Red Hat Enterprise Linux 7

セキュリティーガイド

RHEL サーバーとワークステーションをセキュアにする概念と手法

Last Updated: 2023-08-31

Red Hat Enterprise Linux 7 セキュリティーガイド

RHEL サーバーとワークステーションをセキュアにする概念と手法

Mirek Jahoda
Red Hat Customer Content Services
mjahoda@redhat.com

Jan Fiala
Red Hat Customer Content Services
jafiala@redhat.com

Stephen Wadeley
Red Hat Customer Content Services

Robert Krátký
Red Hat Customer Content Services

Martin Prpič
Red Hat Customer Content Services

Ioanna Gkioka
Red Hat Customer Content Services

Tomáš Čapek
Red Hat Customer Content Services

Yoana Ruseva
Red Hat Customer Content Services

Miroslav Svoboda
Red Hat Customer Content Services

法律上の通知

概要

本書は、ユーザーおよび管理者が、ローカルおよびリモートの侵入、悪用、および悪意のある行為からワークステーションおよびサーバーを保護するプロセスおよびプラクティスを学ぶのに利用できます。 本ガイドでは、Red Hat Enterprise Linux を対象としていますが、概念および手法はすべての Linux システムに適用できます。データセンター、勤務先、および自宅で安全なコンピューティング環境を構築するのに必要な計画およびツールを詳細に説明します。 管理上の適切な知識、警戒体制、およびツールを備えることで、Linux を実行しているシステムの機能をフルに活用して、大概の一般的な侵入や悪用の手法からシステムを保護できます。

目次

第1章 セキュリティーの概要

ビジネスの運営や個人情報の把握ではネットワーク化された強力なコンピューターへの依存度が高まっていることから、各種業界ではネットワークとコンピューターのセキュリティーの実践に関心が向けられています。企業は、システム監査を適正に行い、ソリューションが組織の運営要件を満たすようにするために、セキュリティーの専門家の知識と技能を求めてきました。多くの組織はますます動的になってきていることから、従業員は、会社の重要な IT リソースに、ローカルまたはリモートからアクセスするようになっています。このため、セキュアなコンピューティング環境に対するニーズはより顕著になっています。

にも関わらず、多くの組織 (個々のユーザーも含む) は、機能性、生産性、便利さ、使いやすさ、および予算面の懸念事項にばかり目を向け、セキュリティーをその結果論と見なし、セキュリティーのプロセスが見過ごされています。したがって、適切なセキュリティーの確保は、無許可の侵入が発生して **はじめて** 徹底されることも少なくありません。多くの侵入の試みを阻止する効果的な方法は、インターネットなどの信頼できないネットワークにサイトを接続する前に、適切な措置を講じることです。

注記

本書では、**/lib** ディレクトリー内のファイルにいくつかの参照を行います。64 ビットシステムを使用する場合は、上記のファイルの一部は **/lib64** にある可能性があります。

1.1. コンピューターセキュリティーとは

コンピューターセキュリティーは、コンピューティングと情報処理の幅広い分野で使用される一般的な用語です。コンピューターシステムとネットワークを使用して日々の業務を行い、重要な情報へアクセスしている業界では、企業データを総体的資産の重要な部分であると見なしています。総保有コスト (Total Cost of Ownership: TCO)、投資利益率 (Return on Investment: ROI)、サービスの品質 (Quality of Service: QoS) などの用語や評価指標は日常的なビジネス用語として用いられるようになっています。各種の業界が、計画およびプロセス管理コストの一環として、これらの評価指標を用いてデータ保全性や可用性などを算出しています。電子商取引などの業界では、データの可用性と信頼性は、成功と失敗の違いを意味します。

1.1.1. セキュリティーの標準化

企業はどの業界でも、米国医師会 (AMA: American Medical Association)、米国電気電子学会 (IEEE: Institute of Electrical and Electronics Engineers) などの標準化推進団体が作成する規制やルールに従っています。情報セキュリティーにも同じことが言えます。多くのセキュリティーコンサルタントやベンダーが *機密性 (Confidentiality)*、*保全性 (Integrity)*、*可用性 (Availability)* の頭文字をとった CIA として知られる標準セキュリティーモデルを採用しています。この 3 階層モデルは、機密情報のリスク評価やセキュリティー方針の確立において、一般的に採用されているモデルです。以下でこの CIA モデルを説明します。

- 機密性 - 機密情報は、事前に定義された個人だけが利用できるようにする必要があります。許可されていない情報の送信や使用は、制限する必要があります。たとえば、情報に機密性があれば、権限のない個人が顧客情報や財務情報を悪意のある目的 (ID 盗難やクレジット詐欺など) で入手できません。

- 保全性 - 情報は、改ざんして不完全または不正確なものにすべきではありません。承認されていないユーザーが、機密情報を変更したり破壊したりする機能を使用できないように制限する必要があります。

- 可用性 - 情報は、認証されたユーザーが必要な時にいつでもアクセスできるようにする必要があります。可用性は、合意した頻度とタイミングで情報を入手できることを保証します。これは、パーセンテージで表されることが多く、ネットワークサービスプロバイダーやその企業顧客が使用するサービスレベルアグリーメント (SLA) で正式に合意となります。

1.1.2. 暗号化ソフトウェアおよび認定

以下のナレッジベースの記事では、Red Hat Enterprise Linux コア暗号化コンポーネントの概要 (どのコンポーネントか選択されているか、どのように選択されているか、オペレーティングシステムにどのように統合されているかどうか、ハードウェアセキュリティーモジュールおよびスマートカードにどのように対応しているか、および暗号化認証がどのように適用されているか) を説明します。

- RHEL7 Core Crypto Components

1.2. セキュリティーコントロール

多くの場合、コンピューターセキュリティーは、一般に *コントロール* と呼ばれる以下の 3 つのマスターカテゴリーに分類されます。

- 物理的
- 技術的
- 管理的

この 3 つのカテゴリーは、セキュリティーの適切な実施における主な目的を定義するものです。このコントロールには、コントロールと、その実装方法を詳細化するサブカテゴリーがあります。

1.2.1. 物理的コントロール

物理的コントロールは、機密資料への非認証アクセスの抑止または防止のために、明確な構造でセキュリティー対策を実施します。物理的コントロールの例は以下のとおりです。

- 有線監視カメラ
- 動作または温度の感知アラームシステム
- 警備員
- 写真付き身分証明書
- 施錠された、デッドボルト付きのスチールドア
- バイオメトリクス (本人確認を行うための指紋、声、顔、虹彩、筆跡などの自動認識方法が含まれます)

1.2.2. 技術的コントロール

技術的コントロールでは、物理的な構造物やネットワークにおける機密データのアクセスや使用を制御する基盤となる技術を使用します。技術的コントロールは広範囲に及び、以下のような技術も含まれます。

- 暗号化
- スマートカード
- ネットワーク認証
- アクセス制御リスト (ACL)
- ファイルの完全性監査ソフトウェア

1.2.3. 管理的コントロール

管理的コントロールは、セキュリティーの人的要素を定義します。これは組織内のあらゆるレベルの職員や社員に関連するもので、誰がどのリソースや情報にアクセスするかを、次のような手段で決定します。

- トレーニングおよび認識の向上
- 災害準備および復旧計画
- 人員採用と分離の戦略
- 人員登録とアカウンティング

1.3. 脆弱性のアセスメント

時間やリソースがあり、その気になれば、攻撃者はほとんどすべてのシステムに侵入できます。現在利用できるセキュリティーの手順と技術をすべて駆使しても、すべてのシステムを侵入から完全に保護できる訳ではありません。ルーターは、インターネットへのセキュアなゲートウェイを提供します。ファイアウォールは、ネットワークの境界を保護します。仮想プライベートネットワーク (VPN) では、データが、暗号化されているストリームで安全に通過できます。侵入検知システムは、悪意のある活動を警告します。しかし、これらの技術が成功するかどうかは、以下のような数多くの要因によって決まります。

- 技術の設定、監視、および保守を行うスタッフの専門知識
- サービスとカーネルのパッチ、および更新を迅速かつ効率的に行う能力
- ネットワーク上での警戒を常に怠らない担当者の能力

データシステムと各種技術が動的であることを考えると、企業リソースを保護するタスクは極めて複雑になる可能性もあります。この複雑さゆえに、使用するすべてのシステムの専門家を見つけることは、多くの場合困難になります。情報セキュリティーの多くの分野によく精通している人材を確保することはできても、多くの分野を専門とするスタッフを確保することは容易ではありません。これは、情報セキュリティーの各専門分野で、継続的な注意と重点が必要となるためです。情報セキュリティーは、常に変化しています。

脆弱性アセスメントは、お使いのネットワークとシステムのセキュリティーに関する内部監査です。(「セキュリティーの標準化」 で説明するように) このアセスメントの結果により、ネットワークの機密性、完全性、および可用性の状態が明らかになります。通常、脆弱性アセスメントは、対象システムとリソースに関する重要なデータを収集する調査フェーズから開始します。その後システム準備フェーズとなります。基本的にこのフェーズでは、対象を絞り、すべての既知の脆弱性を調べます。準備フェーズが終わると報告フェーズになります。ここでは、調査結果が高中低のカテゴリーに分類され、対象のセキュリティーを向上させる (または脆弱性のリスクを軽減する) 方法が話し合われます。

たとえば、自宅の脆弱性アセスメントを実施することを想定してみましょう。まずは自宅のドアを点検し、各ドアが閉まっていて、かつ施錠されていることを確認します。また、すべての窓が完全に閉まっていて鍵が閉まっていることも確認します。これと同じ概念が、システム、ネットワーク、および電子データにも適用されます。悪意のあるユーザーはデータを盗んで、破壊します。悪意のあるユーザーが使用するツール、思考、動機に注目すると、彼らの行動にすばやく反応することが可能になります。

1.3.1. アセスメントとテストの定義

脆弱性アセスメントは、*外部からの視点* と *内部からの視点* の 2 種類に分類できます。

外部からの視点で脆弱性アセスメントを実施する場合は、外部からシステムに攻撃を試みます。会社を

外から見ることで、クラッカーの視点に立つことができます。一般にルーティング可能な IP アドレス、*DMZ* にあるシステム、ファイアウォールの外部インターフェイスなど、クラッカーが目を付けるものに着目します。DMZ は非武装地帯 (demilitarized zone) を表し、企業のプライベート LAN などの信頼できる内部ネットワークと、公的なインターネットなどの信頼できない外部ネットワークの間にあるコンピューターまたは小さなサブネットワークに相当します。通常、DMZ には Web (HTTP) サーバー、FTP サーバー、SMTP (e-mail) サーバー、DNS サーバーなど、インターネットのトラフィックにアクセスできるデバイスが含まれます。

内部からの視点で脆弱性アセスメントを実施する場合、実行者は内部関係者であり、信頼されるステータスにあることから、有利な立場になります。内部からの視点は、実行者やその同僚がシステムにログオンした時点で得られるものです。プリントサーバー、ファイルサーバー、データベースなどのリソースを見ることができます。

これら 2 種類の脆弱性アセスメントには大きな違いがあります。社内のユーザーには、部外者が得られない多くの特権が付与されています。多くの組織では、侵入者を締め出すようにセキュリティーが設定されています。しかし、組織内の細かい部分 (部門内ファイアウォール、ユーザーレベルのアクセス制御および内部リソースに対する認証手順など) には、セキュリティー対策がほとんど行われていません。また、一般的にほとんどのシステムは社内にあるため、内部からの方がより多くのリソースを確認できます。いったん社外に移動すると、ステータスは信頼されない状態になります。通常、外部から利用できるシステムやリソースは、非常に限られたものになります。

脆弱性アセスメントと *侵入テスト* の違いを考えてみましょう。脆弱性アセスメントを、侵入テストの第一歩と捉えてください。このアセスメントで得られる情報は、その後のテストで使用します。アセスメントは抜け穴や潜在的な脆弱性を検査する目的で行われるのに対し、侵入テストでは調査結果を実際に使用する試みがなされます。

