができます。**/proc/irq/prof_cpu_mask** ファイルは、IRQ ディレクトリ内の **smp_affinity** ファイル用のデフォルト値を格納するビットマスクです。**smp_affinity** 内の値は、特定の IRQ にどの CPU が対応するかを指定します。

/proc/irq/ ディレクトリに関する詳しい情報は、システムにインストールされている以下のドキュメントを参照して下さい:

```
/usr/share/doc/kernel-
doc-kernel_version/Documentation/filesystems/proc.txt
```

E.3.7. /proc/net/

このディレクトリは、様々なネットワークパラメーターと統計の包括的なビューを提供します。このディレクトリ内の各ディレクトリ及び仮想ファイルには、システムのネットワーク設定に関する状況が記載されています。以下は、**/proc/net/** ディレクトリの一覧の一部です:

- **arp** — カーネルの ARP テーブルが記載されています。このファイルは、ハードウェアアドレスをシステム上の IP アドレスに接続するのに特に有用です。
- **atm/** ディレクトリ — このディレクトリ内のファイルには、*Asynchronous Transfer Mode (ATM)* の設定値と統計が格納されています。このディレクトリは、主に ATM ネットワーキング及び ADSL カードに使用されます。
- **dev** — システム上で設定されている様々なネットワークデバイスの一覧を転送/受信の統計と共に表示します。このファイルは、各インターフェースが送信したバイト数、着信/送信パケット数、エラー発生件数、パケットドロップ数などを表示します。
- **dev_mcast** — デバイスがリッスンしている Layer2 マルチキャストグループを一覧表示します。
- **igmp** — このシステムが参加している IP マルチキャストアドレスを一覧表示します。
- **ip_conntrack** — IP 接続を転送しているマシンのトラッキングされているネットワーク接続を一覧表示します。
- **ip_tables_names** — 使用中の **iptables** のタイプを一覧表示します。このファイルは、**iptables** がシステム上でアクティブで、かつ次の値のいずれか 1 つまたは複数が含まれている場合のみに存在します: **filter**、**mangle**、**nat**
- **ip_mr_cache** — マルチキャストルーティングキャッシュを一覧表示します。
- **ip_mr_vif** — マルチキャスト仮想インターフェースを一覧表示します。
- **netstat** — 広範かつ詳細な一連のネットワーク統計が格納されています。これには、TCP タイムアウトや送受信済みの SYN クッキーなどが含まれます。
- **psched** — グローバルパケットスケジューラーのパラメーターを一覧表示します。
- **raw** — RAW デバイスの統計を一覧表示します。
- **route** — カーネルのルーティングテーブルを一覧表示します。
- **rt_cache** — 現在のルーティングキャッシュが格納されています。
- **snmp** — 使用中の様々なネットワークプロトコルの Simple Network Management Protocol (SNMP) データを一覧表示します。
- **sockstat** — ソケットの統計を提供します。
- **tcp** — 詳細な TCP ソケット情報が格納されています。
- **tr_rif** — トークンリング RIF ルーティングテーブルを一覧表示します。
- **udp** — UDP ソケットの詳細情報が格納されています。
- **unix** — 現在使用中の UNIX ドメインソケットを一覧表示します。
- **wireless** — ワイヤレスインターフェースデータを一覧表示します。

E.3.8. /proc/scsi/

このディレクトリのプライマリファイルは **/proc/scsi/scsi** です。このファイルには、認識されているすべての SCSI デバイスの一覧が記載されています。この一覧から、デバイスのタイプやモデル名、ベンダー、SCSI チャンネル、ID データなどを確認することができます。

例えば、システムに SCSI CD-ROM、テープドライブ、ハードディスク、RAID コントローラーが含まれている場合のこのファイルの出力例は以下のようになります:

```
Attached devices:
Host: scsi1
Channel: 00
Id: 05
Lun: 00
Vendor: NEC
Model: CD-ROM DRIVE:466
Rev: 1.06
Type:   CD-ROM
ANSI SCSI revision: 02
Host: scsi1
Channel: 00
Id: 06
Lun: 00
Vendor: ARCHIVE
Model: Python 04106-XXX
Rev: 7350
Type:   Sequential-Access
ANSI SCSI revision: 02
Host: scsi2
Channel: 00
Id: 06
Lun: 00
Vendor: DELL
Model: 1x6 U2W SCSI BP
Rev: 5.35
Type:   Processor
ANSI SCSI revision: 02
Host: scsi2
Channel: 02
Id: 00
Lun: 00
Vendor: MegaRAID
Model: LD0 RAID5 34556R
Rev: 1.01
Type:   Direct-Access
ANSI SCSI revision: 02
```

/proc/scsi/ 内には、システムが使用している各 SCSI ドライバーの独自のディレクトリーがあり、そのドライバーを使用する 各 SCSI コントローラー固有のファイルが格納されます。上記の例では、2 つのドライバーが使用中であるため、**aic7xxx/** 及び **megaraid/** のディレクトリーが存在しています。これらのディレクトリにそれぞれ格納されているファイルには、I/O アドレス範囲、IRQ 情報、及びそのドライバーを使用している SCSI コントローラーの統計が格納されています。コントローラーによって、レポートする情報のタイプ及び量が異なる場合があります。ここに例示したシステムでは、Adaptec AIC-7880 Ultra SCSI ホストアダプターのファイルは以下のように出力されます:

```
Adaptec AIC7xxx driver version: 5.1.20/3.2.4
Compile Options:
```

```
TCQ Enabled By Default : Disabled
AIC7XXX_PROC_STATS     : Enabled
AIC7XXX_RESET_DELAY    : 5
Adapter Configuration:
SCSI Adapter: Adaptec AIC-7880 Ultra SCSI host adapter
Ultra Narrow Controller     PCI MMAPed
I/O Base: 0xfcffe000
Adapter SEEPROM Config: SEEPROM found and used.
Adaptec SCSI BIOS: Enabled
IRQ: 30
SCBs: Active 0, Max Active 1, Allocated 15, HW 16, Page 255
Interrupts: 33726
BIOS Control Word: 0x18a6
Adapter Control Word: 0x1c5f
Extended Translation: Enabled
Disconnect Enable Flags: 0x00ff
Ultra Enable Flags: 0x0020
Tag Queue Enable Flags: 0x0000
Ordered Queue Tag Flags: 0x0000
Default Tag Queue Depth: 8
Tagged Queue By Device array for aic7xxx
host instance 1:
{255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255}
Actual queue depth per device for aic7xxx host instance 1:
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}
Statistics:

(scsi1:0:5:0) Device using Narrow/Sync transfers at 20.0 MByte/sec, offset
15
Transinfo settings: current(12/15/0/0), goal(12/15/0/0), user(12/15/0/0)
Total transfers 0 (0 reads and 0 writes)
  < 2K      2K+     4K+     8K+    16K+    32K+    64K+   128K+
Reads:         0       0       0       0       0       0       0       0
Writes:        0       0       0       0       0       0       0       0

(scsi1:0:6:0) Device using Narrow/Sync transfers at 10.0 MByte/sec, offset
15
Transinfo settings: current(25/15/0/0), goal(12/15/0/0), user(12/15/0/0)
Total transfers 132 (0 reads and 132 writes)
  < 2K      2K+     4K+     8K+    16K+    32K+    64K+   128K+
Reads:         0       0       0       0       0       0       0       0
Writes:        0       0       0       1     131       0       0       0
```

上記の出力では、チャンネル ID に基づいてコントローラーに接続された SCSI デバイスへの転送速度と、そのデバイスが読み取りもしくは書き込みしたファイルの量とサイズに関する統計が表示されています。例えば、このコントローラーが CD-ROM と通信する速度は 20 メガバイト/秒ですが 、テープドライブとの通信はわずか 10 メガバイト/秒 です。

E.3.9. /proc/sys/

/proc/sys/ ディレクトリは、システムに関する情報を提供するだけでなく、システム管理者がカーネル機能を即時に有効化または無効化できるようするので、**/proc/** 内のその他のディレクトリとは異なります。

警告

/proc/sys/ ディレクトリ内のファイルを使用して、実稼働システムの設定を変更する場合には、慎重に行って下さい。設定を誤って変更すると、カーネルが不安定な状態となり、システムの再起動が必要となる場合があります。

このため、**/proc/sys/** 内の値の変更を試みる前には、そのファイルでオプションが有効であることを確認するようにして下さい。

特定のファイルが設定可能か、情報提供専用に設計されているかを判断するには、シェルプロンプトで **-l** オプションを使用して一覧表示すると簡単に確認することができます。ファイルが書き込み可能の場合には、カーネルの設定に使用することができます。以下は **/proc/sys/fs** の一覧の一部です:

```
-r--r--r--    1 root     root            0 May 10 16:14 dentry-state
-rw-r--r--    1 root     root            0 May 10 16:14 dir-notify-enable
-rw-r--r--    1 root     root            0 May 10 16:14 file-max
-r--r--r--    1 root     root            0 May 10 16:14 file-nr
```

上記の一覧では、**dir-notify-enable** 及び **file-max** のファイルは書き込み可能なので、カーネルの設定に使用できることになります。その他のファイルは、現在の設定についてのフィードバックを提供するのみです。

/proc/sys/ ファイル内の値は、そのファイルに新たな値を echo することによって変更することができます。例えば、実行中のカーネルでシステム要求キーを有効にするには、以下のコマンドを入力します:

```
echo 1 > /proc/sys/kernel/sysrq
```

このコマンドにより、**sysrq** の値が **0** (off) から **1** (on) に変更されます。

/proc/sys/ の設定ファイルには、複数の値が記載されているファイルがいくつかあります。そのようなファイルに新たな値を適正に送信するには、以下の例に示したように、**echo** コマンドで渡す各値の間に空白を入れて下さい:

```
echo 4 2 45 > /proc/sys/kernel/acct
```

注記

echo コマンドを使用して変更した設定はいずれも、システムの再起動後には無効となります。システムの再起動後にも設定の変更が有効となるようにするには、「sysctl コマンドの使用」 を参照して下さい。

/proc/sys/ ディレクトリには、実行中のカーネルの異なる側面を制御する複数のサブディレクトリが格納されています。

E.3.9.1. /proc/sys/dev/

このディレクトリは、システム上の特定のデバイス用のパラメーターを提供します。大半のシステムに

は、少なくとも **cdrom/** と **raid/** の 2 つのディレクトリがあります。カスタマイズされたカーネルには、**parport/** などのその他のディレクトリがあり、これにより複数のデバイスドライバー間で単一のパラレルポートを共有する機能が提供されます。

cdrom/ ディレクトリには、**info** という名前のファイルが格納されており、数多くの重要な CD-ROM パラメーターが記載されています:

```
CD-ROM information, Id: cdrom.c 3.20 2003/12/17
drive name:             hdc
drive speed:            48
drive # of slots:       1
Can close tray:         1
Can open tray:          1
Can lock tray:          1
Can change speed:       1
Can select disk:        0
Can read multisession:  1
Can read MCN:           1
Reports media changed:  1
Can play audio:         1
Can write CD-R:         0
Can write CD-RW:        0
Can read DVD:           0
Can write DVD-R:        0
Can write DVD-RAM:      0
Can read MRW:           0
Can write MRW:          0
Can write RAM:          0
```

このファイルをチェックすることにより、未知の CD-ROM の性質を迅速に確認することができます。システム上で複数の CD-ROM が使用可能の場合には、 デバイスごとの情報カラムが提供されます。

autoclose や **checkmedia** など、**/proc/sys/dev/cdrom** の様々なファイルを使用して、システムの CD-ROM を制御することができます。これらの機能を有効化/無効化するには、**echo** コマンドを使用して下さい。

RAID サポートがカーネルにコンパイルされている場合は、**/proc/sys/dev/raid/** ディレクトリが使用可能となり、少なくとも **speed_limit_min** と **speed_limit_max** の 2 つのファイルが格納されます。これらの設定は、ディスクの再同期のような I/O 集約型タスクを対象とする RAID デバイスのアクセラレーションが決定します。

E.3.9.2. /proc/sys/fs/

このディレクトリには、クォータ、ファイルハンドル、inode、dentry などを含む、ファイルシステムの様々な側面に関する一連のオプションと情報が格納されています。

binfmt_misc/ ディレクトリは、カーネルにその他のバイナリ形式のサポートを提供します。

/proc/sys/fs/ 内の重要なファイルには以下のようなファイルが含まれます:

- **dentry-state** — ディレクトリキャッシュのステータスを提供します。以下は、このファイルの出力例です:

  ```
  57411 52939 45 0 0 0
  ```

第 1 の値は、ディレクトリキャッシュエントリの合計数、第 2 の値は未使用のエントリ数を示しています。第 3 の値はディレクトリが解放されてから、再構築可能となるまでの時間を秒数で示しています。第 4 の値は、システムによって現在要求されているページ数の測定値です。最後の 2 つの値は未使用で、必ずゼロと表示されます。

- **file-max** — カーネルが割り当てるファイルハンドルの最大数を一覧表示します。このファイルで設定値を高くすると、使用可能なファイルハンドルが欠如しているために発生するエラーを解決することができます

- **file-nr** — 割り当て済みファイルハンドル、使用済みファイルハンドル、及びファイルハンドルの最大数を一覧表示します。

- **overflowgid** 及び **overflowuid** — 16 ビットのグループ/ユーザー ID のみをサポートしているシステムで使用するための固定のグループ ID 及びユーザー ID をそれぞれ定義します。

E.3.9.3. /proc/sys/kernel/

このディレクトリには、カーネルの操作に直接影響を及ぼす、様々な異なる設定ファイルが格納されています。最も重要なファイルには、以下のようなファイルがあります:

- **acct** — ログを格納するファイルシステム上での空き容量に基づいて、プロセスアカウンティングの中断を制御します。このファイルの内容は、デフォルトでは以下のようになります:

```
4 2 30
```

  第 1 の値は、ロギングを再開するために必要な空き容量、第 2 の値はロギングが中断されている場合の空き容量の割合の閾値を示しています。第 3 の値は、ロギングを中断すべきか、再開すべきかを判断するためにカーネルがファイルシステムをポーリングする間隔を秒数で表しています。