ネットワークインフラストラクチャーのアセスメントは動的なプロセスです。セキュリティー (情報セキュリティーおよび物理的なセキュリティー) は動的なものです。アセスメントを実施することで概要が明らかになり、誤検出 (False positives) および検出漏れ (False negatives) が示される場合があります。誤検出は、実際には存在しない脆弱性をツールが検出することを指します。検出漏れは、実際の脆弱性が検出されないことを指します。

セキュリティー管理者の力量は、使用するツールとその管理者が有する知識で決まります。現在使用できるアセスメントツールのいずれかを選び、それらをシステムに対して実行すると、ほぼ間違いなく誤検出がいくつか見つかります。プログラム障害でもユーザーエラーでも、結果は同じです。ツールは、誤検出することもあれば、さらに悪い場合は、検出漏れをすることもあります。

脆弱性アセスメントと侵入テストの違いが定義されたところで、新たなベストプラクティスの一環として侵入テストを実施する前に、アセスメントの結果を注意深く確認し、検討してみましょう。

警告

実稼働システムで脆弱性を悪用する試みを行わないでください。システムおよびネットワークの生産性ならびに効率に悪影響を与える可能性があります。

脆弱性アセスメントの実施には、以下のような利点があります。

- 情報セキュリティーに事前にフォーカスできる
- クラッカーが発見する前に潜在的な不正使用を発見できる

- システムを最新の状態に維持し、パッチを適用できる
- スタッフの成長と専門知識の開発を促す
- 経済的な損失や否定的な評判を減らす

1.3.2. 脆弱性評価に関する方法論の確立

脆弱性アセスメントの方法論が確立されれば、脆弱性アセスメント用のツール選択に役立ちます。現時点では、事前定義の方法論や業界で承認された方法論はありませんが、一般常識やベストプラクティスを適切なガイドとして活用できます。

ターゲットとは何を指していますか ?1 台のサーバー、またはネットワーク全体およびネットワーク内にあるすべてのサーバーを確認しますか ?会社外ですか？それとも内部ですか？ この質問に対する回答は、選択したツールだけでなく、そのツールの使用方法を決定する際に重要です。

方法論の確立の詳細は、以下の Web サイトを参照してください。

- https://www.owasp.org/ – 『The Open Web Application Security Project 』

1.3.3. 脆弱性アセスメントのツール

アセスメントは、情報収集ツールを使用することから始まります。ネットワーク全体を評価する際は、最初にレイアウトを描いて、稼働しているホストを把握します。ホストの場所を確認したら、それぞれのホストを個別に検査します。各ホストにフォーカスするには別のツールセットが必要になります。どのツールを使用すべきかを知っておくことは、脆弱性の発見において最も重要なステップになる可能性があります。

日常生活のあらゆる状況と同様に、同じジョブを実行できる異なるツールは数多くあります。この概念は脆弱性アセスメントの実施にも当てはまります。ツールには、オペレーティングシステムやアプリケーションに固有のものや、(使用されるプロトコルに基づいて) ネットワークに固有のツールもあります。無料のツールも、有料のツールもあります。直感的で使いやすいツールもあれば、不可解で文書化が不十分で、かつ他のツールにはない機能を備えているツールもあります。

適切なツールを見つけることは困難なタスクとなる場合もあり、最終的には経験が重要になります。可能であれば、テストラボを立ち上げて、できるだけ多くのツールを試し、それぞれの長所と短所に注意してみてください。ツールの **README** ファイルまたは man ページを参照してください。さらに、インターネット上でそのツールに関する記事、ステップバイステップのガイド、あるいはメーリングリストなど、より多くの情報を検索してください。

以下に説明するツールは、利用可能なツールのほんの一部です。

1.3.3.1. Nmap を使用したホストのスキャン

Nmap は、ネットワークのレイアウトを決定するために使用できる一般的なツールです。**Nmap** は数年にわたって利用でき、情報収集時に最も頻繁に使用されるツールになります。オプションや使い方を詳しく説明した優れたマニュアルページが付属しています。管理者は、ネットワーク上で **Nmap** を使用してホストシステムを検索し、それらのシステムでポートを開くことができます。

Nmap は、脆弱性アセスメントにおける確かな最初のステップです。ネットワーク内のすべてのホストをマッピングし、**Nmap** が特定のホストで実行しているオペレーティングシステムの特定を試みることを許可するオプションを渡すこともできます。**Nmap** は、セキュアなサービスを使用し、未使用のサービスを制限するポリシーを確立するのに適した基盤です。

Nmap をインストールするには、**root** で **yum install nmap** コマンドを実行します。

1.3.3.1.1. Nmap の使用

nmap は、シェルプロンプトから **nmap** コマンドを入力し、スキャンするマシンのホスト名または **IP** アドレスを入力すると実行できます。

```
nmap <hostname>
```

たとえば、ホスト名が **foo.example.com** のマシンをスキャンするには、シェルプロンプトで以下を入力します。

```
~]$ nmap foo.example.com
```

基本的なスキャンの結果は、以下のようになります (ホストの位置やその他のネットワーク状況により、数分かかる場合があります)。

```
Interesting ports on foo.example.com:
Not shown: 1710 filtered ports
PORT    STATE  SERVICE
22/tcp  open   ssh
53/tcp  open   domain
80/tcp  open   http
113/tcp closed auth
```

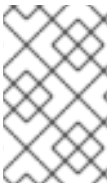
Nmap は、リッスンまたはサービスの待機中の最も一般的なネットワーク通信ポートをテストします。この知識は、不要なサービスや未使用のサービスを終了したい管理者に役立ちます。

Nmap の使用に関する詳細は、以下の URL にある公式のホームページを参照してください。

http://www.insecure.org/

1.3.3.2. Nessus

Nessus はフルサービスセキュリティースキャナーです。**Nessus** のプラグインアーキテクチャーにより、ユーザーはシステムおよびネットワーク用にカスタマイズできます。**Nessus** は、スキャナーと同様に、依存する署名データベースと同じくらい優れています。幸いなことに、**Nessus** は頻繁に更新され、完全なレポート機能、ホストスキャン、およびリアルタイムの脆弱性検索を備えています。強力なツールや **Nessus** として頻繁に更新されるツールであっても、誤検出や誤検出が発生する可能性があることに注意してください。



注記

Nessus クライアントおよびサーバーソフトウェアを使用するには、サブスクリプションが必要です。これは、この人気のあるアプリケーションの使用に興味があるかもしれないユーザーへの参照として、このドキュメントに含まれています。

Nessus の詳細は、以下の URL にある公式 Web サイトを参照してください。

http://www.nessus.org/

1.3.3.3. OpenVAS

OpenVAS (*Open Vulnerability Assessment System*)は、脆弱性のスキャンや包括的な脆弱性管理に使用できるツールおよびサービスのセットです。**OpenVAS** フレームワークは、ソリューションのさまざまなコンポーネントを制御するための Web ベース、デスクトップ、およびコマンドラインツールを多数

提供します。**OpenVAS** のコア機能はセキュリティースキャナーによって提供されます。セキュリティースキャナーは、毎日更新される 33 を超えるネットワーク脆弱性テスト(NVT)を使用します。**Nessus** (「**Nessus**」 を参照)とは異なり、**OpenVAS** はサブスクリプションを必要としません。

OpenVAS の詳細については、以下の URL にある公式 Web サイトを参照してください。

http://www.openvas.org/

1.3.3.4. Nikto

Nikto は優れた *共通ゲートウェイインターフェイス* (CGI)スクリプトスキャナーです。**Nikto** は CGI の脆弱性をチェックするだけでなく、侵入検知システムなど、簡潔な方法でチェックします。これには詳細なドキュメントが含まれており、プログラムを実行する前に入念に確認する必要があります。CGI スクリプトを提供する Web サーバーがある場合、**Nikto** はこれらのサーバーのセキュリティーをチェックするための優れたリソースになります。

Nikto の詳細は、以下の URL を参照してください。

http://cirt.net/nikto2

1.4. セキュリティーへの脅威

1.4.1. ネットワークセキュリティーへの脅威

ネットワークの以下の要素を設定する際に不適当なプラクティスが行われると、攻撃のリスクが増大します。

セキュリティーが十分ではないアーキテクチャー
間違った設定のネットワークは、未承認ユーザーの主要なエントリーポイントになります。信頼に基づいたオープンなローカルネットワークを、安全性が非常に低いインターネットに対して無防備な状態にしておくことは、犯罪の多発地区でドアを半開きにしておくようなものです。すぐに何かが起きることはないかもしれませんが、**いずれ**、誰かが、このチャンスを悪用するでしょう。

ブロードキャストネットワーク
システム管理者は、セキュリティー計画においてネットワーキングハードウェアの重要性を見落としがちです。ハブやルーターなどの単純なハードウェアは、ブロードキャストやノンスイッチの仕組みに基づいています。つまり、あるノードがネットワークを介して受信ノードにデータを送信するときは常に、受信ノードがデータを受信して処理するまで、ハブやルーターがデータパケットのブロードキャストを送信します。この方式は、外部侵入者やローカルホストの未認証ユーザーが仕掛けるアドレス解決プロトコル (*ARP*) およびメディアアクセスコントロール (*MAC*) アドレスの偽装に対して最も脆弱です。

集中化サーバー
ネットワーキングのもうひとつの落とし穴は、集中化されたコンピューティングの使用にあります。多くの企業では、一般的なコスト削減手段として、すべてのサービスを 1 台の強力なマシンに統合しています。集中化は、複数サーバーを設定するよりも管理が簡単で、コストを大幅に削減できるので便利です。ただし、集中化されたサーバーはネットワークにおける単一障害点となります。中央のサーバーが攻撃されると、ネットワークが完全に使用できなくなるか、データの不正操作や盗難が起きやすくなる可能性があります。このような場合は、中央サーバーがネットワーク全体へのアクセスを許可することになります。

1.4.2. サーバーセキュリティーへの脅威

サーバーには組織の重要情報が数多く含まれることが多いため、サーバーのセキュリティーは、ネットワークのセキュリティーと同様に重要です。サーバーが攻撃されると、クラッカーが意のままにすべて

のコンテンツを盗んだり、不正に操作したりできるようになる可能性があります。以下のセクションでは、主要な問題の一部を詳述します。

未使用のサービスと開かれたポート

Red Hat Enterprise Linux 7 のフルインストールを行うと、アプリケーションとライブラリーのパッケージが 1000 個以上含まれます。ただし、サーバー管理者が、ディストリビューションに含まれるすべての個別パッケージをインストールすることはほとんどありません。代わりに、複数のサーバーアプリケーションを含むパッケージのベースインストールを行います。インストールされるパッケージの数を制限する理由の説明と追加のリソースについては「必要なパッケージの最小限のインストール」を参照してください。

システム管理者は、インストールに含まれるプログラムに注意を向けずにオペレーティングシステムをインストールしてしまうことがよくあります。これにより、不要なサービスがインストールされ、デフォルト設定でオンになっていることで、問題が発生する場合があります。つまり、管理者が気が付かないところで、Telnet、DHCP、DNS などの不要なサービスがサーバーやワークステーションで実行し、その結果、サーバーへの不要なトラフィックが発生したり、クラッカーがシステムのパスを悪用できてしまう可能性があります。ポートを閉じる方法や未使用のサービスを無効にする方法についての詳細は、「サービスのセキュア化」を参照してください。

パッチが適用されないサービス

デフォルトのインストールに含まれるほとんどのサーバーアプリケーションは、ソフトウェアの細部まで徹底的にテストされており、堅牢な作りになっています。何年も実稼働環境で使用される中で、そのコードは入念に改良され、数多くのバグが発見されて修正されてきました。

しかし、完璧なソフトウェアというものはなく、改良の余地は常にあります。または、比較的新しいソフトウェアは、実稼働環境に導入されてから日が浅く、他のサーバーソフトウェアほど普及していないこともあるため、厳密なテストが期待通りに行われていない状況も少なくありません。

開発者やシステム管理者が、サーバーアプリケーションで悪用される可能性のあるバグを発見することも多々あり、Bugtraq メーリングリスト (http://www.securityfocus.com)、Computer Emergency Response Team (CERT) Web サイト (http://www.cert.org) などで、バグ追跡やセキュリティー関連の Web サイトに関連する情報が公開されています。このような情報発信は、コミュニティーにセキュリティーの脆弱性を警告する効果的な方法ではありますが、システムに速やかにパッチを当てるかどうかは個々のシステム管理者が決定します。クラッカーも、パッチが適用されていないシステムがあればクラッキングできるように、脆弱性トラッキングサービスにアクセスし、関連情報を利用できることを考慮すると、速やかな対応がとりわけ重要になります。優れたシステム管理を行うには、警戒を怠らず、バグ追跡を絶えず行い、適切なシステム保守を実行して、よりセキュアなコンピューティング環境を維持することが求められます。

システムを最新状態に維持する方法についての詳細は、*3章システムを最新の状態に保つ* を参照してください。

管理における不注意

管理者がシステムにパッチを当てないことが、サーバーのセキュリティーに対する最大の脅威の 1 つになります。*SysAdmin, Audit, Network, Security Institute* (*SANS*) によると、コンピューターのセキュリティー脆弱性の主な原因は、訓練を受けていない人にセキュリティーの保守を任せ、保守を行うために必要な訓練や学ぶ時間を与えないことにあります。[1] これは、管理者の経験の少なさだけでなく、管理者の過信やモチベーションの低さなども原因となります。

管理者が、サーバーやワークステーションにパッチを当てることを忘れたり、システムのカーネルやネットワーク通信のログメッセージを見落とす場合もあります。その他にも、よく起こるケースとして、サービスのデフォルトパスワードや鍵を変更しないまま放置しておくことが挙げられます。たとえば、データベースにはデフォルトの管理パスワードが設定されているものがありますが、ここでは、システム管理者がインストール後すぐにデフォルトパスワードを変更することを、データベース開発者は

想定しています。しかし、データベース管理者がパスワードを変更することを忘れると、クラッカーの経験が浅くても、周知のデフォルトパスワードを使用してデータベースの管理者権限を得ることができます。この他に、管理者の不注意によりサーバーが危険にさらされる場合もあります。

本質的に安全ではないサービス
どんなに注意深い組織であっても、選択するネットワークサービスが本質的に安全でない限り、攻撃を受けやすくなります。たとえば、多くのサービスは、信頼できるネットワークでの使用を想定して開発されますが、このサービスが (本質的に信頼できない) インターネットで利用可能になる時点で、この仮定は成立しなくなります。

安全ではないネットワークサービスの例として、暗号化されていないユーザー名とパスワードを認証時に要求するサービスが挙げられます。具体例としては、Telnet や FTP の 2 つがあげられます。パケット盗聴ソフトウェアがリモートユーザーとこのようなサービスの間のトラフィックを監視していれば、ユーザー名とパスワードは簡単に傍受される可能性があります。

また、基本的にこのようなサービスはセキュリティー業界で *中間者* 攻撃と呼ばれる攻撃の被害者になりやすくなります。この種の攻撃では、クラッカーが、ネットワーク上でクラッキングしたネームサーバーを操って、目標のサーバーではなくクラッカーのマシンを指定して、ネットワークトラフィックをリダイレクトします。誰かがサーバーへのリモートセッションを開くと、攻撃者のマシンがリモートサービスと無防備なユーザーとの間に存在する目に見えないパイプとして機能し、この間を流れる情報を取り込みます。このようにして、クラッカーはサーバーやユーザーに気付かれることなく、管理パスワードや生データを収集できるようになります。

安全ではないサービスの例としては、他にも NFS、NIS などのネットワークファイルシステムおよび情報サービスが挙げられます。このサービスは、LAN 利用を目的として開発されましたが、(リモートユーザー用の) WAN も対象に含まれるように拡張されました。NFS では、クラッカーによる NFS 共有のマウントやそこに格納されているものへのアクセスを防ぐ認証やセキュリティーの仕組みがデフォルトで設定されていません。NIS も、プレーンテキストの ASCII または DBM (ASCII から派生) データベースに、パスワードやファイルパーミッションなど、ネットワーク上の全コンピューターへの周知が必要となる重要な情報を保持しています。クラッカーがこのデータベースのアクセス権を取得すると、管理者のアカウントを含む、ネットワークのすべてのユーザーアカウントにアクセスできるようになります。

Red Hat Enterprise Linux 7 では、デフォルトでは、上記のサービスがすべて無効になっています。ただし、管理者は、このようなサービスを使用しないといけない場合があるため、注意して設定することが重要となります。安全な方法でサービスをセットアップする詳細情報は、「サービスのセキュア化」を参照してください。

1.4.3. ワークステーションおよび家庭用 PC のセキュリティーに対する脅威

ワークステーションや家庭用 PC はネットワークやサーバーほど攻撃にさらされることはないかもしれませんが、クレジットカード情報のような機密データが含まれるため、システムクラッカーの標的になります。ワークステーションは知らぬ間に攻撃者によって選択され、一連の攻撃でスレーブマシンとして使用される可能性もあります。このため、ユーザーはワークステーションの脆弱性を理解しておくと、オペレーティングシステムの再インストールや、深刻な場合はデータ盗難からの回復といった問題から免れることができます。

不適切なパスワード
攻撃者が最も簡単にシステムへのアクセスを得る方法の 1 つとして、パスワードが適切でないことが挙げられます。パスワードを作成する際のよくある落とし穴を避ける方法については、「パスワードセキュリティー」を参照してください。

脆弱なクライアントアプリケーション
管理者がサーバーに十分な安全対策を施し、パッチを当てている場合でも、リモートユーザーによるアクセスが安全であるわけではありません。たとえば、サーバーが公開ネットワーク上で Telnet や FTP のサービスを提供している場合、攻撃者はネットワークを通過するプレーンテキストのユーザー名とパ

スワードを取り込み、アカウント情報を使用してリモートユーザーのワークステーションにアクセスすることが可能です。

SSH などのセキュアなプロトコルを使用している場合であっても、クライアントアプリケーションを定期的に更新していないと、リモートユーザーは特定の攻撃を受けやすくなる可能性があります。たとえば、v.1 SSH クライアントは悪意のある SSH サーバーからの X 転送攻撃に対して脆弱です。クライアントがサーバーに接続すると、攻撃者はネットワーク上でクライアントによるキー入力やマウス操作をひそかに収集できます。この問題は v.2 SSH プロトコルで修正されましたが、ユーザーはどのアプリケーションにこのような脆弱性があるかを追跡し、必要に応じてアプリケーションを更新する必要があります。

「デスクトップのセキュリティー」では、管理者とホームユーザーがコンピューターのワークステーションの脆弱性を限定するために取るべき手順をより詳細に説明しています。

1.5. 一般的な不正使用と攻撃

表1.1「一般的な不正使用」 では、侵入者が組織のネットワークリソースにアクセスするために使用する最も一般的な不正使用とエントリーポイントの例を挙げて詳しく説明します。この一般的な不正使用では、それがどのように実行され、管理者がその攻撃からネットワークをどのように適切に保護できるかを理解していることが重要になります。

表1.1 一般的な不正使用

不正使用	説明	注記
空またはデフォルトのパスワード	管理パスワードを空白のままにしたり、製品ベンダーが設定したデフォルトのパスワードをそのまま使用します。これは、ルーターやファイアウォールなどのハードウェアで最もよく見られますが、Linux で実行するサービスにはデフォルトの管理者パスワードが指定されているものがあります (ただし Red Hat Enterprise Linux 7 には含まれません)。	一般的に、ルーター、ファイアウォール、VPN、ネットワーク接続ストレージ (NAS) の機器など、ネットワークハードウェアに関連するものです。 多数のレガシーオペレーティングシステム、特にサービスをバンドルしたオペレーティングシステム (UNIX や Windows など) でよく見られます。 管理者が急いで特権ユーザーアカウントを作成したためにパスワードが空白のままになっていることがありますが、このような空白のパスワードは、このアカウントを発見した悪意のあるユーザーが利用できる絶好のエントリーポイントとなります。
デフォルトの共有鍵	セキュアなサービスでは、開発や評価テスト向けにデフォルトのセキュリティー鍵がパッケージ化されていることがあります。この鍵を変更せずにインターネットの実稼働環境に置いた場合は、同じデフォルトの鍵を持つ **すべての** ユーザーがその共有鍵のリソースや、そこにあるすべての機密情報にアクセスできるようになります。	無線アクセスポイントや、事前設定済みでセキュアなサーバー機器に最も多く見られます。

不正使用	説明	注記
IP スプーフィング	リモートマシンがローカルネットワークのノードのように動作し、サーバーに脆弱性を見つけるとバックドアプログラムまたはトロイの木馬をインストールして、ネットワークリソース全体へのコントロールを得ようとします。	スプーフィングは、攻撃者が標的となるシステムへの接続を調整するのに、TCP/IP シーケンス番号を予測しなければならないため、かなり難しくなりますが、クラッカーの脆弱性の攻撃を支援する利用可能なツールがいくつかあります。 *ソースベース* の認証技術を使用するサービス(**rsh**、**telnet**、FTP など)を使用するターゲットシステムの実行サービスに依存します。これは、**ssh** または SSL/TLS で使用される PKI またはその他の形式の暗号化認証と比較すると推奨されません。
盗聴	2 つのノード間の接続を盗聴することにより、ネットワーク上のアクティブなノード間を行き交うデータを収集します。	この種類の攻撃には大抵、Telnet、FTP、HTTP 転送などのプレーンテキストの転送プロトコルが使用されます。 リモートの攻撃者がこのような攻撃を仕掛けるには、LAN で、攻撃するシステムへのアクセス権が必要になります。通常、クラッカーは、LAN 上にあるシステムを危険にさらすためにアクティブ攻撃 (IP スプーフィングや中間者攻撃など) を仕掛けます。 パスワードのなりすましに対する防護策としては、暗号化鍵交換、ワンタイムパスワード、または暗号化された認証によるサービス使用が挙げられます。通信中は強力な暗号化を実施することをお勧めします。

不正使用	説明	注記
サービスの脆弱性	攻撃者はインターネットで実行しているサービスの欠陥や抜け穴を見つけます。攻撃者がこの脆弱性を利用する場合は、システム全体と格納されているデータを攻撃するだけでなく、ネットワーク上の他のシステムも攻撃する可能性があります。	CGI などの HTTP ベースのサービスは、リモートのコマンド実行やインタラクティブなシェルアクセスに対しても脆弱です。HTTP サービスが nobody などの権限のないユーザーとして実行している場合でも、設定ファイルやネットワークマップなどの情報が読み取られる可能性があります。または、攻撃者がサービス拒否攻撃を開始して、システムのリソースを浪費させたり、他のユーザーが利用できないようにする可能性もあります。 開発時およびテスト時には気が付かない脆弱性がサービスに含まれることがあります。(アプリケーションのメモリーバッファー領域をあふれさせ、任意のコマンドを実行できるようなインタラクティブなコマンドプロンプトを攻撃者に提供するように、攻撃者が任意の値を使用してサービスをクラッシュさせる *バッファーオーバーフロー*などの) 脆弱性は、完全な管理コントロールを攻撃者に与えるものとなる可能性があります。 管理者は、root 権限でサービスが実行されないようにし、ベンダー、または CERT、CVE などのセキュリティー組織がアプリケーション用のパッチやエラータ更新を提供していないかを常に注意する必要があります。
アプリケーションの脆弱性	攻撃者は、デスクトップやワークステーションのアプリケーション (電子メールクライアントなど) に欠陥を見つけ出し、任意のコードを実行したり、将来のシステム侵害のためにトロイの木馬を移植したり、システムを破壊したりします。攻撃を受けたワークステーションがネットワークの残りの部分に対して管理特権を持っている場合は、さらなる不正使用が起こる可能性があります。	ワークステーションとデスクトップは、ユーザーが侵害を防いだり検知するための専門知識や経験を持たないため、不正使用の対象になりやすくなります。認証されていないソフトウェアをインストールしたり、要求していないメールの添付ファイルを開く際には、それに伴うリスクについて個々に通知することが必須です。 電子メールクライアントソフトウェアが添付ファイルを自動的に開いたり、実行したりしないようにするといった、予防手段を取ることが可能です。さらに、Red Hat Network や他のシステム管理サービスなどからワークステーションのソフトウェアを自動更新することにより、マルチシートのセキュリティーデプロイメントの負担を軽減できます。