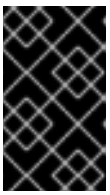
- **ctrl-alt-del** — **Ctrl**+**Alt**+**Delete** で、**init** (**0**) を使用してコンピューターを正常にを再起動するか、ダーティバッファをディスクに同期せずに即再起動を強制するかを制御します。

- **domainname** — システムのドメイン名 (例: **example.com**) を設定します。

- **exec-shield** — カーネルの Exec Shield 機能を設定します。Exec Shield は、特定のバッファオーバーフロー攻撃に対する保護を提供します。

  この仮想ファイルの取り得る値は 2 つあります:

  - **0** — Exec Shield を無効にします。
  - **1** — Exec Shield を有効にします。これはデフォルト値です。



重要

Exec Shield が無効の状態で起動された機密性の高いアプリケーションをシステムが実行している場合には、それらのアプリケーションは、Exec Shield を有効化する際に再起動して、Exec Shield が適用されるようにする必要があります。

- **hostname** — システムのホスト名 (例: **www.example.com**) を設定します。

- **hotplug** —システムによって設定の変更が検出された場合に使用されるユーティリティを設定します。これは、主にUSB と Cardbus PCI で使用されます。新たなプログラムがこの役割を果たすことをテストせずに **/sbin/hotplug** のデフォルト値を変更すべきではありません。

- **modprobe** — カーネルモジュールのロードに使用するプログラムの場所を設定します。デフォルト値は **/sbin/modprobe** です。これは、カーネルスレッドが **kmod** を呼び出した際に **kmod** がそのプログラムを呼び出して、モジュールをロードすることを意味します。

- **msgmax** — 1 つのプロセスから別のプロセスへ送られるメッセージの最大サイズを設定します。デフォルトでは **8192** バイトに設定されます。プロセス間の待ち行列にあるメッセージは、スワップ不可能なカーネルメモリに格納されるので、この値を高くする際には、注意して下さい。**msgmax** の値を高くすると、システムの RAM 要件が高くなります。

- **msgmnb** — 単一のメッセージキュー内の最大バイト数を設定します。デフォルトは **16384** です。

- **msgmni** — メッセージキュー識別子の最大数を設定します。デフォルトは **4008** です。

- **osrelease** — Linux カーネルのリリース番号を表示します。このファイルは、カーネルのソースと再コンパイルを変更することによってのみ変更可能です。

- **ostype** — オペレーティングシステムのタイプを表示します。デフォルトでは、このファイルは **Linux** に設定されており、この値は、カーネルのソースと再コンパイルを変更することによってのみ変更可能です。

- **overflowgid** 及び **overflowuid** — 16 ビットのグループ/ユーザー ID のみをサポートしているアーキテクチャのシステムコールで使用するための固定のグループ ID とユーザー ID をそれぞれ定義します。

- **panic** — システムでカーネルパニックが発生した際に、カーネルが再起動を延期する時間を秒数で定義します。この値は、デフォルトでは **0** に設定されており、パニック後の自動再起動が無効になっています。

- **printk** — このファイルは、エラーメッセージの出力またはロギングに関する様々な設定を制御します。カーネルによってレポートされる各エラーメッセージには、そのメッセージの重要度を定義する *loglevel* が関連付けされます。loglevel の値は、以下のような順序で細分化されます:

  - **0** — カーネルの緊急事態。システムは使用できません。
  - **1** — カーネルアラート。直ちに措置を取る必要があります。
  - **2** — カーネルの状態がクリティカルとみなされます。
  - **3** — 一般的なカーネルエラーの状態。
  - **4** — 一般的なカーネル警告の状態。
  - **5** — 正常ではあるが有意な状態であるというカーネル通知。
  - **6** — カーネルの情報メッセージ。
  - **7** — カーネルデバッグレベルメッセージ

  printk ファイルには、以下のような 4 つの値が記載されています:

  ■

```
6       4       1       7
```

これらの各値は、エラーメッセージへの対応の異なるルールをを定義します。第 1 の値は、*console loglevel* と呼ばれ、コンソールに出力されるメッセージの最も低い優先度を定義します (優先度が低いほど、ログレベルの数値が高くなることに注意して下さい)。第 2 の値は、明示的なログレベルが付いていないメッセージのデフォルトのログレベルを設定します。第 3 の値は、コンソールログレベルのログレベル設定の取り得る最低値を設定します。最後の値は、コンソールログレベルのデフォルト値を設定します。

- **random/** ディレクトリ — カーネルの乱数生成に関連した複数の値を一覧表示します。
- **sem** — カーネル内の *セマフォ* 設定値を設定します。セマフォとは、System V IPC オブジェクトで、特定のプロセスの使用率を制御する目的で使用されます。
- **shmall** — システム上で一度に使用可能な共有メモリ容量の合計をバイト単位で設定します。デフォルトでは、この値は **2097152** です。
- **shmmax** — カーネルによって許可される最大の共有メモリセグメントサイズを設定します。デフォルトでは、この値は **33554432** です。但し、カーネルはこれよりもはるかに大きな値に対応しています。
- **shmmni** — システム全体の共有メモリセグメントの最大数を設定します。デフォルトでは、この値は **4096** です。
- **sysrq** — デフォルト値のゼロ (**0**) 以外の値に設定されている場合には、システム要求キーをアクティベートします。

  システム要求キーは、簡単なキーの組み合わせを使用したカーネルへの即時入力を可能にします。例えば、システム要求キーは、システムを即時にシャットダウンまたは再起動したり、マウントされているファイルシステムをすべて同期したり、重要な情報をコンソールにダンプするのに使用することができます。システム要求キーを開始するには、**Alt**+**SysRq**+ ***system request code*** を入力します。*system request code* は、以下のシステム要求コードのいずれかに置き換えて下さい:

  - **r** — キーボードの raw モードを無効にし、XLATE (すべてのキーで **Alt**、**Ctrl**、**Shift** などの修飾キーを認識しない限定されたキーボードモード) に設定します。
  - **k** — 仮想コンソールでアクティブなプロセスをすべて強制終了します。これは、*Secure Access Key* (*SAK*) とも呼ばれ、ログインプロンプトが **init** から発生したもので、ユーザー名とパスワードをキャプチャーするように設計されたトロイの木馬のコピーではないことを検証します。
  - **b** — ファイルシステムのアンマウントやシステムに接続されたディスクの同期を最初に行わずにカーネルを再起動します。
  - **c** — ファイルシステムのアンマウントやシステムに接続されたディスクの同期を最初に行わずにシステムをクラッシュさせます。
  - **o** — システムをシャットオフします。
  - **s** — システムに接続されたディスクの同期を試みます。
  - **u** — 全ファイルシステムのアンマウントと、読み取り専用としての再マウントを試みます。
  - **p** — 全フラグを出力し、コンソールを登録します。

- **t** — プロセスの一覧をコンソールに出力します。
- **m** — メモリの統計をコンソールに出力します。
- **0** から **9** まで — コンソールのログレベルを設定します。
- **e** — SIGTERM を使用する **init** 以外のプロセスをすべて強制終了します。
- **i** — SIGKILL を使用する **init** 以外の全プロセスを強制終了します。
- **l** — SIGKILL を使用する全プロセス (**init** を含む) を強制終了します。**このシステム要求キーを発行した後には、システムは使用できなくなります。**
- **h** — ヘルプテキストを表示します。

この機能は、開発カーネルを使用する場合、もしくはシステムがフリーズした場合に最も役立ちます。

警告

コンソールが無人状態にあると、攻撃者によるシステム侵入が可能となるため、システム要求キー機能はセキュリティリスクと見なされます。このため、この機能はデフォルトではオフになっています。

システム要求キーに関する詳しい情報は、**/usr/share/doc/kernel-doc-*kernel_version*/Documentation/sysrq.txt** を参照して下さい。

- **tainted** —非 GPL モジュールがロードされているかどうかを示します。
  - **0** — 非 GPL モジュールはロードされていません。
  - **1** — GPL ライセンスのないモジュール (ライセンスのないモジュールを含む) が少なくとも 1 つロードされています。
  - **2** — 少なくとも 1 つのモジュールが **insmod -f** コマンドで強制ロードされています。
- **threads-max** — カーネルによって使用されるスレッドの最大数を設定します。デフォルト値は **2048** です。
- **version** — カーネルが最後にコンパイルされた日付と時刻を表示します。このファイルの第 1 のフィールド (例: **#3**) は、ソースベースからカーネルがビルドされた回数に関連しています。

E.3.9.4. /proc/sys/net/

このディレクトリには、様々なネットワークトピックに関するサブディレクトリが格納されています。カーネルのコンパイル時には、様々な設定により、**ethernet/**、**ipv4/**、**ipx/**、**ipv6/** などの異なるディレクトリがここに作成されます。システム管理者は、これらのディレクトリ内のファイルを変更して、稼働中のシステムのネットワーク設定を調整することができます。

Linux では多種多様なネットワーキングオプションが使用可能であることを考慮し、ここでは最も一般的な **/proc/sys/net/** ディレクトリのみについて説明します。

/proc/sys/net/core/ ディレクトリには、カーネルとネットワーキング層の間のインタラクションを制御する様々な設定が格納されています。それらの中で最も重要なのは、以下にあげるファイルです:

- **message_burst** — 新たな警告メッセージを書き込むのに必要な時間を 1/10 秒単位で設定します。この設定は、*サービス拒否* (*DoS*) 攻撃を軽減するのに使用します。デフォルトの設定値は **10** です。

- **message_cost** — すべての警告メッセージに対してコストを設定します。このファイルの値 (デフォルト値は **5**) が高いほど、警告メッセージが無視される可能性が高くなります。この設定は、DoS 攻撃を軽減するのに使用されます。

  DoS 攻撃の概念は、エラーを生成する要求を利用して、ディスクパーティションをログファイルで満杯にしたり、システムの全リソースがエラーロギングに対応しなければならない状態して、ターゲットシステムを攻撃することです。**message_burst** 及び **message_cost** の設定は、包括的なロギングの必要性に対するシステムの許容リスクに基づいて、変更されるように設計されています。

- **netdev_max_backlog** — 特定のインターフェースが、カーネルの処理可能速度よりも高速にパケットを受信した場合に待ち行列に入れることができるパケットの最大数を設定します。このファイルのデフォルト値は **1000** です。

- **optmem_max** — 1 ソケットあたりで許可される、最大の補助バッファーサイズを設定します。

- **rmem_default** — 受信ソケットバッファーのデフォルトサイズをバイト単位で設定します。

- **rmem_max** — 受信ソケットバッファーの最大サイズをバイト単位で設定します。

- **wmem_default** — 送信ソケットバッファーのデフォルトサイズをバイト単位で設定します。

- **wmem_max** — 送信ソケットバッファーの最大サイズをバイト単位で設定します。

/proc/sys/net/ipv4/ ディレクトリには、追加のネットワーク設定が格納されています。これらの設定値の多くは、相互に連動するように使用され、システムに対する攻撃を防止したり、システムをルーターとして機能させるように使用している場合に役立ちます。

警告

これらのファイルの変更でエラーがあると、システムへのリモート接続に影響を及ぼす可能性があります。

以下は、**/proc/sys/net/ipv4/** ディレクトリ内の重要なファイルの一覧です:

- **icmp_echo_ignore_all** 及び **icmp_echo_ignore_broadcasts** — 各ホストからの ICMP ECHO パケット、もしくはブロードキャスト及びマルチキャストのアドレスを起点とする ICMP ECHO パケットのみを無視することをそれぞれカーネルに許可します。**0** の値は、カーネルによる応答を許可し、**1** の値はパケットを無視します。

- **ip_default_ttl** — デスティネーションに到達するまでに 1 つのパケットが作成することのできるホップ数を制限する、デフォルトの *Time To Live (TTL)* を設定します。この値を高くすると、システムのパフォーマンスが低下する可能性があります。

- **ip_forward** — システム上のインターフェースが相互にパケットを送信するのを許可します。デフォルトでは、このファイルは **0** に設定されています。このファイルを **1** に設定すると、ネットワークパケット転送が有効になります。

- **ip_local_port_range** — ローカルポートが必要な場合に TCP または UDP が使用するポートの範囲を指定します。第 1 の数値は、使用すべき最も低いポートです。第 2 の数値は、最も高いポートを指定します。デフォルトの 1024 から 4999 よりも多くのポートが必要となることが予想されるシステムでは、32768 から 61000 の範囲を使用すべきです。

- **tcp_syn_retries** — システムが 接続を試みる際に SYN パケットを再転送する回数の上限を示します

- **tcp_retries1** — 着信接続への応答を試行する再転送の許容試行回数を設定します。デフォルト値は **3** です。

- **tcp_retries2** — TCP パケットの許容再転送回数を設定します。デフォルトは **15** です。

ファイル名:

```
/usr/share/doc/kernel-doc-kernel_version/Documentation/networking/ip-sysctl.txt
```

/proc/sys/net/ipv4/ ディレクトリで利用可能なファイルとオプションの完全な一覧が記載されています。

/proc/sys/net/ipv4/ ディレクトリ内には、その他のディレクトリが数多く存在します。各ディレクトリは、ネットワークスタックの異なる側面を網羅します。**/proc/sys/net/ipv4/conf/** ディレクトリは、各システムインターフェースを異なる方法で設定できるようにします。これには、未設定のデバイス用のデフォルト設定の使用 (**/proc/sys/net/ipv4/conf/default/** サブディレクトリ内) や全特殊設定の上書き (**/proc/sys/net/ipv4/conf/all/** サブディレクトリ内) が含まれます。

/proc/sys/net/ipv4/neigh/ ディレクトリには、システムに直接接続されたホスト (ネットワークネイバーと呼ばれる) との通信のための設定が格納されています。又、1 ホップ以上離れたシステム用の異なる設定も含まれています。