不正使用	説明	注記
サービス拒否攻撃 (DoS: Denial of Service)	単独の攻撃者または攻撃者のグループは、目標のホスト (サーバー、ルーター、ワークステーションのいずれか) に認証されていないパケットを送り、組織のネットワークまたはサーバーのリソースに対して攻撃を仕掛けます。これにより、正当なユーザーがリソースを使用できなくなります。	米国で最も多く報告された DOS の問題は、2000 年に発生しました。この時、通信量が非常に多い民間および政府のサイトが一部が利用できなくなりました。*ゾンビ (zombie)* や、リダイレクトされたブロードキャストノードとして動作する高帯域幅接続を有し、セキュリティー侵害された複数のシステムを使用して、調整された ping フラッド攻撃が行われたためです。 通常ソースパケットは、真の攻撃元を調査するのが難しくなるよう、偽装 (または再ブロードキャスト) されています。 **iptables** および **snort** などのネットワーク侵入検知システムを使用したイングレスフィルターリング(IETF rfc2267)の進進は、管理者が分散 DoS 攻撃を追跡し、防止するのに役立ちます。

[1] http://www.sans.org/security-resources/mistakes.php

第2章 インストール時におけるセキュリティーのヒント

セキュリティーは、Red Hat Enterprise Linux 7 をインストールするためにディスクドライブに CD または DVD を初めて挿入したときから始まります。最初からシステムのセキュリティーを設定することで、追加のセキュリティー設定を実装することがより簡単になります。

2.1. BIOS のセキュア化

BIOS (もしくは BIOS に相当するもの) およびブートローダーをパスワードで保護することで、システムに物理的にアクセス可能な未承認ユーザーがリムーバブルメディアを使用して起動したり、シングルユーザーモードで root 権限を取得することを防ぐことができます。このような攻撃に対するセキュリティー対策は、ワークステーションの情報の機密性とマシンの場所によって異なります。

たとえば、見本市で使用されていて機密情報を含んでいないマシンでは、このような攻撃を防ぐことが重要ではないかもしれません。しかし、同じ見本市で、企業ネットワークに対して暗号化されていない SSH 秘密鍵のある従業員のノートパソコンが、誰の監視下にもなく置かれていた場合は、重大なセキュリティー侵害につながり、その影響は企業全体に及ぶ可能性があります。

一方で、ワークステーションが権限のあるユーザーもしくは信頼できるユーザーのみがアクセスできる場所に置かれてるのであれば、BIOS もしくはブートローダーの安全確保は必要ない可能性もあります。

2.1.1. BIOS パスワード

コンピューターの BIOS をパスワードで保護する主な 2 つの理由を以下に示します。[2]:

1. **BIOS 設定の変更を防止する** - 侵入者が BIOS にアクセスした場合は、CD-ROM やフラッシュドライブから起動するように設定できます。このようにすると、侵入者がレスキューモードやシングルユーザーモードに入ることが可能になり、システムで任意のプロセスを開始したり、機密性の高いデータをコピーできるようになってしまいます。

2. **システムの起動を防止する** - BIOS の中には起動プロセスをパスワードで保護できるものもあります。これを有効にすると、攻撃者は BIOS がブートローダーを開始する前にパスワード入力を求められます。

BIOS パスワードの設定方法はコンピューターメーカーで異なるため、具体的な方法はコンピューターのマニュアルを参照してください。

BIOS パスワードを忘れた場合は、マザーボードのジャンパーでリセットするか、CMOS バッテリーを外します。このため、可能な場合はコンピューターのケースをロックすることが推奨されます。ただし、CMOS バッテリーを外す前にコンピューターもしくはマザーボードのマニュアルを参照してください。

2.1.1.1. 非 BIOS ベースのシステムの保護

その他のシステムやアーキテクチャーでは、異なるプログラムを使用して x86 システムの BIOS とほぼ同等の低レベルのタスクを実行します。*Unified Extensible Firmware Interface* (*UEFI*) シェルなどがこの例になります。

BIOS のようなプログラムをパスワード保護する方法は、メーカーにお問い合わせください。

2.2. ディスクのパーティション設定

Red Hat は、**/boot** ディレクトリー、/ ディレクトリー、**/home** ディレクトリー、/ tmp ディレクトリー、および / **var** /**tmp** / ディレクトリーに個別のパーティションを作成することを推奨します。各パーティションを作成する理由は以下のようになります。

/boot

このパーティションは、システムの起動時にシステムが最初に読み込むパーティションです。Red Hat Enterprise Linux 7 でシステムを起動するのに使用するブートローダーとカーネルイメージはこのパーティションに保存されます。このパーティションは暗号化しないでください。このパーティションが / に含まれており、そのパーティションが暗号化されているなどの理由で利用できなくなると、システムを起動できなくなります。

/home

ユーザーデータ(**/home**)が別のパーティションではなく / に保存されている場合、パーティションが満杯になり、オペレーティングシステムが不安定になる可能性があります。また、システムを Red Hat Enterprise Linux 7 の次のバージョンにアップグレードする際に、**/home** パーティションにデータを保存できると、このデータはインストール時に上書きされないため、はるかに簡単になります。ルートパーティション(/)が破損すると、データが完全に失われる可能性があります。したがって、パーティションを分けることが、データ損失に対する保護につながります。また、このパーティションを、頻繁にバックアップを作成する対象にすることも可能です。

/tmp および /var/tmp/

/tmp ディレクトリーおよび **/var/tmp/** ディレクトリーは、どちらも長期間保存する必要がないデータを保存するために使用されます。しかし、このいずれかのディレクトリーでデータがあふれると、ストレージ領域がすべて使用されてしまう可能性があります。これが発生し、これらのディレクトリーが / に保存されていると、システムが不安定になり、クラッシュする可能性があります。そのため、このディレクトリーは個別のパーティションに移動することが推奨されます。

注記

インストールプロセス時に、パーティションを暗号化するオプションがあります。パスフレーズを入力する必要があります。これは、パーティションのデータを保護するのに使用されるバルク暗号鍵を解除する鍵として使用されます。詳細は、「LUKS ディスクの暗号化の使用」 を参照してください。

2.3. 必要なパッケージの最小限のインストール

コンピューターの各ソフトウェアには脆弱性が潜んでいる可能性があるため、実際に使用するパッケージのみをインストールすることがベストプラクティスになります。インストールを DVD から行う場合は、インストールしたいパッケージのみを選択するようにします。その他のパッケージが必要になる場合は、後でいつでもシステムに追加できます。

最小 インストール環境のインストールに関する詳細は、Red Hat Enterprise Linux 7 インストールガイドの ソフトウェアの選択 の章を参照してください。最小限のインストールは、**--nobase** オプションを使用したキックスタートファイルでも実行できます。キックスタートインストールの詳細については、Red Hat Enterprise Linux 7 インストールガイドのパッケージの選択 セクションを参照してください。

2.4. インストールプロセス時のネットワーク接続の制限

Red Hat Enterprise Linux をインストールする際に使用するインストールメディアは、特定のタイミングで作成されたスナップショットです。そのため、セキュリティー修正が最新のものではなく、このインストールメディアで設定するシステムが公開されてから修正された特定の問題に対して安全性に欠ける場合があります。

脆弱性が含まれる可能性のあるオペレーティングシステムをインストールする場合には、必ず、公開レベルを、必要最小限のネットワークゾーンに限定してください。最も安全な選択肢は、インストールプロセス時にマシンをネットワークから切断した状態にするネットワークなしのゾーンです。インターネット接続からのリスクが最も高く、一方で LAN またはイントラネット接続で十分な場合もあります。セキュリティーのベストプラクティスに従い、ネットワークから Red Hat Enterprise Linux をインストールする場合は、お使いのリポジトリーに最も近いゾーンを選択するようにしてください。

ネットワーク接続の設定に関する詳しい情報は、Red Hat Enterprise Linux 7 インストールガイドのネットワーク & ホストの章を参照してください。

2.5. インストール後の手順

以下は、Red Hat Enterprise Linux のインストール直後に実行する必要があるセキュリティー関連の手順です。

1. システムを更新します。root で以下のコマンドを入力します。

```
~]# yum update
```

2. ファイアウォールサービスの **firewalld** は、Red Hat Enterprise Linux のインストールで自動的に有効になっていますが、キックスタート設定などで明示的に無効となっている場合もあります。このような場合は、ファイアウォールを再度有効にすることが推奨されます。

   firewalld を開始するには、root で次のコマンドを実行します。

```
~]# systemctl start firewalld
~]# systemctl enable firewalld
```

3. セキュリティーを強化するために、不要なサービスは無効にしてください。たとえば、コンピューターにプリンターがインストールされていない場合は、以下のコマンドを使用して **cups** サービスを無効にします。

```
~]# systemctl disable cups
```

   アクティブなサービスを確認するには、次のコマンドを実行します。

```
~]$ systemctl list-units | grep service
```

2.6. 関連情報

インストールに関する全般的な情報は、Red Hat Enterprise Linux 7 インストールガイド を参照してください。

---

[2] システム BIOS はメーカーによって異なるため、いずれかのタイプのパスワード保護のみをサポートするものもあれば、いずれのタイプのパスワード保護もサポートしないものもあります。

第3章 システムを最新の状態に保つ

この章では、システムを最新の状態に保つためのプロセスについて説明します。これには、セキュリティー更新のインストール方法の計画と設定、新しく更新されたパッケージによって導入された変更の適用、およびセキュリティー勧告を追跡するための Red Hat カスタマーポータルの使用が含まれます。

3.1. インストール済みソフトウェアのメンテナーンス

セキュリティー上の脆弱性が検出されると、潜在的なセキュリティーリスクを制限するために、影響を受けるソフトウェアを更新する必要があります。ソフトウェアが現在サポートされている Red Hat Enterprise Linux ディストリビューション内のパッケージの一部である場合、Red Hat では、脆弱性を修正する更新されたパッケージをできるだけ早くリリースするよう尽力しています。

多くの場合、特定のセキュリティーの不正使用に関する発表には、問題を修正するパッチ (またはソースコード) も含まれます。このパッチは、その後、Red Hat Enterprise Linux パッケージに適用され、エラータ更新としてテストされ、リリースされます。しかし、発表にパッチが含まれていない場合、Red Hat の開発者はまずソフトウェアのメンテナーと協力して問題を解決します。問題が修正されると、パッケージはエラータ更新としてテストされ、リリースされます。

お使いのシステムで使用されているソフトウェアのエラータ更新がリリースされた場合は、システムが潜在的に脆弱となる時間を最小限に抑えるため、影響を受けるパッケージをできるだけ早く更新することを強く推奨します。

3.1.1. セキュリティーの更新の立案と設定

すべてのソフトウェアにはバグが含まれます。多くの場合、これらのバグは、システムを悪意のあるユーザーにさらす可能性のある脆弱性をもたらすことがあります。更新されていないパッケージは、コンピューターへの不正侵入の原因としてよく知られています。発見された脆弱性を迅速に排除し、悪用されないようにするために、セキュリティーパッチを適時にインストールする計画を実施します。

セキュリティー更新が利用可能になったらテストし、インストールするようにスケジュールします。更新がリリースされてからシステムにインストールされるまでの間、システムを保護するために追加の制御を行う必要があります。これらの制御は脆弱性そのものによって異なりますが、追加のファイアウォールルール、外部ファイアウォールの使用、またはソフトウェア設定の変更が含まれる場合があります。

サポート対象パッケージのバグは、エラータメカニズムを使用して修正されます。エラータは、1つ以上のRPMパッケージとそのエラータが対処する問題の簡単な説明で設定されています。すべてのエラータは、アクティブなサブスクリプションをお持ちのお客様に **Red Hat Subscription Management**サービスで配布されます。セキュリティー問題に対処するエラータは、*Red Hat セキュリティーアドバイザリー* と呼ばれます。

セキュリティーエラータを使った作業についての詳細情報は、「カスタマーポータルでのセキュリティーアドバイザリーの表示」を参照してください。**RHN Classic** からの移行方法など、**Red Hat Subscription Management** サービスの詳細は、Red Hat Subscription Management に関するドキュメントを参照してください。

3.1.1.1. Yum のセキュリティー機能の使用

Yum パッケージマネージャーには、セキュリティーエラータの検索、一覧表示、表示、インストールに使用できるセキュリティー関連の機能が複数含まれています。これらの機能を使用すると、**Yum** を使用してセキュリティー更新以外のものをインストールすることもできます。

システムで利用可能なセキュリティー関連の更新を確認するには、**root** で以下のコマンドを入力します。

```
~]# yum check-update --security
Loaded plugins: langpacks, product-id, subscription-manager
rhel-7-workstation-rpms/x86_64                | 3.4 kB  00:00:00
No packages needed for security; 0 packages available
```

上記のコマンドは非対話型モードで実行されるため、利用可能な更新があるかどうかを自動チェックするためのスクリプトで使用できることに注意してください。このコマンドは、利用可能なセキュリティー更新がある場合は 100 を、ない場合は 0 を終了値として返します。エラーが発生すると、1 が返されます。

同様に、以下のコマンドを使用して、セキュリティー関連の更新のみをインストールします。

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

updateinfo サブコマンドを使用して、利用可能な更新に関するリポジトリーが提供する情報を表示または操作します。**updateinfo** サブコマンド自体は、多くのコマンドを受け入れ、セキュリティー関連の使用に関連するいくつかのコマンドを利用できます。これらのコマンドの概要については、表 3.1「yum updateinfo で使用できるセキュリティー関連コマンドです。」 を参照してください。

表3.1 yum updateinfo で使用できるセキュリティー関連コマンドです。

コマンド	説明
[アドバイザリー]	1 つ以上のアドバイザリーに関する情報を表示します。*advisories* をアドバイザリー番号に置き換えます。
cves	CVE (*Common Vulnerabilities and Exposures*) に関連する情報のサブセットを表示します。
セキュリティー または **秒**	すべてのセキュリティー関連情報を表示します。
severity [severity_level] または **sev [severity_level]**	指定された *severity_level* のセキュリティー関連パッケージに関する情報を表示します。

3.1.2. パッケージの更新とインストール

システムでソフトウェアを更新する場合は、信頼できるソースから更新をダウンロードすることが重要です。攻撃者は、問題を解決するはずのパッケージと同じバージョン番号で、異なるセキュリティーエクスプロイトを施したパッケージを簡単に作り直し、インターネット上で公開することができます。こうした事態が発生すると、元の RPM に対するファイル検証などのセキュリティー対策を講じても、不正アクセスを検知することができません。このため、RPM は Red Hat などの信頼できるソールからのみダウンロードし、その保全性を検証するためにパッケージの署名を確認することが極めて重要になります。

Yum パッケージマネージャーの使用方法に関する詳細情報は、Red Hat Enterprise Linux 7 システム管理者のガイドの **Yum** の章を参照してください。

3.1.2.1. 署名パッケージの検証

すべての Red Hat Enterprise Linux パッケージは、Red Hat **GPG** キーで署名されます。**gpg** は **GNU Privacy Guard** または **GnuPG** の略です。これは、分散ファイルの信頼性を確保するために使用するフリーソフトウェアパッケージです。パッケージ署名の検証に失敗した場合は、パッケージが改ざんされ

ている可能性があるため、信頼することができません。

Yum パッケージマネージャーを使用すると、インストールまたはアップグレードするすべてのパッケージを自動的に検証できます。この機能はデフォルトで無効にされています。システムでこのオプションを設定するには、**/etc/yum.conf** 設定ファイルで **gpgcheck** 設定ディレクティブが **1** に設定されていることを確認します。

ファイルシステム上のパッケージファイルを手動で確認するには、次のコマンドを使用します。

```
rpmkeys --checksig package_file.rpm
```

Red Hat パッケージの署名のプラクティスに関する追加情報は、Red Hat カスタマーポータルの Product Signing (GPG) Keys の記事を参照してください。

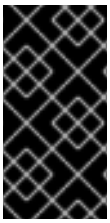
3.1.2.2. 署名パッケージのインストール

ファイルシステムから検証済みのパッケージをインストールするには、「署名パッケージの検証」を参照してください。パッケージの検証方法は、以下のように **root** で **yum install** コマンドを実行します。

```
yum install package_file.rpm
```

シェルグロブを使用して、複数のパッケージを一度にインストールします。たとえば、次のコマンドは、現在のディレクトリーにすべての **.rpm** パッケージをインストールします。

```
yum install *.rpm
```



重要

セキュリティーエラータをインストールする前に、エラータレポートに含まれる特別な指示を必ず読み、それに従って実行してください。エラータ更新に基づく変更の適用についての全般的な指示は、「インストールされた更新によって導入された変更の適用」を参照してください。

3.1.3. インストールされた更新によって導入された変更の適用

セキュリティーエラータや更新をダウンロードしてインストールした後は、古いソフトウェアの使用を停止し、新しいソフトウェアの使用を開始することが重要です。これがどのように行われるかは、更新されたソフトウェアのタイプによって異なります。以下のリストは、ソフトウェアの一般的なカテゴリーを項目別に分類し、パッケージのアップグレード後に更新されたバージョンを使用するための手順を示したものです。

注記

一般に、システムを再起動することが、ソフトウェアパッケージの最新バージョンが使用されていることを確認する最も確実な方法です。ただし、このオプションは必ずしも必須ではなく、システム管理者が常に利用できるわけでもありません。

アプリケーション

ユーザースペースアプリケーションとは、ユーザーが起動することができるすべてのプログラムのことです。通常、このようなアプリケーションは、ユーザー、スクリプト、または自動タスクユーティリティーが、これらのアプリケーションを起動したときにのみ使用されます。

このようなユーザースペースのアプリケーションが更新されたら、システム上のアプリケーションのインスタンスをすべて停止し、プログラムを再度起動して更新されたバージョンを使用します。

カーネル

カーネルは、Red Hat Enterprise Linux 7 オペレーティングシステムの中核となるソフトウェアコンポーネントです。メモリー、プロセッサー、周辺機器へのアクセスを管理し、すべてのタスクをスケジューリングします。

カーネルはその中心的なロールを担っているため、コンピューターを再起動せずにカーネルを再起動することはできません。そのため、システムを再起動するまで、更新されたバージョンのカーネルを使用することはできません。

KVM

qemu-kvm および libvirt のパッケージが更新されると、すべてのゲスト仮想マシンを停止して、関連の仮想化モジュールを再読み込みし (またはホストシステムを再起動し)、仮想マシンを再起動する必要があります。

lsmod コマンドを使用して、kvm、**kvm-intel**、または **kvm -amd** からのモジュールが読み込まれます。次に、**modprobe -r** コマンドを使用して **modprobe -a** コマンドを削除してから、影響を受けるモジュールを再読み込みします。以下に例を示します。

```
~]# lsmod | grep kvm
kvm_intel             143031  0
kvm                   460181  1 kvm_intel
~]# modprobe -r kvm-intel
~]# modprobe -r kvm
~]# modprobe -a kvm kvm-intel
```

共有ライブラリー

共有ライブラリーは、多くのアプリケーションやサービスで使用される **glibc** などのコードの単位です。共有ライブラリーを利用するアプリケーションは、通常、アプリケーションの初期化時に共有コードをロードするため、更新されたライブラリーを利用するアプリケーションはすべて、一旦停止して再スタートする必要があります。

実行中のどのアプリケーションが特定のライブラリーにリンクするかを確認するには、**lsof** コマンドを使用します。

```
lsof library
```

たとえば、実行中のアプリケーションが **libwrap.so.0** ライブラリーにリンクするものを確認するには、以下を入力します。

```
~]# lsof /lib64/libwrap.so.0
COMMAND     PID USER  FD   TYPE DEVICE SIZE/OFF     NODE NAME
pulseaudi 12363 test mem    REG  253,0    42520 34121785 /usr/lib64/libwrap.so.0.7.6
gnome-set 12365 test mem    REG  253,0    42520 34121785 /usr/lib64/libwrap.so.0.7.6
gnome-she 12454 test mem    REG  253,0    42520 34121785 /usr/lib64/libwrap.so.0.7.6
```

このコマンドは、ホストアクセス制御に **TCP** ラッパーを使用する実行中のすべてのプログラムの一覧を返します。したがって、tcp_wrappers パッケージの更新時に一覧表示されるプログラムをすべて停止し、再起動する必要があります。

systemd サービス

systemd サービスは通常ブートプロセス中に起動する永続的なサーバープログラムです。systemd サービスの例としては、**sshd** または **vsftpd** などがあります。

これらのプログラムは通常、マシンが実行されている限りメモリーに保持されるため、更新された各 systemd サービスは、パッケージがアップグレードされた後に停止して再起動する必要があります。これは、**systemctl** コマンドを使用して、**root** ユーザーとして実行できます。

```
systemctl restart service_name
```

service_name を、**sshd** など、再起動するサービスの名前に置き換えます。

他のソフトウェア

下のアプリケーションを正しく更新するには、以下のリンク先のリソースに記載されている手順に従ってください。

- **Red Hat Directory Server** - 問題の Red Hat Directory Server バージョンの 『 リリースノート』 (https://access.redhat.com/documentation/ja-JP/Red_Hat_Directory_Server/)を参照してください。
- **Red Hat Enterprise Virtualization Manager** - 問題となっている Red Hat Enterprise Virtualization のバージョンの 『インストールガイド』 (https://access.redhat.com/documentation/ja-JP/Red_Hat_Enterprise_Virtualization/)を参照してください。

3.2. RED HAT カスタマーポータルの使用

https://access.redhat.com/ の Red Hat カスタマーポータルは、Red Hat 製品に関連する公式情報を提供するお客様向けのメインリソースです。カスタマーポータルでは、ドキュメントの検索、サブスクリプションの管理、製品や更新のダウンロード、サポートケースの作成のほか、セキュリティーの更新情報についても知ることができます。

3.2.1. カスタマーポータルでのセキュリティーアドバイザリーの表示

アクティブなサブスクリプションがあるシステムに関連するセキュリティーアドバイザリー（エラータ）を表示するには、https://access.redhat.com/ でカスタマーポータルにログインし、メインページの **Download Products & Updates** ボタンをクリックします。**Software & Download Center** ページを入力するときは、**エラータ** ボタンをクリックして、登録済みのシステムに対応するアドバイザリーの一覧が表示されます。

アクティブなすべての Red Hat 製品のセキュリティー更新の一覧を表示するには、ページ上部のナビゲーションメニューを使用して **Security → Security Updates → Active Products** に移動します。

表の左側にあるエラータコードをクリックして、個々のアドバイザリーに関する詳細情報を表示します。次のページには、原因、結果、必要な修正など、特定のエラータに関する説明だけでなく、特定のエラータが更新するすべてのパッケージのリストと、更新の適用方法についても説明されています。このページには、関連するリファレンス (関連する CVE など) へのリンクも含まれています。

3.2.2. カスタマーポータルページでの CVE への移動

CVE*Common Vulnerabilities and Exposures*) プロジェクトは、MITRE Corporation によって管理されており、脆弱性やセキュリティーエクスポージャーの標準的な名称のリストです。カスタマーポータルの Red Hat 製品に関連する CVE の一覧を表示するには、https://access.redhat.com/ でアカウントにログ

インし、ページ上部のナビゲーションメニューを使用して **Security → Resources → CVE Database** に移動します。

表の左側にある CVE コードをクリックすると、個々の脆弱性の詳細情報が表示されます。次のページには、指定されたCVEの説明だけでなく、影響を受ける Red Hat 製品のリストと関連する Red Hat のエラータへのリンクが記載されています。

3.2.3. 問題の重大度の分類を理解する

Red Hat 製品で発見されたすべてのセキュリティー問題は、問題の重大度に応じて *Red Hat 製品セキュリティー*によって影響度が評価されます。影響度は、低度の影響、中程度の影響、重要な影響、および重大な影響の 4 段階評価になります。さらに、すべてのセキュリティー問題は、CVSS(*Common Vulnerability Scoring System*) ベーススコアを使用して評価されます。