IPV4 上のルーティングにも、独自のディレクトリ **/proc/sys/net/ipv4/route/** があります。**conf/** や **neigh/** とは異なり、**/proc/sys/net/ipv4/route/** ディレクトリにはシステム上の任意のインターフェースとのルーティングに適用される指定値が格納されています。**max_size**、**max_delay**、**min_delay** など、これらの設定の多くは、ルーティングキャッシュサイズの制御に関連しています。ルーティングキャッシュをクリアするには、**flush** ファイルに任意の値を書き込んで下さい。

これらのディレクトリ及び設定ファイルの取り得る値に関する更なる情報は、以下の資料を参照して下さい:

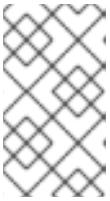
```
/usr/share/doc/kernel-doc-kernel_version/Documentation/filesystems/proc.txt
```

E.3.9.5. /proc/sys/vm/

このディレクトリにより、Linux カーネルの仮想メモリ (VM) サブシステムの設定を容易に行うことができます。カーネルは、通常 swap 領域と呼ばれる仮想メモリを広範かつ賢明に活用します。

/proc/sys/vm/ ディレクトリには、通常以下のようなファイルが格納されています:

- **block_dump** — 有効化されている場合には、ブロック I/O デバッグを設定します。ファイルに対して実行される読み取り/書き込みならびにブロックをダーティにするすべての操作は、適宜にロギングされます。これは、ラップトップのバッテリーを節約するためにディスクのスピンアップとスピンダウンを診断する場合に役立ちます。**block_dump** が有効化されている場合、すべての出力は、**dmesg** で取得することができます。デフォルト値は **0** です。

  注記

  カーネルデバッグと同時に **block_dump** が有効化されている場合には、**block_dump** により、誤ったディスクアクティビティが発生するため、**klogd** デーモンを停止するのが賢明です。

- **dirty_background_ratio** — pdflush デーモンを使用して、総メモリ合計に対するパーセンテージで、バックグラウンドでのダーティデータのライトバックを開始します。デフォルト値は **10** です。

- **dirty_expire_centisecs** — ダーティインメモリデータがライトアウトの対象となる時間間隔を定義します。これよりも長くダーティインメモリであったデータは、pdflush デーモンが次回にウェイクアップする時に書き出されます。デフォルト値は **3000** で、1/100 秒単位で表示されます。

- **dirty_ratio** — pdflush を介し、ダーティデータのジェネレーター用のメモリ合計に対してこの割合で、ダーティデータのアクティブライトバックを開始します。デフォルト値は **20** です。

- **dirty_writeback_centisecs** — ダーティインメモリデータをディスクに定期的に書き出す、pdflush デーモンのウェイクアップ間隔を定義します。デフォルト値は **500** で、1/100 秒単位で表示されます。

- **laptop_mode** — 可能な限り長時間、ディスクスパンを低くすることにより、ハードディスクがスピンアップする必要のある回数を最小限に抑え、ラップトップのバッテリーを節約します。これにより、将来の I/O プロセスがすべて統合されるため、効率が高くなり、スピンアップの頻度が低減します。デフォルト値は **0** ですが、ラップトップのバッテリーを使用する場合には自動的に有効化されます。

  バッテリー電源が有効化されたことがユーザーに通知された後には、この値は、acpid デーモンにより自動的に制御されます。ラップトップが ACPI (Advanced Configuration and Power Interface) 仕様をサポートしている場合には、ユーザーによる変更やインタラクションは必要ありません。

  更なる情報は、システムにインストールされている以下のドキュメントを参照して下さい:

  /usr/share/doc/kernel-doc-*kernel_version*/Documentation/laptop-mode.txt

- **max_map_count** — 1 プロセスで使用できるメモリマップ領域の最大数を設定します。大半の場合、デフォルト値は **65536** が適切となります。

- **min_free_kbytes** — Linux VM (仮想メモリマネージャー) が最小キロバイト数を空けておくように強制します。VM は、この数を使用して、システム内の各 **lowmem** ゾーンの **pages_min** 値を算出します。デフォルト値はマシン上のメモリ合計に対応します。

- **nr_hugepages** — カーネル内の設定済み **hugetlb** ページの現在の数を示します。

  更なる情報は、システムにインストールされている以下のドキュメントを参照して下さい:

  /usr/share/doc/kernel-doc-*kernel_version*/Documentation/vm/hugetlbpage.txt

- **nr_pdflush_threads** — 現在実行中の pdflush デーモンの数を示します。このファイルは、読み取り専用で、ユーザーが変更してはなりません。I/O 負荷が高い場合には、カーネルにより、デフォルト値の 2 に加算されます。

- **overcommit_memory** — 大容量メモリ要求が受理または拒否される条件を設定します。使用可能なモードは以下の 3 つです:

  - **0** — カーネルは、メモリの空き容量を概算し、明らかに無効な要求を失敗させることにより、ヒューリスティックなメモリオーバーコミット処理を実行します。残念ながら、メモリーは、厳密なアルゴリズムではなくヒューリスティックなアルゴリズムを使用して割り当てられるため、この設定により、システム上のメモリの空き容量のオーバーロードが可能となる場合があります。これがデフォルトの設定です。

  - **1** — カーネルはメモリオーバーコミット処理を行いません。この設定では、メモリーがオーバーロードされる可能性が高くなりますが、メモリ集約型のタスク (例: 一部の科学ソフトウェアによって実行されるタスク) のパフォーマンスも向上します。

  - **2** — 要求によって、総アドレス空間が、割り当て済み swap 領域と **/proc/sys/vm/overcommit_ratio** で指定されている割合の物理 RAM の合計を超過してしまう場合、カーネルはその要求を失敗さます。この設定は、メモリオーバーコミットのリスクを低減したい場合に最適です。

    注記

    この設定は、スワップ領域が物理メモリよりも大きいシステムのみで使用することを推奨します

- **overcommit_ratio** — **/proc/sys/vm/overcommit_memory** が **2** に設定されている場合に考慮される物理 RAM の割合を指定します。デフォルト値は **50** です。

- **page-cluster** — 1 回の試行で読み込まれるページ数を設定します。デフォルト値は **3** で、実際には 16 ページを意味しており、大半のシステムで適切な値です。

- **swappiness** — 1 台のマシンでどれだけスワップすべきかを決定します。値が高いほど、多くスワップが発生します。デフォルト値は **60** パーセントです。

カーネル関連の全ドキュメントがローカルでインストールされている場所は以下の通りです:

/usr/share/doc/kernel-doc-*kernel_version*/Documentation/ (更なる情報が含まれています)

E.3.10. /proc/sysvipc/

このディレクトリには、System V IPC リソースに関する情報が格納されています。このディレクトリ内のファイルは、メッセージ (**msg**)、セマフォ (**sem**)、及び共有メモリ (**shm**) の System V IPC コールに関する内容です。

E.3.11. /proc/tty/

このディレクトリには、システム上で使用可能及び現在使用中の *tty devices* についての情報が格納されています。これは、元々は *teletype devices* と呼ばれていました。文字ベースのデータターミナルはいずれも tty デバイスと呼ばれます。

Linux では、3 つの異なる tty デバイスがあります。*シリアルデバイス* は、モデムやシリアルケーブルを使用したシリアル接続で使用されるデバイスです。*仮想ターミナル* は、システムコンソールで **Alt+<F-key>** を押すと使用可能な仮想コンソールなど、共通のコンソール接続を作成します。*擬似ターミナル* は、XFree86 などの、一部のハイレベルアプリケーションで使用される双方向通信を構築します。**drivers** ファイルは、現在使用中の tty デバイス一覧で、以下の例のような内容が記載されています:

```
serial               /dev/cua        5  64-127 serial:callout
serial               /dev/ttyS       4  64-127 serial
pty_slave            /dev/pts      136   0-255 pty:slave
pty_master           /dev/ptm      128   0-255 pty:master
pty_slave            /dev/ttyp       3   0-255 pty:slave
pty_master           /dev/pty        2   0-255 pty:master
/dev/vc/0            /dev/vc/0       4       0 system:vtmaster
/dev/ptmx            /dev/ptmx       5       2 system
/dev/console         /dev/console    5       1 system:console
/dev/tty             /dev/tty        5       0 system:/dev/tty
unknown              /dev/vc/%d      4    1-63 console
```

/proc/tty/driver/serial ファイルは、各シリアル tty 回線の使用率の統計とステータスを示します。

tty デバイスをネットワークデバイスとして使用するために、Linux カーネルは デバイスに対して *回線規範* を強制します。これによりドライバーは、そのデバイス上で転送される各データブロックに特定タイプのヘッダーを配置することができるため、通信のリモート側は一つのデータブロクを複数のデータブロックからなるストリームの中の単なる一つのとして扱うことが可能となります。 SLIP および PPP は一般的な回線規範で、いずれも、シリアルリンクを介したシステム間の接続に広く利用されます。

E.3.12. /proc/*PID*/

Out of Memory (OOM) とは、swap 領域を含むすべてのメモリ空き容量が割り当て済みの状態であることを指します。このような状態が発生した場合には、システムがパニックを起し、正常機能を停止してしまいます。OOM の動作を制御するスイッチは、**/proc/sys/vm/panic_on_oom** にあります。**1** に設定すると、カーネルは、OOM でパニックを起こします。**0** は、カーネルが **oom_killer** という名前の関数を OOM に対して呼び出すように指示します。通常、 **oom_killer** はローグプロセスを強制終了し、システムは稼働状態を維持することができます。

この設定を変更する最も簡単な方法は、**/proc/sys/vm/panic_on_oom** に新たな値を echo する方法です。

```
# cat /proc/sys/vm/panic_on_oom
1

# echo 0 > /proc/sys/vm/panic_on_oom
```

```
# cat /proc/sys/vm/panic_on_oom
0
```

oom_killer スコアを調整して、強制終了するプロセスの優先度を設定することもできます。**/proc/*PID*/** には、**oom_adj** 及び **oom_score** という名前の付いた 2 つのツールがあります。**oom_adj** の有効なスコアは、-16 から +15 までの範囲内になります。現行の **oom_killer** スコアは、そのプロセスの **oom_score** で確認して下さい。**oom_killer** は、スコアが最も高いプロセスを最初に強制終了します。

以下の例では、*PID* が 12465 のプロセスの oom_score of a process を調整し、**oom_killer** によって強制終了される優先度を低くします。

```
# cat /proc/12465/oom_score
79872

# echo -5 > /proc/12465/oom_adj

# cat /proc/12465/oom_score
78
```

又、-17 という特殊な値もあり、そのプロセスに対して、**oom_killer** を無効にします。以下の例では、**oom_score** は 0 の値を返しており、このプロセスが強制終了されないことを示しています。

```
# cat /proc/12465/oom_score
78

# echo -17 > /proc/12465/oom_adj

# cat /proc/12465/oom_score
0
```

badness() と呼ばれる関数は、各プロセスの実際のスコアを判断するのに使用されます。これは、検討の対象となる各プロセス用に「ポイント」を加算することによって実行されます。プロセスのスコア調整は以下の方法で行われます:

1. 各プロセスのメモリサイズが、スコアのベースとなります。
2. プロセスの子プロセス (カーネルスレッドは除く) のメモリサイズも、このスコアに加算されます。
3. 'niced' (nice 度の高い) プロセスにはプロセスのスコアが加算され、長時間実行されているプロセスには削減されます。
4. **CAP_SYS_ADMIN** 及び **CAP_SYS_RAWIO** の機能に伴うプロセスは、スコアが削減されます。
5. 最終スコアは、**oom_adj** ファイルに保管されている値によってビットシフトされます。

従って、**oom_score** 値が最も高いプロセスは、優先されず、最近開始したプロセスの可能性が高く、これはその子プロセスとともに、大容量のメモリを使用し、「nice 度が高く」、raw I/O を一切処理しないものです。

E.4. SYSCTL コマンドの使用

/sbin/sysctl コマンドは、**/proc/sys/** ディレクトリ内のカーネルの表示、設定、及び自動化に使用します。

/proc/sys/ ディレクトリ内で設定可能な全設定値を手早く確認するには、root として **/sbin/sysctl -a** コマンドを入力します。これにより、大きな包括的一覧が作成されます。以下は部分的な出力の例です:

```
net.ipv4.route.min_delay = 2 kernel.sysrq = 0 kernel.sem = 250     32000
32     128
```

この情報は、各ファイルを個別に表示するのと同じ内容です。唯一の相違点は、ファイルの場所です。例えば、**/proc/sys/net/ipv4/route/min_delay** ファイルは、**net.ipv4.route.min_delay** として記載され、ディレクトリのスラッシュがドットに置き換えられる他、**proc.sys** の部分は前提とされます。

echo の代わりに **sysctl** コマンドを使用して、**/proc/sys/** ディレクトリ内の書き込み可能なファイルに値を割り当てることができます。以下が例となります:

```
echo 1 > /proc/sys/kernel/sysrq
```

echo の代わりとして、相当する **sysctl** コマンドを以下の例のように使用します。

```
sysctl -w kernel.sysrq="1"
kernel.sysrq = 1
```

テスト中などには、**/proc/sys/** 内にあるような単一の値を迅速に設定できると便利ですが、実稼働システムでは、**/proc/sys/** 内の特殊な設定がマシンの再起動時に消去されてしまうため、このコマンドは適切に機能しません。カスタム設定を維持するには、**/etc/sysctl.conf** ファイルに追加する必要があります。

システムの起動時には毎回、**init** プログラムによって **/etc/rc.d/rc.sysinit** スクリプトが実行されます。このスクリプトには、**/etc/sysctl.conf** を使用して **sysctl** を実行し、カーネルに渡す値を決定するためのコマンドが含まれています。このため、**/etc/sysctl.conf** に追加された値はいずれも、システムの起動時に毎回有効となります。

E.5. その他のリソース

以下の資料は、**proc** ファイルシステムに関するその他のリソースです。

E.5.1. インストールされているドキュメント

proc ファイルシステムについての最も役立つ資料は、デフォルトでシステムにインストールされています。

- **/usr/share/doc/kernel-doc-*kernel_version*/Documentation/filesystems/proc.txt** — C=**/proc/** ディレクトリの全側面に関する、多様な情報が記載されていますが、内容は限定されています。
- **/usr/share/doc/kernel-doc-*kernel_version*/Documentation/sysrq.txt** — システム要求キーに関する概要が記載されています。
- **/usr/share/doc/kernel-doc-*kernel_version*/Documentation/sysctl/** — **sysctl** に関する様々なヒントをまとめたディレクトリです。カーネルに関連した値の変更