これらの評価を組み合わせることで、セキュリティー問題の影響を理解し、システムのアップグレード戦略のスケジュールと優先順位を決定することができます。なお、この評価は、現在の脅威レベルではなく、バグの技術的分析に基づいた特定の脆弱性の潜在的リスクを反映したものであることに注意してください。つまり、特定の欠陥に対するエクスプロイトがリリースされても、セキュリティーの影響度の評価は変更されないことを意味します。

カスタマーポータルの各レベルの重大度評価の詳細は、Severity Ratingsのページをご覧ください。

3.3. 関連情報

セキュリティー更新、その適用方法、Red Hat カスタマーポータル、および関連トピックの詳細については、以下の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

- yum(8) : **Yum** パッケージマネージャーの man ページには、**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、および削除する方法に関する情報が記載されています。
- rpmkeys(8) : **rpmkeys** ユーティリティーの man ページでは、このプログラムを使用してダウンロードしたパッケージの信頼性を検証する方法が説明されています。

オンラインドキュメント

- Red Hat Enterprise Linux 7 システム管理者のガイド : Red Hat Enterprise Linux 7 『 のシステム管理者ガイド』 では、Red Hat Enterprise Linux 7 システムでパッケージのインストール、更新、および削除に使用する **Yum** コマンドおよび **rpm** コマンドの使用について説明しています。
- Red Hat Enterprise Linux 7 SELinux ユーザーおよび管理者のガイド : Red Hat Enterprise Linux 7 『の SELinux ユーザーおよび管理者のガイド』 では、**SELinux** の *強制アクセス制御* メカニズムの設定が説明されています。

Red Hat カスタマーポータル

- Red Hat カスタマーポータル (セキュリティー) – カスタマーポータルのセキュリティーセクションには、Red Hat CVE データベースなどの最も重要なリソースへのリンクのほか、Red Hat 製品セキュリティーの連絡先が含まれています。
- Red Hat セキュリティーブログ – Red Hat のセキュリティー専門家による最新のセキュリティー問題に関する記事。

関連項目

- 2章 *インストール時におけるセキュリティーのヒント* は、最初からシステムを安全に設定し、後で追加のセキュリティー設定を簡単に実装する方法について説明しています。

- 「GPG 鍵の作成」 では、通信を認証するための個人 **GPG** キーのセットを作成する方法を説明します。

第4章 ツールとサービスでシステムを強化する

4.1. デスクトップのセキュリティー

Red Hat Enterprise Linux 7 は、攻撃に対してデスクトップを強化し、不正アクセスを防止するためのいくつかの方法を提供します。このセクションでは、ユーザーパスワード、セッションとアカウントのロック、およびリムーバブルメディアの安全な取り扱いに関する推奨事項を説明します。

4.1.1. パスワードセキュリティー

パスワードは、Red Hat Enterprise Linux 7 がユーザーの ID を確認するために使用する主な方法です。このため、パスワードのセキュリティーは、ユーザー、ワークステーション、ネットワークを保護するために非常に重要です。

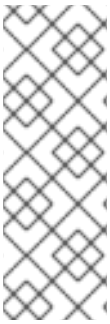
セキュリティー上の理由から、インストールプログラムは、*セキュアハッシュアルゴリズム 512*(**SHA512**) およびシャドウパスワードを使用するようにシステムを設定します。これらの設定を変更しないことを強く推奨します。

インストール時にシャドウパスワードの選択を解除すると、すべてのパスワードが、誰でも読み取り可能な **/etc/passwd** ファイルに一方向ハッシュとして保存されます。これにより、システムはオフラインのパスワードクラッキング攻撃に対して脆弱になります。侵入者が通常のユーザーとしてマシンにアクセスできる場合は、**/etc/passwd** ファイルを自分のマシンにコピーし、それに対して任意の数のパスワードクラッキングプログラムを実行できます。ファイル内に安全でないパスワードがあれば、パスワードクラッカーに発見されるのは時間の問題です。

シャドウパスワードは、root ユーザーのみが読み取り可能なファイル **/etc/shadow** にパスワードハッシュを保存することで、このタイプの攻撃を排除します。

これにより、潜在的な攻撃者は、SSH や FTP などのマシン上のネットワークサービスにログインして、リモートでパスワードクラッキングを試みるようになります。この種のブルートフォース攻撃ははるかに遅く、何百回ものログイン試行の失敗がシステムファイルに書き込まれるため、明らかな痕跡が残ります。もちろん、クラッカーがパスワードの弱いシステムで深夜に攻撃を開始した場合、夜明け前にアクセスできるようになり、ログファイルを編集して痕跡を消している可能性もある。

形式およびストレージの考慮事項に加えて、コンテンツの問題があります。パスワードクラッキング攻撃からアカウントを保護するためにユーザーができる最も重要なことは、強力なパスワードを作成することです。



注記

Red Hat では、Red Hat Identity Management (IdM) などの中央認証ソリューションを使用することを推奨しています。ローカルパスワードの使用よりも、中央ソリューションの使用が推奨されます。詳細は、次を参照してください。

- Red Hat Identity Management の概要
- パスワードポリシーの定義

4.1.1.1. 強固なパスワードの作成

安全なパスワードを作成する際、ユーザーは、短いパスワードや複雑なパスワードよりも長いパスワードの方が強力であることを覚えておく必要があります。数字や特殊文字、大文字を含むとはいえ、たった 8 文字のパスワードを作成することは得策ではありません。John The Ripper などのパスワードクラッキングツールは、このような、人が覚えにくいパスワードの解読に最適化されています。

情報理論では、エントロピーは確率変数に関連する不確実性のレベルであり、ビットで示されます。エントロピー値が高いほど、パスワードの安全性は高くなります。NIST SP 800-63-1 によると、一般的に選択される 50000 個のパスワードで設定される辞書に存在しないパスワードには、少なくとも 10 ビットのエントロピーが必要です。そのため、4 つのランダムな単語で設定されるパスワードは、約 40 ビットのエントロピーを含んでいます。セキュリティーを強化するために複数の単語で設定される長いパスワードは、*パスフレーズ*とも呼ばれます。次に例を示します。

```
randomword1 randomword2 randomword3 randomword4
```

システムが大文字、数字、または特殊文字の使用を強制する場合、上記の推奨事項に続くパスフレーズは、最初の文字を大文字に変更し、**1!**を追加するなどして簡単に変更できます。このような変更は、パスフレーズのセキュリティーを大幅に **向上させるものではない** ことに注意してください。

自分でパスワードを作成するもう 1 つの方法は、パスワードジェネレータを使用することです。**pwmake** は、大文字、小文字、数字、特殊文字の 4 つの文字グループすべてで設定されるランダムパスワードを生成するためのコマンドラインツールです。このユーティリティーでは、パスワードの生成に使用するエントロピービットの数を指定することができます。エントロピーは **/dev/urandom** からプルされます。指定できる最小のビット数は 56 ビットで、ブルートフォース攻撃がまれなシステムやサービスのパスワードとしては十分と言えます。攻撃者がパスワードハッシュファイルに直接アクセスできないアプリケーションでは、64 ビットで十分です。攻撃者がパスワードハッシュを直接入手する可能性がある場合や、パスワードを暗号化キーとして使用する場合は、80 ビットから 128 ビットを使用する必要があります。無効な数のエントロピービットを指定すると、**pwmake** はデフォルトのビットを使用します。128 ビットのパスワードを作成する場合は、次のコマンドを入力します。

```
pwmake 128
```

安全なパスワードの作成にはさまざまなアプローチがありますが、以下のような悪い習慣は必ず避けてください。

- 辞書の単語 1 つ、外国語の単語 1 つ、倒置語、または数字のみを使用すること。
- パスワードまたはパスフレーズが 10 文字未満であること。
- キーボードレイアウトの一連のキーを使用すること。
- パスフレーズを書き留めること。
- 生年月日、記念日、家族の名前、またはペットの名前など、個人情報をパスワードに使用すること。
- 複数のマシンで同じパスフレーズやパスワードを使用すること。

安全なパスワードの作成は必須ですが、特に大規模な組織内のシステム管理者にとっては、パスワードを適切に管理することも重要です。以下のセクションでは、組織内でユーザーパスワードを作成し、管理するためのグッドプラクティスについて詳しく説明します。

4.1.1.2. 強固なパスワードの強制

多数のユーザーを抱える組織では、システム管理者は、強力なパスワードの使用を強制するために、2 つの基本的なオプションを利用することができます。システム管理者がユーザーのパスワードを作成することもできますし、パスワードの強度が十分であることを確認しながら、ユーザーが独自のパスワードを作成できるようにすることもできます。

ユーザーのパスワードを作成することで、パスワードの安全性は確保されますが、組織が大きくなるにつれ、大変な作業となります。また、ユーザーがパスワードを書き留めることで、パスワードが流出する危険性も高まります。

このような理由から、多くのシステム管理者は、ユーザー自身がパスワードを作成することを望んでいますが、それらのパスワードが十分に強力であるかについては積極的に検証しています。場合によっては、管理者は、パスワードエージングを通じて、ユーザーに定期的にパスワードを変更するように強制することもあります。

ユーザーがパスワードを作成または変更するように求められると、*PAM*-aware (*Pluggable Authentication Modules*)である **passwd** コマンドラインユーティリティーを使用して、パスワードが短すぎるか、または簡単に解読できるかどうかを確認できます。このチェックは、**pam_pwquality.so** PAM モジュールにより実行されます。

注記

Red Hat Enterprise Linux 7 では、PAM モジュール **pam_pwquality** が、パスワード品質チェックのデフォルトモジュールとして Red Hat Enterprise Linux 6 で使用されていました。**これは、pam_splunk と同じバックエンドを使用します。**

pam_pwquality モジュールは、パスワード強度を一連のルールに対してチェックするために使用されます。その手順は 2 つのステップで設定されています。最初に、提供されたパスワードが辞書にあるかどうかをチェックします。辞書にない場合は、さらにいくつものチェックを続けます。pam_pwquality は、/etc/pam.d/passwd ファイルの password コンポーネント内の他の PAM モジュールとともにスタックされ、ルールのカスタムセットは /etc/security/pwquality.conf 設定ファイルで指定します。これらのチェックの完全なリストについては、pwquality.conf (8) の man ページを参照してください。

例4.1 pwquality.confでパスワードの強度チェックの設定

pam_quality の使用を有効にするには、以下の行を /etc/pam.d/passwd ファイルの パスワードスタックに追加します。

```
password    required    pam_pwquality.so retry=3
```

チェックのオプションは、1 行に 1 つずつ指定します。たとえば、文字の最小長が 8 文字のパスワードを要求するには、以下の行を /etc/security/pwquality.conf ファイルに追加します。

```
minlen = 8
minclass = 4
```

文字シーケンスと同じ連続した文字のパスワード強度チェックを設定するには、以下の行を /etc/security/pwquality.conf に追加します。

```
maxsequence = 3
maxrepeat = 3
```

この例では、入力されたパスワードには、abcd などの単調なシーケンスで 3 文字以上を含

み、1111 などの同一の 3 文字を超える連続した文字を含めることはできません。

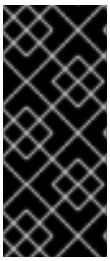
注記

root ユーザーは、パスワード作成のルールを適用するユーザーであるため、警告メッセージが表示されても、自分自身または通常のユーザーに任意のパスワードを設定できます。

4.1.1.3. パスワードエージングの設定

パスワードエージングは、組織内の不正なパスワードから保護するためにシステム管理者が使用するもう一つの技術です。パスワードエージングとは、指定された期間 (通常は 90 日間) が経過すると、ユーザーは新しいパスワードを作成するように求められることを意味します。この背景には、ユーザーが定期的にパスワードを変更することを強制された場合、クラックされたパスワードは侵入者にとって限られた時間しか有効でないという理論があります。しかし、パスワードエージングには、ユーザーがパスワードを書き留める可能性が高くなるというデメリットがあります。

Red Hat Enterprise Linux 7 でパスワードエージングを指定するには、コマンドを使用し て ください。



重要

Red Hat Enterprise Linux 7 では、シャドウパスワードはデフォルトで有効になっています。詳細は、Red Hat Enterprise Linux 7 System Administrator's Guide を参照してください。