(**kernel.txt**)、ファイルシステムへのアクセス (**fs.txt**)、仮想メモリの使用 (**vm.txt**) などに関する情報が含まれます。

- **/usr/share/doc/kernel-doc-*kernel_version*/Documentation/networking/ip-sysctl.txt** — IP ネットワークオプションについての詳しい説明をまとめています。

E.5.2. 役立つ Web サイト

- http://www.linuxhq.com/ — この Web サイトは、様々なバージョンの Linux カーネルのソース、パッチ、ドキュメントを提供しています。

付録F 改訂履歴

改訂 4-0.1.400 **2013-10-31** **Rüdiger Landmann**
Rebuild with publican 4.0.0

改訂 4-0.1 **Tue Apr 23 2013** **Credit Translator's**
翻訳完成

改訂 4-0 **Thu Feb 21 2013** **Jaromír Hradílek**
Red Hat Enterprise Linux 6.4 GA リリースの導入ガイド

改訂 3-1.1 **Mon Dec 17 2012** **Chester Cheng**
翻訳ファイルをXMLソース 3-1 と同期

改訂 3-0 **Wed Jun 20 2012** **Jaromír Hradílek**
Red Hat Enterprise Linux 6.3 GA リリースの導入ガイド

改訂 2-1 **Tue Dec 6 2011** **Jaromír Hradílek**
Red Hat Enterprise Linux 6.2 GA リリースの導入ガイド

改訂 2-0 **Mon Oct 3 2011** **Jaromír Hradílek**
Red Hat Enterprise Linux 6.2 Beta リリースの導入ガイド

改訂 1-1 **Wed May 19 2011** **Jaromír Hradílek**
Red Hat Enterprise Linux 6.1 GA リリースの導入ガイド

改訂 1-0 **Tue Mar 22 2011** **Jaromír Hradílek**
Red Hat Enterprise Linux 6.1 Beta リリースの導入ガイド

改訂 0-1 **Tue Nov 09 2010** **Douglas Silas**
Red Hat Enterprise Linux 6.0 GA リリースの導入ガイド

改訂 0-0 **Mon Nov 16 2009** **Douglas Silas**
初版作成。Red Hat Enterprise Linux 6 導入ガイド

索引

シンボル

.fetchmailrc, Fetchmail の設定オプション
    サーバーオプション, サーバーオプション
    ユーザーオプション, ユーザーオプション

.htaccess , 一般的な httpd.conf の指示文
    (参照 Apache HTTP サーバー)

.htpasswd , 一般的な httpd.conf の指示文
    (参照 Apache HTTP サーバー)

.procmailrc, Procmail の設定
/dev/oprofile/, /dev/oprofile/ を理解する
/etc/named.conf (参照 BIND)
/etc/sysconfig/ ディレクトリ (参照 sysconfig ディレクトリ)
/etc/sysconfig/dhcpd, サーバーの起動と停止
/proc/ ディレクトリ (参照 proc ファイルシステム)
/var/spool/anacron , Anacron ジョブの設定
/var/spool/cron , Cron ジョブの設定
アクセス制御
    SSSD での設定, ドメインの作成: アクセス制御
    SSSD ルール, Simple Access Provider の使用

イーサネット (参照 ネットワーク)
ウィンドウマネージャー (参照 X)
エポック, /proc/stat
    (参照 /proc/stat)

カーネル
    RPM パッケージ, 手動のカーネルアップグレード
    アップグレード
        機能するブートメディア, アップグレードへの準備
        準備, アップグレードへの準備

    カーネルのアップグレード, 手動のカーネルアップグレード
    カーネルアップグレードの実行, アップグレードの実行
    カーネルパッケージ, カーネルパッケージの概要
    カーネルパッケージのインストール, 手動のカーネルアップグレード
    ダウンロード, アップグレードされたカーネルのダウンロード
    パッケージ, 手動のカーネルアップグレード
    利用可能なカーネルアップグレード, アップグレードされたカーネルのダウンロード
        Red Hat network 経由で, アップグレードされたカーネルのダウンロード

セキュリティ Errata, アップグレードされたカーネルのダウンロード

カーネルのアップグレード
準備, アップグレードへの準備

カーネルのインストール, 手動のカーネルアップグレード
カーネルダンプの設定 (参照 kdump)
カーネルパッケージ
kernel-devel
カーネルヘッダと makefiles, カーネルパッケージの概要

kernel-doc
ドキュメントファイル, カーネルパッケージの概要

kernel-firmware
ファームウェアファイル, カーネルパッケージの概要

kernel-headers
C header files のファイル, カーネルパッケージの概要

perf
ファームウェアファイル, カーネルパッケージの概要

カーネル
シングル、マルチコア、及びマルチプロセッサのシステム対応, カーネルパッケージの概要

カーネルモジュール
bonding モジュール, チャンネルボンディングの使用
ボンディングされたインターフェースのパラメーター, bonding モジュールの指示文
詳細, チャンネルボンディングの使用

アンロード, モジュールのアンロード
イーサネットモジュール
複数のカードに対応, 複数のイーサネットカードの使用

ディレクトリ
/etc/sysconfig/modules/, 永続的なモジュールのロード
/lib/modules/<kernel_version>/kernel/drivers/, モジュールのロード

ファイル
/proc/modules, 現在ロード済みモジュールの一覧表示

モジュールパラメーター
bonding モジュールのパラメーター, bonding モジュールの指示文
指定, モジュールパラメーターの設定

ユーティリティ
insmod, モジュールのロード
lsmod, 現在ロード済みモジュールの一覧表示
modinfo, モジュール情報の表示
modprobe, モジュールのロード, モジュールのアンロード
rmmod, モジュールのアンロード

ロード
ブート時, 永続的なモジュールのロード
現行のセッション用, モジュールのロード

一覧表示
モジュール情報, モジュール情報の表示
現在ロード済みのモジュール, 現在ロード済みモジュールの一覧表示

定義, カーネルモジュールでの作業

キャッシュファイルの削除
SSSD, ドメインキャッシュファイルの削除

キャラクターデバイス, /proc/devices
(参照 /proc/devices)
定義, /proc/devices

キーボードの設定 (参照 キーボードの設定)
キーボードの設定 ユーティリティ, キーボードレイアウトの変更
キーボード表示器 アプレット, キーボードレイアウト表示器の追加
レイアウト, キーボードレイアウトの変更
一休み, 一休みの設定

キーボード表示器 (参照 キーボードの設定)
キーボード設定, キーボード設定
グループ (参照 グループ設定)
GID, ユーザーとグループの管理
その他のリソース, その他のリソース
インストールされているドキュメント, インストールされているドキュメント

ユーザープライベート, ユーザープライベートグループ
以下を管理するためのツール
groupadd, ユーザープライベートグループ, コマンドラインツールの使用
system-config-users, ユーザープライベートグループ

以下を管理するツール
ユーザー管理, コマンドラインツールの使用

共有ディレクトリ, グループディレクトリの作成

導入, ユーザーとグループの管理

グループ設定
groupadd, 新規グループの追加
グループの一覧をフィルター, ユーザーとグループの表示
グループの一覧を表示, ユーザー管理ツールの使用
グループの追加, 新規グループの追加
グループプロパティの修正, グループプロパティの変更
グループ内のユーザーを修正, グループプロパティの変更

サブスクリプション, システム登録およびサブスクリプションの管理
アタッチ
GUI 使用, サブスクリプションのアタッチ
コマンドライン使用, サブスクリプションのアタッチ

アタッチと削除, サブスクリプションのアタッチと削除
クライアントツール, Red Hat サブスクリプションマネージャーツールの使用
CLI コマンド, サブスクリプションマネージャーのコマンドラインツールの実行
Red Hat サブスクリプションマネージャーの起動, Red Hat サブスクリプションマネージャー GUI の起動

再登録, システムの登録および登録解除
削除
GUI 使用, サブスクリプションの削除
コマンドライン使用, コマンドラインを使用したサブスクリプションの削除

登録, システムの登録および登録解除
GUI から, GUI からの登録
コマンドラインから, コマンドラインからの登録

登録解除, システムの登録および登録解除, 登録解除
通知, サブスクリプションの有効期限と通知の管理

サービスの設定, サービスとデーモン
chkconfig, chkconfig ユーティリティの使用
ntsysv, ntsysv ユーティリティの使用
runlevel, デフォルトのランレベルの設定
service, サービスの実行
system-config-services, サービス設定ユーティリティの使用

サービス拒否攻撃, /proc/sys/net/
(参照 /proc/sys/net/ ディレクトリ)
定義, /proc/sys/net/

システム
アタッチ

コマンドライン使用, サブスクリプションのアタッチ

サブスクライブ
GUI 使用, サブスクリプションのアタッチ

サブスクリプションのアタッチと削除, サブスクリプションのアタッチと削除
サブスクリプション管理, システム登録およびサブスクリプションの管理
再登録, システムの登録および登録解除
削除
GUI 使用, サブスクリプションの削除
コマンドライン使用, コマンドラインを使用したサブスクリプションの削除

登録, システム登録およびサブスクリプションの管理, システムの登録および登録解除
GUI から, GUI からの登録
コマンドラインから, コマンドラインからの登録

登録解除, システムの登録および登録解除, 登録解除
通知, サブスクリプションの有効期限と通知の管理

システムの情報
プロセス, システムプロセスの表示
現在実行中, top コマンドの使用

収集, システムモニタリングツール

システムモニター, システムモニターツールの使用, システムモニターツールの使用, システムモニターツールの使用, システムモニターツールの使用
システム分析
OProfile (参照 OProfile)

システム情報
CPU 使用率, CPU 使用率の表示
ハードウェア, ハードウェア情報の表示
ファイルシステム, ブロックデバイスとファイルシステムの表示
メモリ使用量, メモリ使用量の表示

システム要求キー
タイミングの設定, /proc/sys/kernel/
定義, /proc/sys/
有効化, /proc/sys/

シャドウパスワード
概要, シャドウパスワード

スラブプール (参照 /proc/slabinfo)
セカンダリネームサーバー (参照 BIND)

セキュリティ関連パッケージ
セキュリティ関連パッケージの更新, パッケージの更新

タイムゾーンの設定, タイムゾーンのプロパティ
チャンネルボンディング
インターフェース
その設定, チャンネルボンディングインターフェース

ボンディングされたインターフェースのパラメーター, bonding モジュールの指示文
設定, チャンネルボンディングの使用
詳細, チャンネルボンディングの使用

チャンネルボンディングインターフェース (参照 カーネルモジュール)
ツール
認証設定ツール , システム認証の設定

ディスプレイマネージャー (参照 X)
ディレクトリサーバー (参照 OpenLDAP)
デスクトップ環境 (参照 X)
デフォルトゲートウェイ, 静的ルートおよびデフォルトゲートウェイ
ドキュメンテーション
インストール済みを検索, 実用的で一般的な RPM の使用例

ドライバー (参照 カーネルモジュール)
ネットワーク
その他のリソース, その他のリソース
インターフェース
802.1q, 802.1q VLAN タグのセットアップ
ethtool, Ethtool
VLAN, 802.1q VLAN タグのセットアップ
イーサネット, イーサネットインターフェース
エイリアス, エイリアスファイルとクローンファイル
クローン, エイリアスファイルとクローンファイル
ダイヤルアップ, ダイヤルアップインターフェース
チャンネルボンディング, チャンネルボンディングインターフェース

インターフェース設定ファイル, インターフェース設定ファイル
コマンド
/sbin/ifdown , インターフェース制御スクリプト
/sbin/ifup , インターフェース制御スクリプト
/sbin/service network , インターフェース制御スクリプト

スクリプト, ネットワークインターフェース
ブリッジ

ブリッジング, ネットワークブリッジ

設定, インターフェース設定ファイル
設定ファイル, ネットワーク設定ファイル
関数, ネットワーク関数ファイル

ネットワーク時刻プロトコル (参照 NTP)
ハードウェア
表示, ハードウェア情報の表示

パスワード
シャドウ, シャドウパスワード

パッケージ
Enterprise Linux 用エキストラパッケージ (EPEL), RPM パッケージの検索
iRed Hat Enterprise Linux インストールメディア, RPM パッケージの検索
kernel-devel
カーネルヘッダと makefiles, カーネルパッケージの概要

kernel-doc
ドキュメントファイル, カーネルパッケージの概要

kernel-firmware
ファームウェアファイル, カーネルパッケージの概要

kernel-headers
C header files のファイル, カーネルパッケージの概要

PackageKit を使用した Yum リポジトリの表示, ソフトウェアソースを最新の情報への更新 (yum リポジトリ)
PackageKit を使用したパッケージのアンインストール, PackageKit
PackageKit を使用したパッケージのインストール, PackageKit, パッケージ (及び依存関係) のインストールと削除
PackageKit を使用したパッケージのインストール
依存関係, パッケージ (及び依存関係) のインストールと削除

PackageKit を使用したパッケージのフィルター, フィルターを使ったパッケージ検索
PackageKit を使用したパッケージの削除, パッケージ (及び依存関係) のインストールと削除
PackageKit を使用したパッケージの更新, PackageKit
PolicyKit, ソフトウェア更新を使ったパッケージ更新
ソフトウェア更新, ソフトウェア更新を使ったパッケージ更新

PackageKit を使用したパッケージの管理, PackageKit
PackageKit を使用したパッケージの表示, PackageKit
PackageKit を使用したパッケージ設定
確認頻度, ソフトウェア更新を使ったパッケージ更新

PackageKit を使用したフィルター, フィルターを使ったパッケージ検索
インストール済み, フィルターを使ったパッケージ検索
インストール済みのみ, フィルターを使ったパッケージ検索
エンドユーザーファイルのみ, フィルターを使ったパッケージ検索
グラフィカルのみ, フィルターを使ったパッケージ検索
サブパッケージを隠す, フィルターを使ったパッケージ検索
ネイティブパッケージのみ, フィルターを使ったパッケージ検索
フィルターなし, フィルターを使ったパッケージ検索
利用可能のみ, フィルターを使ったパッケージ検索
最新パッケージのみ, フィルターを使ったパッケージ検索
未使用, フィルターを使ったパッケージ検索
開発, フィルターを使ったパッケージ検索
開発のみ, フィルターを使ったパッケージ検索

PackageKit を使用した追加と削除, ソフトウェアの追加/削除の使用
perf
ファームウェアファイル, カーネルパッケージの概要

RPM, RPM
すでにインストール, パッケージがすでにインストールされている場合
アンインストール, アンインストール
クエリ, クエリ
ソースおよびバイナリーパッケージ, RPM
ヒント, 実用的で一般的な RPM の使用例
依存関係のエラー, 未解決の依存
削除, アンインストール
最新の状態にする, 最新の状態にする
検証, 検証
競合, ファイルの競合
純粋なソース, RPM の設計目標
設定ファイルの変更, 設定ファイルの変更

RPM のアップグレード, インストールとアップグレード
RPM のインストール, インストールとアップグレード
RPM の代わりに Yum, RPM
RPM パッケージの検索, RPM パッケージの検索
Yum を使用したインストール, パッケージのインストール
Yum を使用したパッケージのアンインストール, パッケージの削除
yum remove package_name, パッケージの削除

Yum を使用したパッケージの一覧表示
yum grouplist, パッケージの一覧表示
yum list all, パッケージの一覧表示

yum list available, パッケージの一覧表示
yum list installed, パッケージの一覧表示
yum repolist, パッケージの一覧表示
yum search, パッケージの一覧表示

yum を使用したパッケージの一覧表示
glob 表現, パッケージの一覧表示

Yum を使用したパッケージの検索
yum search, パッケージの検索

Yum を使用したパッケージの表示
yum info, パッケージ情報の表示

Yum を使用したパッケージグループのインストール, パッケージのインストール
Yum を使用したパッケージグループの削除, パッケージの削除
アンインストール済みのパッケージのクエリ, 実用的で一般的な RPM の使用例
カーネル
シングル、マルチコア、及びマルチプロセッサのシステム対応, カーネルパッケージの概要

カーネル RPM, 手動のカーネルアップグレード
トランザクションログの表示, トランザクションログの表示
ドキュメンテーションの場所の特定, 実用的で一般的な RPM の使用例
パッケージとパッケージグループ, パッケージとパッケージグループ
パッケージの表示
yum info, パッケージ情報の表示

パッケージグループのインストールと削除, パッケージグループのインストールと削除
ファイルの所有権の決定, 実用的で一般的な RPM の使用例
ファイル一覧の取得, 実用的で一般的な RPM の使用例
依存関係, 未解決の依存
初期 RPM リポジトリ, RPM パッケージの検索
削除, アンインストール
削除ファイルの検索, 実用的で一般的な RPM の使用例
現在インストール済みのパッケージの更新
利用可能な更新, ソフトウェア更新を使ったパッケージ更新

パッケージグループのインストール
PackageKit を使用したパッケージグループのインストール, パッケージグループのインストールと削除

パッケージグループの削除
PackageKit を使用したパッケージグループの削除, パッケージグループのインストールと削除

ヒュージページ

設定, /proc/sys/vm/

ファイル、proc ファイルシステム
変更, 仮想ファイルの変更, sysctl コマンドの使用
表示, 仮想ファイルの表示, sysctl コマンドの使用

ファイルシステム, ブロックデバイスとファイルシステムの表示
仮想 (参照 proc ファイルシステム)

フレームバッファーデバイス, /proc/fb
(参照 /proc/fb)

ブロックデバイス, /proc/devices
(参照 /proc/devices)
定義, /proc/devices

ブートメディア, アップグレードへの準備
ブートローダー
確証, ブートローダーの確証

プライマリネームサーバー (参照 BIND)
プリンター (参照 プリンターの設定)
プリンターの設定
CUPS, プリンターの設定
IPP プリンター, IPP プリンターの追加
LDP/LPR プリンター, LPD/LPR Host or Printer の追加
Samba プリンター, Samba (SMB) プリンターの追加
セッティング, 設定のページ
プリンターの共有, プリンターの共有
ローカルプリンター, ローカルプリンターの追加
印刷ジョブ, 印刷ジョブの管理
新規のプリンター, プリンター設定の開始

プロセス, システムプロセスの表示
ボンディング (参照 チャンネルボンディング)
マルチホーム DHCP
サーバーの設定, マルチホーム DHCP サーバーの設定
ホストの設定, ホストの設定

メモリ使用量, メモリ使用量の表示
メールユーザーエージェント, MTA の設定 (参照 電子メール)
メール転送エージェント (参照 MTA) (参照 電子メール)
メール配信エージェント (参照 電子メール)
モジュール (参照 カーネルモジュール)
モジュールパラメーター (参照 カーネルモジュール)

ユーザー (参照 ユーザー設定)
UID, ユーザーとグループの管理
その他のリソース, その他のリソース
インストールされているドキュメント, インストールされているドキュメント

以下を管理するツール
useradd, コマンドラインツールの使用
ユーザー管理, コマンドラインツールの使用

導入, ユーザーとグループの管理

ユーザープライベートグループ (参照 グループ)
及び共有ディレクトリ, グループディレクトリの作成

ユーザー管理 (参照 ユーザー設定)
ユーザー設定
コマンドラインの設定
passwd, 新規ユーザーの追加
useradd, 新規ユーザーの追加

パスワードの変更, ユーザープロパティの変更
ホームディレクトリの変更, ユーザープロパティの変更
ユーザーの一覧をフィルター, ユーザーとグループの表示
ユーザーの一覧を表示, ユーザー管理ツールの使用
ユーザーの修正, ユーザープロパティの変更
ユーザーの追加, 新規ユーザーの追加
ユーザー用のグループ修正, ユーザープロパティの変更
ログインシェルの変更, ユーザープロパティの変更
氏名の変更, ユーザープロパティの変更

リソースレコード (参照 BIND)
ログビューア
フィルタリング, ログファイルの表示
モニタリング, ログファイルのモニタリング
更新間隔, ログファイルの表示
検索, ログファイルの表示

ログファイル, ログファイルの表示と管理
(参照 ログビューア)
rsyslogd デーモン, ログファイルの表示と管理
その他のリソース
インストールされているドキュメント, インストールされているドキュメント
役立つ Web サイト, 役立つ Web サイト

モニタリング, ログファイルのモニタリング

ローテーション, ログファイルの場所の特定
場所の特定, ログファイルの場所の特定
表示, ログファイルの表示
説明, ログファイルの表示と管理

仮想ファイル (参照 proc ファイルシステム)
仮想ファイルシステム (参照 proc ファイルシステム)
仮想ホスト (参照 Apache HTTP サーバー)
再帰ネームサーバー (参照 BIND)
初期 RAM ディスクイメージ
検証, 初期 RAM ディスクイメージの検証
確証
IBM eServer System i, 初期 RAM ディスクイメージの検証

初期 RPM リポジトリ
インストール可能なパッケージ, RPM パッケージの検索

動的ホスト構成プロトコル (参照 DHCP)
完全修飾ドメイン名, ネームサーバーのゾーン
実行ドメイン, /proc/execdomains
(参照 /proc/execdomains)
定義, /proc/execdomains

情報
システム情報, システムモニタリングツール

日付の設定
date, 日付と時刻の設定
system-config-date, 日付と時刻のプロパティ

時刻の設定
date, 日付と時刻の設定
NTP サーバーと同期する, ネットワーク時刻プロトコルのプロパティ, ネットワーク時刻プロトコルの設定
system-config-date, 日付と時刻のプロパティ

権威ネームサーバー (参照 BIND)
現在インストール済みのパッケージの更新
利用可能な更新, ソフトウェア更新を使ったパッケージ更新

自動化タスク, システムタスクの自動化
設定ファイルの変更, 設定ファイルの変更の保存
認証
スマートカード認証の使用, スマートカード認証の有効化
指紋サポートの使用, 指紋認証の使用

認証設定ツール , システム認証の設定

認証設定ツール
及び Kerberos 認証, LDAP または NIS 認証で Kerberos の使用
及び LDAP, LDAP 認証の設定
及び NIS, NIS 認証の設定
及び Winbind, Winbind 認証の設定
及び Winbind 認証, Winbind 認証の設定

追加
グループ, 新規グループの追加
ユーザー, 新規ユーザーの追加

電子メール
Fetchmail, Fetchmail
Postfix, Postfix
Procmail, メール配信エージェント(Mail Delivery Agents)(MDA)
Sendmail, Sendmail
その他のリソース, その他のリソース
インストールされているドキュメント, インストールされているドキュメント
役立つ Web サイト, 役立つ Web サイト
関連書籍, 関連資料

その歴史, メールサーバー
スパム
フィルターによる除去, スパムフィルター

セキュリティ, 通信のセキュリティ保護
クライアント, セキュアな電子メールクライアント
サーバー, 電子メールクライアントの通信のセキュリティ保護

タイプ
メールユーザーエージェント, メールユーザーエージェント (Mail User Agent)
メール転送エージェント, メール転送エージェント (Mail Transport Agent)
メール配信エージェント, メール配信エージェント (Mail Delivery Agent)

プログラムの分類, 電子メールプログラムの分類
プロトコル, 電子メールプロトコル
IMAP, IMAP
POP, POP
SMTP, SMTP

メールサーバー
Dovecot, Dovecot

静的ルート, 静的ルートおよびデフォルトゲートウェイ

(参照 OProfile)

A

anacron, Cron と Anacron
- anacron 設定ファイル, Anacron ジョブの設定
- ユーザー定義のタスク, Anacron ジョブの設定

anacrontab , Anacron ジョブの設定
Apache HTTP サーバー
- SSL サーバー
  - パブリックキー, 証明書とセキュリティの概要
  - プライベートキー, 証明書とセキュリティの概要, 既存のキーと証明書の使用, 新しいキーと証明書の生成
  - 証明書, 証明書とセキュリティの概要, 既存のキーと証明書の使用, 新しいキーと証明書の生成
  - 認証局, 証明書とセキュリティの概要
- その他のリソース
  - インストールされているドキュメント, インストールされているドキュメント
  - 役立つ Web サイト, 役立つ Web サイト
- ステータスの確認, サービスステータスの確認
- ディレクティブ
  - <;Directory> , 一般的な httpd.conf の指示文
  - <IfDefine> , 一般的な httpd.conf の指示文
  - <IfModule> , 一般的な httpd.conf の指示文
  - <Location> , 一般的な httpd.conf の指示文
  - <Proxy> , 一般的な httpd.conf の指示文
  - AccessFileName , 一般的な httpd.conf の指示文
  - Action , 一般的な httpd.conf の指示文
  - AddDescription , 一般的な httpd.conf の指示文
- ディレクトリ
  - /etc/httpd/ , 一般的な httpd.conf の指示文
  - /etc/httpd/conf.d/ , 設定ファイルの編集, 一般的な httpd.conf の指示文
  - /usr/lib/httpd/modules/ , 一般的な httpd.conf の指示文, モジュールでの作業
  - /usr/lib64/httpd/modules/ , 一般的な httpd.conf の指示文, モジュールでの作業
  - /var/cache/mod_proxy/ , 一般的な httpd.conf の指示文
  - /var/www/cgi-bin/ , 一般的な httpd.conf の指示文
  - /var/www/html/ , 一般的な httpd.conf の指示文
  - ~/public_html/ , 一般的な httpd.conf の指示文
- バージョン 2.2
  - バージョン 2.0 から更新, 設定の更新

変更, 注目すべき変更
機能, 新しい機能

ファイル
.htaccess , 一般的な httpd.conf の指示文
.htpasswd, 一般的な httpd.conf の指示文
/etc/httpd/conf.d/ssl.conf , 一般的な ssl.conf 指示文, mod_ssl モジュールを有効にする
/etc/httpd/conf/httpd.conf , 設定ファイルの編集, 一般的な httpd.conf の指示文, 一般的な Multi-Processing Module 指示文
/etc/httpd/logs/access_log , 一般的な httpd.conf の指示文
/etc/httpd/logs/error_log , 一般的な httpd.conf の指示文
/etc/httpd/run/httpd.pid , 一般的な httpd.conf の指示文
/etc/mime.types , 一般的な httpd.conf の指示文

モジュール
mod_asis, 注目すべき変更
mod_cache, 新しい機能
mod_cern_meta, 注目すべき変更
mod_disk_cache, 新しい機能
mod_ext_filter, 注目すべき変更
mod_proxy_balancer, 新しい機能
mod_rewrite , 一般的な httpd.conf の指示文
mod_ssl , SSL サーバーのセットアップ
mod_userdir, 設定の更新
ロード, モジュールのロード
開発, モジュールの書き込み

仮想ホスト, 仮想ホストのセットアップ
停止, サービスの停止
再開始, サービスの再開始
指示文
/var/www/icons/ , 一般的な httpd.conf の指示文
<VirtualHost> , 一般的な httpd.conf の指示文
AddEncoding , 一般的な httpd.conf の指示文
AddHandler , 一般的な httpd.conf の指示文
AddIcon , 一般的な httpd.conf の指示文
AddIconByEncoding , 一般的な httpd.conf の指示文
AddIconByType , 一般的な httpd.conf の指示文
AddLanguage , 一般的な httpd.conf の指示文
AddType , 一般的な httpd.conf の指示文
Alias , 一般的な httpd.conf の指示文
Allow , 一般的な httpd.conf の指示文
AllowOverride , 一般的な httpd.conf の指示文