コマンドの -M オプションは、パスワードの最大有効日数を指定します。たとえば、ユーザーのパスワードを 90 日間で期限切れになるように設定するには、以下のコマンドを使用します。

```
chage -M 90 username
```

上記のコマンドでは、*username* をユーザー名に置き換えます。パスワードの有効期限を無効にするには、-M オプションの 後に -1 の値を使用します。

このコマンドで使用できるオプションの詳細は、以下 の表を参照してください。

表4.1 chage コマンドラインオプション

オプション	説明
-d *days*	1970 年 1 月 1 日以降にパスワードを変更した日数を指定します。
-E *date*	アカウントがロックされる日付を YYYY-MM-DD のフォーマットで指定します。日付の代わりに、1970 年 1 月 1 日からの日数を使うこともできます。
-I *days*	パスワードの有効期限後、アカウントをロックするまでの非アクティブ日数を指定します。値が **0** の場合、パスワードが失効してもアカウントはロックされません。
-l	現在のアカウントエージングの設定を一覧表示します。
-m *days*	ユーザーがパスワードを変更しなければならない最短日数を指定します。値が **0** の場合、パスワードは期限切れになりません。
-M *days*	パスワードの最大有効日数を指定します。このオプションで指定した日数と、**-d** オプションで指定した日数を足した日数が、現在の日数より少ない場合、ユーザーはアカウントを使用する前にパスワードを変更する必要があります。
-W *days*	パスワードの有効期限の何日前にユーザーに警告を発するかを指定します。

インタラクティブモードで コマンドを使用し て、複数のパスワードエージングおよびアカウントの詳細を変更することもできます。次のコマンドを使用して、インタラクティブモードに入ります。

```
chage <username>
```

以下は、このコマンドを使用した対話セッションの例です。

```
~]# chage juan
Changing the aging information for juan
Enter the new value, or press ENTER for the default
Minimum Password Age [0]: 10
Maximum Password Age [99999]: 90
Last Password Change (YYYY-MM-DD) [2006-08-18]:
Password Expiration Warning [7]:
Password Inactive [-1]:
Account Expiration Date (YYYY-MM-DD) [1969-12-31]:
```

ユーザーが初めてログインしたときにパスワードが失効するように設定することができます。これにより、ユーザーはすぐにパスワードを変更せざるを得なくなります。

1.

初期パスワードを設定します。デフォルトのパスワードを割り当てるには、root で次のコマンドを実行します。

```
passwd username
```

警告

passwd ユーティリティーには、null パスワードを設定するオプションがあります。null パスワードの使用は便利ですが、安全ではありません。第三者がログインして、安全でないユーザー名を使用してシステムにアクセスする可能性があるためです。null パスワードの使用は可能な限り避けてください。避けられない場合は、null パスワードでアカウントのロックを解除する前に、ユーザーがログインする準備ができていることを常に確認してください。

2. root で以下のコマンドを実行して、パスワードの即時有効期限を強制します。

```
chage -d 0 username
```

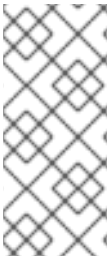
このコマンドは、パスワードが最後に変更された日付の値をエポック (1970 年 1 月 1 日) に設定します。この値は、パスワードエージングポリシーがある場合、それに関係なく、パスワードの即時期限切れを強制します。

最初のログイン時に、ユーザーは新しいパスワードの入力を求められるようになりました。

4.1.2. アカウントのロック

Red Hat Enterprise Linux 7 では、pam_faillock PAM モジュールを使用すると、システム管理者は、指定された回数試行の失敗後にユーザーアカウントをロックアウトできます。ユーザーのログイン試行を制限することは、主に、ユーザーのアカウントパスワードを取得することを目的としたブルートフォース攻撃の可能性を防ぐことを目的としたセキュリティー対策として機能します。

pam_faillock モジュールでは、失敗したログイン試行が、各ユーザーの個別のファイル(/var/run/faillock ディレクトリー内)に保存されます。



注記

失敗した試行のログファイルの行の順序は重要です。この順序を変更すると、even_deny_root オプションが使用されると、root ユーザーアカウントを含むすべてのユーザーアカウントがロックされる可能性があります。

以下の手順でアカウントロックを設定します。

1. **3 回失敗した後に root 以外のユーザーをロックアウトし、10 分後にそのユーザーのロックを解除するには、/etc/pam.d/system- auth ファイルおよび /etc/pam.d/password-auth ファイルの auth セクションに 2 行を追加します。編集後、両方のファイルの auth セクション全体は以下のようになります。**

```
auth        required      pam_env.so
auth        required      pam_faillock.so preauth silent audit deny=3 unlock_time=600
auth        sufficient    pam_unix.so nullok try_first_pass
auth        [default=die] pam_faillock.so authfail audit deny=3 unlock_time=600
auth        requisite     pam_succeed_if.so uid >= 1000 quiet_success
auth        required      pam_deny.so
```

2 行目、4 行目が追加されました。

2. **前の手順で指定した両方のファイルの account セクションに、以下の行を追加します。**

```
account     required      pam_faillock.so
```

3. **root ユーザーのアカウントロックを適用するには、/etc/pam.d/system-auth ファイルおよび /etc/pam.d/password-auth ファイルの pam_faillock エントリーに even_deny_root オプションを追加します。**

```
auth        required      pam_faillock.so preauth silent audit deny=3 even_deny_root
unlock_time=600
auth        sufficient    pam_unix.so nullok try_first_pass
auth        [default=die] pam_faillock.so authfail audit deny=3 even_deny_root
unlock_time=600

account     required      pam_faillock.so
```

ユーザー john が 3 回ログインに失敗した後に 4 回目のログインを試みると、4 回目のアカウントはロックされます。

```
~]$ su - john
Account locked due to 3 failed logins
su: incorrect password
```

ログインに複数回失敗した後でもシステムをロックアウトしないようにするには、/etc/pam.d/system-auth と /etc/pam.d/password-auth の両方で、pam_faillock が初めて呼び出される行のすぐ上に以下の行を追加します。また、user1、user2、および user3 を実際のユーザー名に置き換えます。

```
auth [success=1 default=ignore] pam_succeed_if.so user in user1:user2:user3
```

ユーザーごとの試行失敗回数を表示するには、root で次のコマンドを実行します。

```
~]$ faillock
john:
When                Type  Source                                           Valid
2013-03-05 11:44:14 TTY   pts/0                                               V
```

ユーザーのアカウントをアンロックするには、root で次のコマンドを実行します。

```
faillock --user <username> --reset
```

重要

cron ジョブを実行すると、cron ジョブを実行しているユーザーの pam_faillock の失敗カウンターがリセットされるため、pam_faillock は cron 用に設定しないでください。詳細は、Knowledge Centered Support (KCS) solution を参照してください。

authconfig でのカスタム設定の保持

authconfig ユーティリティーを使用して認証設定を変更する場合、system-auth および password-auth ファイルは authconfig ユーティリティーの設定で上書きされます。これは、authconfig が認識して上書きしない設定ファイルの代わりにシンボリックリンクを作成することで回避できます。設定ファイルと authconfig でカスタム設定を同時に使用するには、以下の手順に従ってアカウントロックを設定します。

1. **system-auth ファイルと password-auth ファイルがすでに system-auth-ac および password-auth-ac を指すシンボリックリンクであるかどうかを確認します（これはシステムのデフォルトです）。**

```
~]# ls -l /etc/pam.d/{password,system}-auth
```

以下のような出力であれば、シンボリックリンクは正しく設定されているので、手順 3 までスキップできます。

```
lrwxrwxrwx. 1 root root 16 24. Feb 09.29 /etc/pam.d/password-auth -> password-auth-ac
lrwxrwxrwx. 1 root root 28 24. Feb 09.29 /etc/pam.d/system-auth -> system-auth-ac
```

system-auth および password-auth ファイルがシンボリックリンクでない場合は、次の手順に進みます。

2. **設定ファイルの名前を変更します。**

```
~]# mv /etc/pam.d/system-auth /etc/pam.d/system-auth-ac
~]# mv /etc/pam.d/password-auth /etc/pam.d/password-auth-ac
```

3. **カスタム設定で設定ファイルを作成します。**

```
~]# vi /etc/pam.d/system-auth-local
```

/etc/pam.d/system-auth-local ファイルには、次の行が含まれている必要があります。

```
auth        required      pam_faillock.so preauth silent audit deny=3 unlock_time=600
auth        include       system-auth-ac
auth        [default=die] pam_faillock.so authfail silent audit deny=3 unlock_time=600

account     required      pam_faillock.so
account     include       system-auth-ac

password    include       system-auth-ac

session     include       system-auth-ac
```

```
~]# vi /etc/pam.d/password-auth-local
```

/etc/pam.d/password-auth-local ファイルには、次の行が含まれている必要があります。

```
auth        required      pam_faillock.so preauth silent audit deny=3 unlock_time=600
auth        include       password-auth-ac
auth        [default=die] pam_faillock.so authfail silent audit deny=3 unlock_time=600

account     required      pam_faillock.so
account     include       password-auth-ac
```

```
password    include       password-auth-ac

session     include       password-auth-ac
```

4. 以下のシンボリックリンクを作成します。

```
~]# ln -sf /etc/pam.d/system-auth-local /etc/pam.d/system-auth
~]# ln -sf /etc/pam.d/password-auth-local /etc/pam.d/password-auth
```

さまざまな pam_faillock 設定オプションの詳細は、pam_faillock(8) の man ページを参照してください。

nullok オプションの削除

/etc/shadow ファイルのパスワードフィールドが空の場合、ユーザーは空のパスワードでログインできるようにする nullok オプションがデフォルトで有効になります。nullok オプションを無効にするには、/etc/pam.d/ ディレクトリーの設定ファイル(/etc/pam.d/ system-auth、/etc/pam.d/password-auth など)から nullok 文字列を削除します。

Will nullok option allow users to login without entering a password? を参照してください。詳細は、KCS ソリューションを参照してください。

4.1.3. セッションのロック

ユーザーは、平日の操作時に多数の理由でワークステーションを無人状態にしておく必要がある場合があります。特に物理的なセキュリティー対策が不十分な環境では、これにより、攻撃者にマシンに物理的にアクセスする機会を提供することになる可能性があります (「物理的コントロール」 を参照)。特にノートパソコンは、その可動性から物理的なセキュリティーが阻害されるため、危険にさらされています。正しいパスワードが入力されるまで、システムにアクセスできないようにするセッションロック機能を使用することで、これらのリスクを軽減することができます。

注記

ログアウトする代わりに画面をロックする主な利点は、ロックによってユーザーのプロセス (ファイル転送など) を実行し続けることができることです。ログアウトしてしまうと、このようなプロセスは中断してしまいます。

4.1.3.1. vlock を使った仮想コンソールのロック

仮想コンソールをロックするには、vlock ユーティリティーを使用します。root で以下のコマンドを入力してインストールします。

```
~]# yum install kbd
```

インストール後に、パラメーターを追加せずに vlock コマンドを使用して、任意のコンソールセッションをロックできます。これは、現在アクティブな仮想コンソールセッションをロックし、他のセッションにはアクセスできるようにします。ワークステーション上のすべての仮想コンソールにアクセスできないようにするには、以下を実行します。

```
vlock -a
```

この場合、vlock は現在アクティブなコンソールをロックし、-a オプションは他の仮想コンソールに切り替えることを防ぎます。

詳細は、vlock (1) man ページを参照してください。

4.1.4. リムーバブルメディアの読み取り専用マウントの強制

リムーバブルメディア(USB フラッシュディスクなど)の読み取り専用マウントを強制するには、udev ルールを使用してリムーバブルメディアを検出し、blockdev ユーティリティーを使用して読み取り専用にマウントするように設定できます。これは、物理メディアの読み取り専用マウントを強制するには十分です。

リムーバブルメディアの読み取り専用マウントを強制するための blockdev の使用法

すべてのリムーバブルメディアを強制的に読み取り専用でマウントするには、/etc/ udev /rules.d/ ディレクトリーに、80-readonly-removables.rules などの新しい udev 設定ファイルを以下の内容で作成します。

```
SUBSYSTEM=="block",ATTRS{removable}=="1",RUN{program}="/sbin/blockdev --setro %N"
```