BrowserMatch , 一般的な httpd.conf の指示文
CacheDefaultExpire , 一般的な httpd.conf の指示文
CacheDisable , 一般的な httpd.conf の指示文
CacheEnable , 一般的な httpd.conf の指示文
CacheLastModifiedFactor , 一般的な httpd.conf の指示文
CacheMaxExpire , 一般的な httpd.conf の指示文
CacheNegotiatedDocs , 一般的な httpd.conf の指示文
CacheRoot , 一般的な httpd.conf の指示文
CustomLog , 一般的な httpd.conf の指示文
DefaultIcon , 一般的な httpd.conf の指示文
DefaultType , 一般的な httpd.conf の指示文
Deny , 一般的な httpd.conf の指示文
DirectoryIndex , 一般的な httpd.conf の指示文
DocumentRoot , 一般的な httpd.conf の指示文
ErrorDocument , 一般的な httpd.conf の指示文
ExtendedStatus , 一般的な httpd.conf の指示文
Group , 一般的な httpd.conf の指示文
HeaderName , 一般的な httpd.conf の指示文
HostnameLookups , 一般的な httpd.conf の指示文
Include , 一般的な httpd.conf の指示文
IndexIgnore , 一般的な httpd.conf の指示文
IndexOptions , 一般的な httpd.conf の指示文
KeepAlive , 一般的な httpd.conf の指示文
KeepAliveTimeout , 一般的な httpd.conf の指示文
LanguagePriority , 一般的な httpd.conf の指示文
Listen , 一般的な httpd.conf の指示文
LoadModule , 一般的な httpd.conf の指示文
LogFormat , 一般的な httpd.conf の指示文
LogLevel , 一般的な httpd.conf の指示文
MaxClients , 一般的な Multi-Processing Module 指示文
MaxKeepAliveRequests , 一般的な httpd.conf の指示文
MaxSpareServers , 一般的な Multi-Processing Module 指示文
MaxSpareThreads , 一般的な Multi-Processing Module 指示文
MinSpareServers , 一般的な Multi-Processing Module 指示文
MinSpareThreads , 一般的な Multi-Processing Module 指示文
NameVirtualHost , 一般的な httpd.conf の指示文
Options , 一般的な httpd.conf の指示文
Order , 一般的な httpd.conf の指示文
PidFile , 一般的な httpd.conf の指示文
ProxyRequests , 一般的な httpd.conf の指示文
ReadmeName , 一般的な httpd.conf の指示文
Redirect , 一般的な httpd.conf の指示文

ScriptAlias , 一般的な httpd.conf の指示文
ServerAdmin , 一般的な httpd.conf の指示文
ServerName , 一般的な httpd.conf の指示文
ServerRoot , 一般的な httpd.conf の指示文
ServerSignature , 一般的な httpd.conf の指示文
ServerTokens , 一般的な httpd.conf の指示文
SetEnvIf , 一般的な ssl.conf 指示文
StartServers , 一般的な Multi-Processing Module 指示文
SuexecUserGroup , 一般的な httpd.conf の指示文
ThreadsPerChild , 一般的な Multi-Processing Module 指示文
Timeout , 一般的な httpd.conf の指示文
TypesConfig , 一般的な httpd.conf の指示文
UseCanonicalName , 一般的な httpd.conf の指示文
User , 一般的な httpd.conf の指示文
UserDir , 一般的な httpd.conf の指示文

設定の確認, 設定ファイルの編集
開始, サービスの開始

Apache HTTP サーバー
指示文
ErrorLog , 一般的な httpd.conf の指示文

at , At コマンドと Batch コマンド
その他のリソース, その他のリソース

authconfig (参照 認証設定ツール)
コマンド, コマンドライン上で 認証の設定

B

batch , At コマンドと Batch コマンド
その他のリソース, その他のリソース

Berkeley Internet Name Domain (参照 BIND)
BIND
その他のリソース
インストールされているドキュメント, インストールされているドキュメント
役立つ Web サイト, 役立つ Web サイト
関連書籍, 関連資料

ゾーン
$INCLUDE 指示文, 一般的な指示文
$ORIGIN 指示文, 一般的な指示文

$TTL 指示文, 一般的な指示文
A (Address) リソースレコード, 一般的なリソースレコード
CNAME (Canonical Name) リソースレコード, 一般的なリソースレコード
MX (Mail Exchange) リソースレコード, 一般的なリソースレコード
NS (Nameserver) リソースレコード, 一般的なリソースレコード
PTR (Pointer) リソースレコード, 一般的なリソースレコード
SOA (Start of Authority) リソースレコード, 一般的なリソースレコード
コメントタグ, コメントタグ
使用法の例, 単純なゾーンファイル, 逆引き名前解決ゾーンファイル
詳細, ネームサーバーのゾーン

タイプ
セカンダリ (スレーブ) ネームサーバー, ネームサーバーのゾーン, ネームサーバーのタイプ
プライマリ (マスター) ネームサーバー, ネームサーバーのゾーン, ネームサーバーのタイプ
再帰ネームサーバー, ネームサーバーのタイプ
権威ネームサーバー, ネームサーバーのタイプ

ディレクトリ
/etc/named/ , named サービスの設定
/var/named/ , ゾーンファイルの編集
/var/named/data/ , ゾーンファイルの編集
/var/named/dynamic/ , ゾーンファイルの編集
/var/named/slaves/ , ゾーンファイルの編集

ファイル
/etc/named.conf , named サービスの設定, ユーティリティの設定
/etc/rndc.conf , ユーティリティの設定
/etc/rndc.key , ユーティリティの設定

ユーティリティ
dig, ネームサーバーとしての BIND, dig ユーティリティの使用, DNSSEC (DNS Security Extensions)
named, ネームサーバーとしての BIND, named サービスの設定
rndc, ネームサーバーとしての BIND, rndc ユーティリティの使用

リソースレコード, ネームサーバーのゾーン
一般的な間違い, 回避すべき一般的な間違い
機能
Automatic Zone Transfer (AXFR), IXFR (Incremental Zone Transfers 差分ゾーン転送)
DNS Security Extensions (DNSSEC), DNSSEC (DNS Security Extensions)
Incremental Zone Transfer (IXFR), IXFR (Incremental Zone Transfers 差分ゾーン転送)
Internet Protocol version 6 (IPv6), インターネットプロトコルバージョン 6 (IPv6)
Transaction SIGnature (TSIG), Transaction SIGnatures トランザクション署名 (TSIG)
複数表示, 複数表示

設定
acl ステートメント, 一般的なステートメントのタイプ
controls ステートメント, その他のステートメントタイプ
include ステートメント, 一般的なステートメントのタイプ
key ステートメント, その他のステートメントタイプ
logging ステートメント, その他のステートメントタイプ
options ステートメント, 一般的なステートメントのタイプ
server ステートメント, その他のステートメントタイプ
trusted-keys ステートメント, その他のステートメントタイプ
view ステートメント, その他のステートメントタイプ
zone ステートメント, 一般的なステートメントのタイプ
コメントタグ, コメントタグ

blkid, blkid コマンドの使用

C

ch-email .fetchmailrc
グローバルオプション, グローバルオプション

chkconfig (参照 サービスの設定)
CPU 使用率, CPU 使用率の表示
crash
システム要件, コアダンプの分析
ダンプの分析
オープンファイル, オープンファイルの表示
スタックトレース, バックトレースの表示
プロセス, プロセスステータスの表示
メッセージバッファー, メッセージバッファーの表示
仮想メモリ, 仮想メモリ情報の表示

ダンプイメージを開く, crash ユーティリティの実行

createrepo, Yum リポジトリの作成
cron, Cron と Anacron
cron 設定ファイル, Cron ジョブの設定
その他のリソース, その他のリソース
ユーザー定義のタスク, Cron ジョブの設定

crontab , Cron ジョブの設定
CUPS (参照 プリンターの設定)

D

date (参照 日付の設定)

df, df コマンドの使用
DHCP, DHCP サーバー
    dhcpd.conf, 設定ファイル
    dhcpd.leases, サーバーの起動と停止
    dhcpd6.conf, IPv6 の DHCP (DHCPv6)
    DHCPv6, IPv6 の DHCP (DHCPv6)
    dhcrelay, DHCP リレーエージェント
    shared-network, 設定ファイル
    subnet, 設定ファイル
    その他のリソース, その他のリソース
    オプション, 設定ファイル
    クライアントの設定, DHCP クライアントの設定
    グループ, 設定ファイル
    グローバルパラメーター, 設定ファイル
    コマンドラインオプション, サーバーの起動と停止
    サーバーの停止, サーバーの起動と停止
    サーバーの設定, DHCP サーバーの設定
    サーバーの起動, サーバーの起動と停止
    リレーエージェント, DHCP リレーエージェント
    使用する理由, DHCP を使用する理由
    接続, DHCP クライアントの設定

dhcpd.conf, 設定ファイル
dhcpd.leases, サーバーの起動と停止
dhcrelay, DHCP リレーエージェント
dig (参照 BIND)
DNS
    定義, DNS サーバー
        (参照 BIND)

DoS 攻撃 (参照 サービス拒否攻撃)
DSA 鍵
    生成, 鍵ペアの生成

du, du コマンドの使用

E

Enterprise Linux 用エキストラパッケージ (EPEL)
    インストール可能なパッケージ, RPM パッケージの検索

epoch
    定義, /proc/stat

Ethtool
オプション
--advertise , Ethtool
--autoneg , Ethtool
--duplex , Ethtool
--identify , Ethtool
--msglvl , Ethtool
--phyad , Ethtool
--port , Ethtool
--sopass , Ethtool
--speed , Ethtool
--statistics , Ethtool
--test , Ethtool
--wol , Ethtool
--xcvr , Ethtool
コマンド
devname , Ethtool

exec-shield
有効化, /proc/sys/kernel/
概要, /proc/sys/kernel/

F

Fetchmail, Fetchmail
その他のリソース, その他のリソース
コマンドオプション, Fetchmail のコマンドオプション
情報提供, 情報提供またはデバッグのオプション
特別, 特別なオプション
設定オプション, Fetchmail の設定オプション
グローバルオプション, グローバルオプション
サーバーオプション, サーバーオプション
ユーザーオプション, ユーザーオプション

findmnt, findmnt コマンドの使用
findsmb, コマンドライン
findsmb プログラム, Samba ディストリビューションプログラム
FQDN (参照 完全修飾ドメイン名)
free, free コマンドの使用
FTP, FTP
(参照 vsftpd)

アクティブモード, File Transfer Protocol (ファイル転送プロトコル)
コマンドポート, File Transfer Protocol (ファイル転送プロトコル)
データポート, File Transfer Protocol (ファイル転送プロトコル)
パッシブモード, File Transfer Protocol (ファイル転送プロトコル)
定義, FTP
紹介, File Transfer Protocol (ファイル転送プロトコル)

G

GNOME, デスクトップ環境
(参照 X)

gnome-system-log (参照 ログビューア)
gnome-system-monitor, システムモニターツールの使用, システムモニターツールの使用, システムモニターツールの使用, システムモニターツールの使用
GnuPG
RPM パッケージの署名確認, パッケージの署名確認

GRUB ブートローダー
設定, GRUB ブートローダーの設定
設定ファイル, GRUB ブートローダーの設定

H

HTTP サーバー (参照 Apache HTTP サーバー)
httpd (参照 Apache HTTP サーバー)

I

ifdown , インターフェース制御スクリプト
ifup , インターフェース制御スクリプト
insmod, モジュールのロード
(参照 カーネルモジュール)

K

KDE, デスクトップ環境
(参照 X)

kdump
その他のリソース
man ページ, インストールされているドキュメント
Web サイト, 役立つ Web サイト
インストールされているドキュメント, インストールされているドキュメント

インストール, kdump サービスのインストール

サービスの実行, サービスの有効化
サービスの有効化, サービスの有効化, サービスの有効化, サービスの有効化
サービスの設定
カーネルイメージ, エキスパートセッティング タブ, メモリー使用量の設定
カーネルオプション, エキスパートセッティング タブ, メモリー使用量の設定
サポートされているターゲット, ターゲットセッティング タブ, ターゲットタイプの設定
ターゲットロケーション, ターゲットセッティング タブ, ターゲットタイプの設定
ダンプイメージの圧縮, エキスパートセッティング タブ, コアコレクターの設定
デフォルトの動作, エキスパートセッティング タブ, デフォルトの動作の変更
フィルタリングレベル, フィルタリングセッティング タブ, コアコレクターの設定
メモリ使用量, メモリー使用量の設定, 基本設定タブ, メモリー使用量の設定
初期 RAM ディスク, エキスパートセッティング タブ, メモリー使用量の設定

システム要件, kdump サービスの設定
ダンプの分析 (参照 crash)
設定のテスト, 設定のテスト

kwin, ウィンドウマネージャー
(参照 X)

L

LDAP (参照 OpenLDAP)
logrotate, ログファイルの場所の特定
lsblk, lsblk コマンドの使用
lscpu, lscpu コマンドの使用
lsmod, 現在ロード済みモジュールの一覧表示
(参照 カーネルモジュール)

lspci, lspci コマンドの使用, /proc/bus/pci
lspcmcia, lspcmcia コマンドの使用
lsusb, lsusb コマンドの使用

M

Mail Transport Agent Switcher, MTA の設定
MDA (参照 メール配信エージェント)
metacity, ウィンドウマネージャー
(参照 X)

modinfo, モジュール情報の表示
(参照 カーネルモジュール)

modprobe, モジュールのロード, モジュールのアンロード
(参照 カーネルモジュール)

MTA (参照 メール転送エージェント)
Mail Transport Agent Switcher で切り替え, MTA の設定
デフォルトの設定, MTA の設定

MUA, MTA の設定 (参照 メールユーザーエージェント)
mwm, ウィンドウマネージャー
(参照 X)

N

named (参照 BIND)
nameserver (参照 DNS)
net プログラム, Samba ディストリビューションプログラム
NIC
単一のチャンネルにバインド, チャンネルボンディングの使用

nmblookup プログラム, Samba ディストリビューションプログラム
NSCD
及び SSSD, SSSD で NSCD の使用

NTP
ntpd, ネットワーク時刻プロトコルのプロパティ, ネットワーク時刻プロトコルの設定
ntpdate, ネットワーク時刻プロトコルの設定
設定, ネットワーク時刻プロトコルのプロパティ, ネットワーク時刻プロトコルの設定

ntpd (参照 NTP)
ntpdate (参照 NTP)
ntsysv (参照 サービスの設定)

O

opannotate (参照 OProfile)
opcontrol (参照 OProfile)
OpenLDAP
インストール, OpenLDAP スイートのインストール
クライアントアプリケーション, 一般的な LDAP クライアントアプリケーションの概要
スキーマ, スキーマの拡張
ディレクトリ
/etc/openldap/slapd.d/, OpenLDAP サーバーの設定
/etc/openldap/slapd.d/cn=config/cn=schema/, スキーマの拡張

パッケージ, OpenLDAP スイートのインストール
ファイル
/etc/openldap/ldap.conf, OpenLDAP サーバーの設定
/etc/openldap/slapd.d/cn=config.ldif, グローバル設定の変更

/etc/openldap/slapd.d/cn=config/olcDatabase={1}bdb.ldif, データベース特有の設定の変更

ユーティリティ, OpenLDAP サーバーユーティリティの概要, OpenLDAP クライアントユーティリティの概要
停止, サービスの停止
再起動, サービスの再開始
実行, サービスの開始
指示文
olcAllows, グローバル設定の変更
olcConnMaxPending, グローバル設定の変更
olcConnMaxPendingAuth, グローバル設定の変更
olcDisallows, グローバル設定の変更
olcIdleTimeout, グローバル設定の変更
olcLogFile, グローバル設定の変更
olcReadOnly, データベース特有の設定の変更
olcReferral, グローバル設定の変更
olcRootDN, データベース特有の設定の変更
olcRootPW, データベース特有の設定の変更
olcSuffix, データベース特有の設定の変更
olcWriteTimeout, グローバル設定の変更

機能, OpenLDAP の機能
状態の確認, サービスステータスの確認
用語
LDIF, LDAP の用語
エントリ, LDAP の用語
属性, LDAP の用語

設定
グローバル, グローバル設定の変更
データベース, データベース特有の設定の変更
概要, OpenLDAP サーバーの設定

認証情報の移行, 旧認証情報を LDAP 形式に移行

OpenSSH, OpenSSH, 主要な機能
(参照 SSH)
DSA 鍵
生成, 鍵ペアの生成

RSA バージョン 1 の鍵
生成, 鍵ペアの生成

RSA 鍵
生成, 鍵ペアの生成

ssh-add, ssh-agent の設定
ssh-agent, ssh-agent の設定
ssh-keygen
DSA, 鍵ペアの生成
RSA, 鍵ペアの生成
RSA バージョン 1, 鍵ペアの生成

その他のリソース, その他のリソース
クライアント, OpenSSH クライアント
scp, scp ユーティリティの使用
sftp, sftp ユーティリティの使用
ssh, ssh ユーティリティの使用

サーバー, OpenSSH サーバーの起動
停止, OpenSSH サーバーの起動
起動, OpenSSH サーバーの起動

鍵ベース認証の使用, 鍵ベース認証の使用

OpenSSL
SSL (参照 SSL)
TLS (参照 TLS)
その他のリソース, その他のリソース

ophelp, 監視するイベントのセッティング
opreport (参照 OProfile)
OProfile, OProfile
/dev/oprofile/, /dev/oprofile/ を理解する
Java, Java 対応の OProfile サポート
opannotate, opannotate の使用
opcontrol, OProfile の設定
--no-vmlinux, カーネルの指定
--start, OProfile の開始と停止
--vmlinux=, カーネルの指定

ophelp, 監視するイベントのセッティング
opreport, opreport の使用, モジュールについてより詳細な出力を取得
単独の実行可能ファイルで, 単独実行可能ファイルで opreport の使用

oprofiled, OProfile の開始と停止
ログファイル, OProfile の開始と停止

SystemTap, OProfile と SystemTap
その他のリソース, その他のリソース
イベント

サンプルのレート, サンプルのレート
セッティング, 監視するイベントのセッティング

カーネルの監視, カーネルの指定
ツールの概要, ツールの概要
データの保存, データの保存
データの読み込み, データの分析
ユニットマスク, ユニットマスク
設定, OProfile の設定
プロファイルの分離, カーネルとユーザースペースのプロファイルの分離

開始, OProfile の開始と停止

oprofiled (参照 OProfile)
oprof_start, グラフィカルインターフェース
OS/400 ブートローダー
設定, OS/400 ブートローダーの設定
設定ファイル, OS/400 ブートローダーの設定

P

PackageKit, PackageKit
PolicyKit
認証, ソフトウェア更新を使ったパッケージ更新

アーキテクチャ, PackageKit アーキテクチャ
トランザクションログの表示, トランザクションログの表示
パッケージのアンインストール, PackageKit
パッケージのインストール, PackageKit
パッケージの更新, PackageKit
パッケージの管理, PackageKit
パッケージの表示, PackageKit
パッケージグループのインストールと削除, パッケージグループのインストールと削除
追加と削除, ソフトウェアの追加/削除の使用

PackageKit を使用したパッケージの更新
PolicyKit, ソフトウェア更新を使ったパッケージ更新

pdbedit プログラム, Samba ディストリビューションプログラム
PolicyKit, ソフトウェア更新を使ったパッケージ更新
Postfix, Postfix
デフォルトインストール, Postfix のデフォルトインストール

postfix, MTA の設定
prefdm (参照 X)

proc ファイルシステム
/proc/buddyinfo, /proc/buddyinfo
/proc/bus/ ディレクトリ, /proc/bus/
/proc/bus/pci
lspci を使用した表示, /proc/bus/pci
/proc/cmdline, /proc/cmdline
/proc/cpuinfo, /proc/cpuinfo
/proc/crypto, /proc/crypto
/proc/devices
キャラクターデバイス, /proc/devices
ブロックデバイス, /proc/devices
/proc/dma, /proc/dma
/proc/driver/ ディレクトリ, /proc/driver/
/proc/execdomains, /proc/execdomains
/proc/fb, /proc/fb
/proc/filesystems, /proc/filesystems
/proc/fs/ ディレクトリ, /proc/fs
/proc/interrupts, /proc/interrupts
/proc/iomem, /proc/iomem
/proc/ioports, /proc/ioports
/proc/irq/ ディレクトリ, /proc/irq/
/proc/kcore, /proc/kcore
/proc/kmsg, /proc/kmsg
/proc/loadavg, /proc/loadavg
/proc/locks, /proc/locks
/proc/mdstat, /proc/mdstat
/proc/meminfo, /proc/meminfo
/proc/misc, /proc/misc
/proc/modules, /proc/modules
/proc/mounts, /proc/mounts
/proc/mtrr, /proc/mtrr
/proc/net/ ディレクトリ, /proc/net/
/proc/partitions, /proc/partitions
/proc/PID/ ディレクトリ, /proc/PID/
/proc/scsi/ ディレクトリ, /proc/scsi/
/proc/self/ ディレクトリ, /proc/self/
/proc/slabinfo, /proc/slabinfo
/proc/stat, /proc/stat
/proc/swaps, /proc/swaps
/proc/sys/ ディレクトリ, /proc/sys/, sysctl コマンドの使用

(参照 sysctl)
/proc/sys/dev/ ディレクトリ, /proc/sys/dev/
/proc/sys/fs/ ディレクトリ, /proc/sys/fs/
/proc/sys/kernel/ ディレクトリ, /proc/sys/kernel/
/proc/sys/kernel/exec-shield, /proc/sys/kernel/
/proc/sys/kernel/sysrq (参照 システム要求キー)
/proc/sys/net/ ディレクトリ, /proc/sys/net/
/proc/sys/vm/ ディレクトリ, /proc/sys/vm/

/proc/sysrq-trigger, /proc/sysrq-trigger
/proc/sysvipc/ ディレクトリ, /proc/sysvipc/
/proc/tty/ ディレクトリ, /proc/tty/
/proc/uptime, /proc/uptime
/proc/version, /proc/version
subdirectories within, /proc/ 内のディレクトリ
その他のリソース, その他のリソース
インストールされているドキュメント, インストールされているドキュメント
役立つ Web サイト, 役立つ Web サイト

ファイル、最上位, proc ファイルシステム内の最上位のファイル
ファイルの変更, 仮想ファイルの変更, /proc/sys/, sysctl コマンドの使用
ファイルの表示, 仮想ファイルの表示
プロセスディレクトリ, プロセスディレクトリ

Procmail, メール配信エージェント(Mail Delivery Agents)(MDA)
その他のリソース, その他のリソース
レシピ, Procmail のレシピ
SpamAssassin, スパムフィルター
フラグ, フラグ
ローカルロックファイル, ローカルロックファイルの指定
例, レシピの例
特別なアクション, 特別な条件とアクション
特別な条件, 特別な条件とアクション
配信, 配信レシピと非配信レシピの比較
非配信, 配信レシピと非配信レシピの比較

設定, Procmail の設定

procファイルシステム
概要, proc ファイルシステム

ps, ps コマンドの使用

R

RAM, メモリ使用量の表示
rcp, scp ユーティリティの使用
Red Hat Enterprise Linux インストールメディア
    インストール可能なパッケージ, RPM パッケージの検索

Red Hat サブスクリプションマネージャー, Red Hat サブスクリプションマネージャーツールの使用
rmmod, モジュールのアンロード
    (参照 カーネルモジュール)

rndc (参照 BIND)
root ネームサーバー (参照 BIND)
rpcclient プログラム, Samba ディストリビューションプログラム
RPM, RPM
    GnuPG, パッケージの署名確認
    md5sum, パッケージの署名確認
    RPM パッケージの検索, RPM パッケージの検索
    Web サイト, 役立つ Web サイト
    すでにインストール, パッケージがすでにインストールされている場合
    その他のリソース, その他のリソース
    アップグレード, インストールとアップグレード
    アンインストール, アンインストール
    アンインストール済みのパッケージのクエリ, 実用的で一般的な RPM の使用例
    インストール, インストールとアップグレード
    クエリ, クエリ
    ドキュメンテーション, 実用的で一般的な RPM の使用例
    パッケージの署名確認, パッケージの署名確認
    ヒント, 実用的で一般的な RPM の使用例
    ファイルの所有権の決定, 実用的で一般的な RPM の使用例
    ファイルの競合
        解決, ファイルの競合

    ファイル一覧のクエリ, 実用的で一般的な RPM の使用例
    ファイル名, インストールとアップグレード
    ベーシックモード, RPM の使用法
    依存関係, 未解決の依存
    依存関係のエラー, 未解決の依存
    削除ファイルの検索, 実用的で一般的な RPM の使用例
    最新の状態にする, 最新の状態にする
    検証, 検証
    競合, ファイルの競合
    設定ファイルの変更, 設定ファイルの変更
        conf.rpmsave, 設定ファイルの変更

設計目標, RPM の設計目標
アップグレードの可能性, RPM の設計目標
システムの検証, RPM の設計目標
強力なクエリ, RPM の設計目標

関連書籍, 関連資料

RPM パッケージマネージャ (参照 RPM)
RSA バージョン 1 の鍵
生成, 鍵ペアの生成

RSA 鍵
生成, 鍵ペアの生成

rsyslog, ログファイルの表示と管理
runlevel (参照 サービスの設定)

S

Samba (参照 Samba)
CUPS 印刷サポート, CUPS 印刷サポートを実装した Samba
CUPS smb.conf, smb.conf の簡易設定

findsmb, コマンドライン
Samba プリンター, Samba (SMB) プリンターの追加
smb.conf, Samba サーバーのタイプと smb.conf ファイル
Active Directory を使用する PDC, Active Directory を使用する Primary Domain Controller (PDC)
Active Directory メンバーサーバーの例, Active Directory ドメインメンバーサーバー
NT4 式 ドメインメンバーサーバーの例, Windows NT4 ベースのドメインメンバーサーバー
PDCtdbsam を使用, tdbsam を使用する Primary Domain Controller (PDC)
セキュアなファイルとプリントサーバーの例, セキュアな読み取り/書き込みファイルとプリントサーバー
匿名のプリントサーバーの例, 匿名のプリントサーバー
匿名の読み取り/書き込みの例, 匿名の読み取り/書き込み
匿名の読み取り専用の例, 匿名の読み取り専用

smbclient, コマンドライン
Windows NT 4.0、2000、ME、及び XP での使用, 暗号化されたパスワード
WINS, WINS (Windows Internet Name Server)
その他のリソース, その他のリソース
インストールされているドキュメント, インストールされているドキュメント
関連書籍, 関連資料

アカウント情報データベース, Samba アカウント情報データベース
ldapsam, Samba アカウント情報データベース

ldapsam_compat, Samba アカウント情報データベース
mysqlsam, Samba アカウント情報データベース
smbpasswd, Samba アカウント情報データベース
tdbsam, Samba アカウント情報データベース
xmlsam, Samba アカウント情報データベース
プレーンテキスト, Samba アカウント情報データベース

グラフィカル設定, グラフィカル設定
サーバーのタイプ, Samba サーバーのタイプと smb.conf ファイル
サーバータイプ
スタンドアロン, スタンドアロンサーバー
ドメインコントローラー, ドメインコントローラー
ドメインメンバー, ドメインメンバーサーバー

サービス
停止, Samba の起動と停止
再読み込み, Samba の起動と停止
再起動, Samba の起動と停止
条件付き再起動, Samba の起動と停止
起動, Samba の起動と停止

セキュリティモード, Samba のセキュリティモード
Active Directory セキュリティモード, Active Directory セキュリティモード (ユーザーレベルのセキュリティ)
サーバーセキュリティモード, サーバーセキュリティモード (ユーザーレベルのセキュリティ)
ドメインセキュリティモード, ドメインセキュリティモード (ユーザーレベルのセキュリティ)
ユーザーレベルのセキュリティ, ユーザーレベルのセキュリティ
共有レベルのセキュリティ, 共有レベルのセキュリティ

デーモン, Samba のデーモンと関連サービス
nmbd, Samba のデーモン
smbd, Samba のデーモン
winbindd, Samba のデーモン
概要, Samba のデーモン

ネットワークブラウジング , Samba のネットワークブラウジング
WINS, WINS (Windows Internet Name Server)
ドメインブラウジング, ドメインブラウジング

ブラウジング, Samba のネットワークブラウジング
プログラム, Samba ディストリビューションプログラム
net, Samba ディストリビューションプログラム
nmblookup, Samba ディストリビューションプログラム
pdbedit, Samba ディストリビューションプログラム

rpcclient, Samba ディストリビューションプログラム
smbcacls, Samba ディストリビューションプログラム
smbclient, Samba ディストリビューションプログラム
smbcontrol, Samba ディストリビューションプログラム
smbpasswd, Samba ディストリビューションプログラム
smbspool, Samba ディストリビューションプログラム
smbstatus, Samba ディストリビューションプログラム
smbtar, Samba ディストリビューションプログラム
testparm, Samba ディストリビューションプログラム
wbinfo, Samba ディストリビューションプログラム

プロフラム
findsmb, Samba ディストリビューションプログラム

下位互換性のあるデータベースバックエンド, Samba アカウント情報データベース
共有
Nautilus を使用した接続, Samba 共有への接続
コマンドラインを使用した接続, コマンドライン
マウント, 共有のマウント

参考資料, Samba
新たなデータベースバックエンド, Samba アカウント情報データベース
暗号化されたパスワード, 暗号化されたパスワード
概要, Samba の概要
機能, Samba の機能
設定, Samba サーバーの設定, コマンドラインからの設定
デフォルト, Samba サーバーの設定

追加のリソース
役立つ Web サイト, 役立つ Web サイト

scp (参照 OpenSSH)
security プラグイン (参照 セキュリティ)
Sendmail, Sendmail
LDAP との併用, LDAP での Sendmail の使用
UUCP を使用, Sendmail の一般的な設定変更
その他のリソース, その他のリソース
エイリアス, マスカレード
スパム, スパムの防止
デフォルトのインストール, Sendmail のデフォルトのインストール
マスカレード, マスカレード
一般的な設定変更, Sendmail の一般的な設定変更
制約, 用途と制約

用途, 用途と制約

sendmail, MTA の設定
service (参照 サービスの設定)
sftp (参照 OpenSSH)
slapd (参照 OpenLDAP)
smbcacls プログラム, Samba ディストリビューションプログラム
smbclient, コマンドライン
smbclient プログラム, Samba ディストリビューションプログラム
smbcontrol プログラム, Samba ディストリビューションプログラム
smbpasswd プログラム, Samba ディストリビューションプログラム
smbspool プログラム, Samba ディストリビューションプログラム
smbstatus プログラム, Samba ディストリビューションプログラム
smbtar プログラム, Samba ディストリビューションプログラム
SpamAssassin
Procmail との併用, スパムフィルター

ssh (参照 OpenSSH)
SSH プロトコル
X11 転送, X11 転送
セキュリティリスク, SSH を使用する理由
セキュリティ保護されていないプロトコル, リモート接続に必要な SSH
バージョン 1, プロトコルのバージョン
バージョン 2, プロトコルのバージョン
ポート転送, ポート転送
リモートログインに必要, リモート接続に必要な SSH
層
チャンネル, チャンネル
トランスポート層, トランスポート層

接続のシーケンス, SSH 接続のイベントシーケンス
機能, 主要な機能
設定ファイル, 設定ファイル
システム全体の設定ファイル, 設定ファイル
ユーザー固有の設定ファイル, 設定ファイル

認証, 認証

ssh-add, ssh-agent の設定
ssh-agent, ssh-agent の設定
SSL , SSL サーバーのセットアップ
(参照 Apache HTTP サーバー)

SSL サーバー (参照 Apache HTTP サーバー)

SSSD
ID プロバイダー
ローカル, sssd.conf ファイルの作成

Kerberos 認証, ドメインの作成: Kerberos 認証
LDAP ドメイン, ドメインの作成: LDAP
サポートされている LDAP ディレクトリ, ドメインの作成: LDAP

Microsoft Active Directory ドメイン, Active Directory ID プロバイダーの設定, LDAP プロバイダーとしての Active Directory の設定
sudo ルール
ホストごとに保存されたルール, sudo、LDAP、SSSD について

プロキシドメイン, ドメインの作成: Proxy
及び NSCD, SSSD で NSCD の使用
設定ファイル
セクション, sssd.conf ファイルの作成
作成, sssd.conf ファイルの設定
場所, カスタム設定ファイルの使用

startx, ランレベル 3 (参照 X)
(参照 X)

stunnel, 電子メールクライアントの通信のセキュリティ保護
sysconfig ディレクトリ
/etc/sysconfig/apm-scripts/ ディレクトリ, /etc/sysconfig/ ディレクトリ内のディレクトリ
/etc/sysconfig/arpwatch, /etc/sysconfig/arpwatch
/etc/sysconfig/authconfig, /etc/sysconfig/authconfig
/etc/sysconfig/autofs, /etc/sysconfig/autofs
/etc/sysconfig/cbq/ ディレクトリ, /etc/sysconfig/ ディレクトリ内のディレクトリ
/etc/sysconfig/clock, /etc/sysconfig/clock
/etc/sysconfig/dhcpd, /etc/sysconfig/dhcpd
/etc/sysconfig/firstboot, /etc/sysconfig/firstboot
/etc/sysconfig/init, /etc/sysconfig/init
/etc/sysconfig/ip6tables-config, /etc/sysconfig/ip6tables-config
/etc/sysconfig/keyboard, /etc/sysconfig/keyboard
/etc/sysconfig/ldap, /etc/sysconfig/ldap
/etc/sysconfig/named, /etc/sysconfig/named
/etc/sysconfig/network, /etc/sysconfig/network
/etc/sysconfig/network-scripts/ ディレクトリ, ネットワークインターフェース, /etc/sysconfig/ ディレクトリ内のディレクトリ
(参照 ネットワーク)

/etc/sysconfig/networking/ ディレクトリ, /etc/sysconfig/ ディレクトリ内のディレクトリ

/etc/sysconfig/ntpd, /etc/sysconfig/ntpd
/etc/sysconfig/quagga, /etc/sysconfig/quagga
/etc/sysconfig/radvd, /etc/sysconfig/radvd
/etc/sysconfig/rhn/ ディレクトリ, /etc/sysconfig/ ディレクトリ内のディレクトリ
/etc/sysconfig/samba, /etc/sysconfig/samba
/etc/sysconfig/saslauthd, /etc/sysconfig/saslauthd
/etc/sysconfig/selinux, /etc/sysconfig/selinux
/etc/sysconfig/sendmail, /etc/sysconfig/sendmail
/etc/sysconfig/spamassassin, /etc/sysconfig/spamassassin
/etc/sysconfig/squid, /etc/sysconfig/squid
/etc/sysconfig/system-config-users, /etc/sysconfig/system-config-users
/etc/sysconfig/vncservers, /etc/sysconfig/vncservers
/etc/sysconfig/xinetd, /etc/sysconfig/xinetd
その他のリソース, その他のリソース
インストールされているドキュメント, インストールされているドキュメント

ディレクトリ, /etc/sysconfig/ ディレクトリ内のディレクトリ
ファイル, /etc/sysconfig/ ディレクトリのファイル
追加情報, sysconfig ディレクトリ

sysctl
/etc/sysctl.conf を使用した設定, sysctl コマンドの使用
/proc/sys/ の制御, sysctl コマンドの使用

SysRq (参照 システム要求キー)
system-config-authentication (参照 認証設定ツール)
system-config-date (参照 日付と時刻の設定)
system-config-kdump (参照 kdump)
system-config-services (参照 サービスの設定)
system-config-users (参照 ユーザー設定とグループ設定)

T

testparm プログラム, Samba ディストリビューションプログラム
TLB キャッシュ (参照 ヒュージページ)
TLS , SSL サーバーのセットアップ
(参照 Apache HTTP サーバー)

top, top コマンドの使用
twm, ウィンドウマネージャー
(参照 X)

U

useradd コマンド

次を使用したユーザーアカウントの作成, 新規ユーザーの追加

V

vsftpd
condrestart, vsftpd の起動と停止
RPM
インストールされたファイル, vsftpd によるファイルのインストール
その他のリソース, その他のリソース
インストールされているドキュメント, インストールされているドキュメント
役に立つ Web サイト, 役に立つ Web サイト
その複数コピーの起動, vsftpd の複数コピーの起動
マルチホーム設定, vsftpd の複数コピーの起動
停止, vsftpd の起動と停止
再起動, vsftpd の起動と停止
状態, vsftpd の起動と停止
設定ファイル
/etc/vsftpd/vsftpd.conf, vsftpd 設定オプション
アクセス制御, ログインオプションとアクセス制御
ディレクトリのオプション, ディレクトリのオプション
デーモンのオプション, デーモンのオプション
ネットワークのオプション, ネットワークのオプション
ファイル転送オプション, ファイル転送のオプション
ログインオプション, ログインオプションとアクセス制御, ロギングのオプション
ローカルユーザーのオプション, ローカルユーザーのオプション
匿名ユーザーのオプション, 匿名ユーザーのオプション
形式, vsftpd 設定オプション
起動, vsftpd の起動と停止

W

wbinfo プログラム, Samba ディストリビューションプログラム
Web サーバー (参照 Apache HTTP サーバー)
Windows 2000
Samba を使用した共有への接続, 暗号化されたパスワード
Windows 98
Samba を使用した共有への接続, 暗号化されたパスワード
Windows ME
Samba を使用した共有への接続, 暗号化されたパスワード

Windows NT 4.0
Samba を使用した共有への接続, 暗号化されたパスワード

Windows XP
Samba を使用した共有への接続, 暗号化されたパスワード

X

X
/etc/X11/xorg.conf
Device, Device セクション
DRI, DRI セクション
Files セクション, Files セクション
InputDevice セクション, InputDevice セクション
Monitor, Monitor セクション
Screen, Screen セクション
Section タグ, 設定の構造
ServerFlags セクション, ServerFlags セクション
ServerLayout セクション, ServerLayout セクション
ブール値, 設定の構造
導入, xorg.conf.d ディレクトリ, xorg.conf ファイル
構造, 設定の構造

fonts
Fontconfig、フォントの追加先, Fontconfig へのフォントの追加
導入, フォント

X クライアント, X Window System, デスクトップ環境とウィンドウマネージャー
startx コマンド, ランレベル 3
xinit コマンド, ランレベル 3
ウィンドウマネージャー, ウィンドウマネージャー
デスクトップ環境, デスクトップ環境

X サーバー, X Window System
機能, X サーバー

その他のリソース, その他のリソース
インストールされているドキュメント, インストールされているドキュメント
役立つ Web サイト, 役立つ Web サイト

ウィンドウマネージャー
kwin, ウィンドウマネージャー
metacity, ウィンドウマネージャー
mwm, ウィンドウマネージャー
twm, ウィンドウマネージャー

ディスプレイマネージャー
GNOME, ランレベル 5
KDE, ランレベル 5
prefdm スクリプト, ランレベル 5
xdm, ランレベル 5
お好みの設定, ランレベル 5
定義, ランレベル 5

デスクトップ環境
GNOME, デスクトップ環境
KDE, デスクトップ環境

フォント
Fontconfig, フォント
FreeType, フォント
Xft, フォント

ランレベル, ランレベルと X
3, ランレベル 3
5, ランレベル 5

導入, X Window System
設定ディレクトリ
/etc/X11/xorg.conf.d, xorg.conf.d ディレクトリ

設定ファイル
/etc/X11/ ディレクトリ, X サーバーの設定ファイル
/etc/X11/xorg.conf, xorg.conf ファイル
オプション, X サーバーの設定ファイル
サーバーオプション, xorg.conf.d ディレクトリ, xorg.conf ファイル

X Window System (参照 X)
X.500 (参照 OpenLDAP)
X.500 Lite (参照 OpenLDAP)
xinit (参照 X)
Xorg (参照 Xorg)

Y

Yum
Yum と Yum リポジトリの設定, Yum と Yum リポジトリの設定
Yum を使用したインストール, パッケージのインストール
Yum を使用したパッケージのアンインストール, パッケージの削除
yum remove package_name, パッケージの削除

Yum を使用したパッケージの一覧表示
    glob 表現, パッケージの一覧表示
    yum grouplist, パッケージの一覧表示
    yum list, パッケージの一覧表示
    yum list all, パッケージの一覧表示
    yum list available, パッケージの一覧表示
    yum list installed, パッケージの一覧表示
    yum repolist, パッケージの一覧表示

Yum を使用したパッケージの検索
    yum search, パッケージの検索

Yum を使用したパッケージの表示
    yum info, パッケージ情報の表示

Yum を使用したパッケージグループのアンインストール, パッケージの削除
Yum を使用したパッケージグループのインストール, パッケージのインストール
Yum プラグイン, Yum のプラグイン
Yum リポジトリ
    Yum と Yum リポジトリの設定, Yum と Yum リポジトリの設定

[main] オプションの設定, [main] オプションの設定
[repository] オプションの設定, [repository] オプションの設定
その他のリソース, その他のリソース
パッケージとパッケージグループ, パッケージとパッケージグループ
パッケージの表示
    yum info, パッケージ情報の表示

プラグイン
    fs-snapshot, プラグインの詳細
    kabi, プラグインの詳細
    presto, プラグインの詳細
    product-id, プラグインの詳細
    protect-packages, プラグインの詳細
    refresh-packagekit, プラグインの詳細
    rhnplugin, プラグインの詳細
    security, プラグインの詳細
    subscription-manager, プラグインの詳細
    yum-downloadonly, プラグインの詳細

プラグインの有効化, Yum プラグインを有効/設定/無効にする方法
プラグインの無効化, Yum プラグインを有効/設定/無効にする方法
プラグインの設定, Yum プラグインを有効/設定/無効にする方法
リポジトリ, Yum リポジトリを有効/設定/無効にする方法, Yum リポジトリの作成

変数, Yum 変数の使用

Yum での更新

すべてのパッケージと依存関係の更新, パッケージの更新

セキュリティ関連パッケージの更新, パッケージの更新

パッケージの更新, パッケージの更新

単一パッケージの更新, パッケージの更新

更新の確認, 更新の確認

Yum リポジトリ

PackageKit を使用した Yum リポジトリの表示, ソフトウェアソースを最新の情報への更新 (yum リポジトリ)