上記の udev ルールは、新しく接続されたリムーバブルブロック（ストレージ）デバイスが、blockdev ユーティリティーを使用して読み取り専用として自動的に設定されます。

新しい udev 設定の適用

これらの設定を有効にするには、新しい udev ルールを適用する必要があります。udev サービスは設定ファイルへの変更を自動的に検出しますが、既存のデバイスには新しい設定が適用されません。新しい設定の影響を受けるのは、新しく接続されたデバイスのみです。そのため、次に接続したときに新しい設定が適用されるように、接続されているすべてのリムーバブルメディアをアンマウントして取り外す必要があります。

既存のデバイスにすべてのルールを再適用するように udev を強制するには、root で次のコマンド

を実行します。

```
~# udevadm trigger
```

上記のコマンドを使用して udev がすべてのルールを再適用するように強制しても、すでにマウントされているストレージデバイスには影響がないことに注意してください。

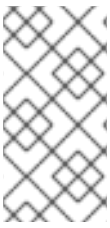
udev がすべてのルールを再ロードするように強制するには（何らかの理由で新しいルールが自動的に検出されない場合）、次のコマンドを使用します。

```
~# udevadm control --reload
```

4.2. ROOT アクセスの制御

ホームマシンを管理する場合、ユーザーは一部のタスクを root ユーザーとして実行するか、sudo や su などの *setuid* プログラムを使用して有効な root 権限を取得する必要があります。setuid プログラムとは、プログラムを操作するユーザーではなく、プログラムの所有者のユーザー ID(UID) で動作するプログラムのことです。このようなプログラムは、以下の例のように、長い形式一覧の所有者セクションの s で示されます。

```
~]$ ls -l /bin/su
-rwsr-xr-x. 1 root root 34904 Mar 10  2011 /bin/su
```



注記

s は大文字または小文字になります。大文字で表示されている場合は、基になる許可ビットが設定されていないことを意味します。

しかし、組織のシステム管理者の場合、組織内のユーザーが自分のマシンに対してどの程度の管理アクセスを持つべきかについて選択する必要があります。pam_console.so と呼ばれる PAM モジュールを介して、リムーバブルメディアの再起動やマウントなど、通常は root ユーザーに対してのみ予約されている一部のアクティビティーは、物理コンソールにログインする最初のユーザーに許可されます。しかし、ネットワーク設定の変更、新しいマウスの設定、ネットワークデバイスのマウントなど、その他の重要なシステム管理タスクは、管理者権限がなければできません。その結果、システム管理者は、ネットワーク上のユーザーがどの程度のアクセスを受けるべきかを決定する必要があります。

4.2.1. Root アクセスの拒否

管理者がこれらの理由や他の理由で root としてログインできないようにする場合は、root パスワードを秘密にして、ブートローダーのパスワード保護によりランレベル 1 または単一ユーザーモードへの

アクセスを禁止する必要があります（本トピックの詳細については、「ブートローダーのセキュア化」を参照してください）。

管理者は、次の 4 つの方法を使用して、root ログインを許可しないようにできます。

root シェルの変更

ユーザーが root として直接ログインできないようにするために、システム管理者は、/etc/passwd ファイルで root アカウントのシェルを /sbin/nologin に設定します。

表4.2 Root シェルの無効化

影響あり	影響なし
root シェルへのアクセスを阻止し、そのような試みをログに記録します。以下のプログラムは root アカウントにアクセスできません。 • login • gdm • kdm • xdm • su • ssh • scp • sftp	FTP クライアント、メールクライアント、多くの setuid プログラムなど、シェルを必要としないプログラム。以下のプログラムは root アカウントにアクセスでき ません。 • sudo • FTP クライアント • Email クライアント

任意のコンソールデバイス (tty) を使用した root アクセスの無効化

root アカウントへのアクセスをさらに制限するために、管理者は /etc/securetty ファイルを編集してコンソールで root ログインを無効にできます。このファイルは、root ユーザーがログインできるすべてのデバイスを一覧表示します。ファイルがまったく存在しない場合、root ユーザーは、

コンソールまたは raw ネットワークインターフェイスを介して、システム上の任意の通信デバイスを介してログインできます。これは危険です。これは、ユーザーが Telnet を使用して root としてマシンにログインできるため、ネットワーク経由でプレーンテキストでパスワードを送信します。

デフォルトでは、Red Hat Enterprise Linux 7 の /etc/securetty ファイルでは、マシンに物理的に接続されているコンソールで root ユーザーがログインすることしかできません。root ユーザーがログインできないようにするには、root でシェルプロンプトで以下のコマンドを入力して、このファイルの内容を削除します。

```
echo > /etc/securetty
```

KDM、GDM、および XDM ログインマネージャーで securetty サポートを有効にするには、以下の行を追加します。

```
auth [user_unknown=ignore success=ok ignore=ignore default=bad] pam_securetty.so
```

追加対象のファイルは以下のとおりです。

- /etc/pam.d/gdm
- /etc/pam.d/gdm-autologin
- /etc/pam.d/gdm-fingerprint
- /etc/pam.d/gdm-password
- /etc/pam.d/gdm-smartcard
- /etc/pam.d/kdm
- /etc/pam.d/kdm-np
- /etc/pam.d/xdm

警告

空の /etc/securetty ファイルでは、root ユーザーが OpenSSH ツールスイートを使用してリモートでログインできません。これは、認証後までコンソールが開かれないためです。

表4.3 Root ログインの無効化

影響あり	影響なし
コンソールまたはネットワークを使用して root アカウントにアクセスできないようにします。以下のプログラムは root アカウントにアクセスできません。 • login • gdm • kdm • xdm • tty を開くその他のネットワークサービス	root としてログインしないが、setuid またはその他のメカニズムを使用して管理タスクを実行するプログラム。以下のプログラムは root アカウントにアクセスできません。 • su • sudo • ssh • scp • sftp

Root SSH ログインの無効化

SSH プロトコルを介した root ログインを防ぐには、SSH デーモンの設定ファイル /etc/ssh/sshd_config を編集し、以下の行を変更します。

```
#PermitRootLogin yes
```

の行を以下のように変更します。

```
PermitRootLogin no
```

表4.4 Root SSH ログインの無効化

影響あり	影響なし
ツールの OpenSSH スイートを使用した root アクセスを防ぎます。以下のプログラムは root アカウントにアクセスできません。 • ssh • scp • sftp	OpenSSH のツール群に含まれないプログラム。

PAM を使用して、サービスへの root アクセスを制限する

PAM は、/lib/security/pam_listfile.so モジュールを通じて、特定のアカウントを拒否するための優れた柔軟性を提供します。管理者は、このモジュールを使用して、ログインを許可されていないユーザーのリストを参照できます。システムサービスへの root アクセスを制限するには、/etc/pam.d/ ディレクトリーの target サービスのファイルを編集し、pam_listfile.so モジュールが認証に必要であることを確認します。

以下は、/etc/pam.d/vsftpd PAM 設定ファイルの vsftpd FTP サーバーに モジュールを使用する方法の例です（ディレクティブが 1 行にある場合は最初の行の最後にある \ 文字は必要あり ません ）。

```
auth    required    /lib/security/pam_listfile.so    item=user \
sense=deny file=/etc/vsftpd.ftpusers onerr=succeed
```

これにより、PAM は /etc/vsftpd.ftpusers ファイルを参照し、一覧表示されたユーザーのサービスへのアクセスを拒否します。管理者はこのファイルの名前を変更することができ、各サービスごとに個別のリストを保持することも、1 つの中央リストを使用して複数のサービスへのアクセスを拒否することもできます。

管理者が複数のサービスへのアクセスを拒否する場合は、同様の行を PAM 設定ファイル（メールクライアントの場合は /etc/pam.d/pop および /etc/pam.d/imap、SSH クライアントの場合は /etc/pam.d/ssh など）に追加できます。

PAM の詳細は、/usr/share/doc/pam-<version>/html/ ディレクトリーにある 『The Linux-』PAM System Administrator's Guide を参照してください。

表4.5 PAM を使用した root の無効化

影響あり	影響なし

影響あり	影響なし
PAM が認識しているネットワークサービスへの root アクセスを防ぎます。以下のサービスは、root アカウントにアクセスできません。 • login • gdm • kdm • xdm • ssh • scp • sftp • FTP クライアント • Email クライアント • すべての PAM 対応サービス	PAM を意識していないプログラム、サービス。

影響あり	影響なし

4.2.2. Root アクセスの許可

組織内のユーザーが信頼され、コンピューターネーカレートである場合は、root アクセスを許可することが問題ではない可能性があります。ユーザーによる root アクセスの許可とは、デバイスの追加やネットワークインターフェイスの設定などのマイナーなアクティビティーを個々のユーザーが処理できるため、システム管理者はネットワークセキュリティーやその他の重要な問題に対処できます。

一方、個々のユーザーに root アクセスを付与すると、次の問題が発生する可能性があります。

- マシン の設定ミス - root アクセスを持つユーザーはマシンの設定を誤設定でき、問題の解決に支援が必要になる場合があります。さらに悪いことに、知らずにセキュリティーホールを発生させてしまう可能性があります。

- 安全で ないサービス の実行 - root アクセスを持つユーザーは、FTP や Telnet などのマシン上で安全でないサーバーを実行する可能性があり、ユーザー名とパスワードが危険にさらされる可能性があります。これらのサービスは、この情報をプレーンテキストでネットワーク経由で送信します。

- 電子メールの添付ファイルを root で実行 — まれにですが、Linux に影響を与える電子メールウィルスが存在します。悪意のあるプログラムは、root ユーザーによって実行されると、最大の脅威となります。

- 監査証跡の維持 - root アカウントは複数のユーザーが共有されるため、複数のシステム管理者がシステムを保守できるため、特定の時点でどのユーザーが root であったかを把握することはできません。セパレートログインの場合、ユーザーがログインしたアカウントと、セッション追跡のための一意の番号がタスク構造に入れられ、ユーザーが起動するすべてのプロセスに継承されます。同時ログインを使用する場合、一意の番号を使用して、特定のログインへのアクションを追跡することができます。アクションが監査イベントを生成すると、その一意な番号に関連するログインアカウントとセッションが記録されます。これらのログインとセッションを表示するには、aulast コマンドを使用します。aulast コマンドの --proof オプションを使用すると、特定の ausearch クエリーを提案し、特定のセッションによって生成された監査可能なイベントを分離できます。監査システムの詳細については、7章 *システム監査*を参照してください。

4.2.3. Root アクセスの制限

管理者は、root ユーザーへのアクセスを拒否するのではなく、su または sudo などの setuid プログラムを介したアクセスのみを許可したい場合があります。su および sudo の詳細は、Red Hat

Enterprise Linux 7 System Administrator's Guide の Gaining Privileges の章および su (1) および sudo (8)の man ページを参照してください。

4.2.4. 自動ログアウトの有効化

ユーザーが root としてログインすると、無人ログインセッションが重大なセキュリティーリスクを引き起こす可能性があります。このリスクを減らすために、一定時間後にアイドル状態のユーザーを自動的にログアウトさせるようにシステムを設定することができます。

1. root として、/etc/profile ファイルの先頭に次の行を追加して、このファイルの処理が中断されないようにします。

```
trap "" 1 2 3 15
```

2. root として、以下の行を /etc/profile ファイルに追加し、120 秒後に自動的にログアウトします。

```
export TMOUT=120
readonly TMOUT
```

TMOUT 変数は、指定された秒数のアクティビティーがない場合にシェルを終了します（上記の例では 120 に設定）。特定のインストールのニーズに応じて、制限を変更することができます。

4.2.5. ブートローダーのセキュア化

Linux ブートローダーをパスワードで保護する主な理由は、以下のとおりです。

1. シングルユーザーモードへのアクセスの防止 - 攻撃者がシステムをシングルユーザーモードで起動できる場合、root パスワードを求められることなく、root として自動的にログインします。

警告

/etc/sysconfig/init ファイルの *SINGLE* パラメーターを編集して、パスワードを使用してシングルユーザーモードへのアクセスを保護することは推奨されません。攻撃者は、GRUB 2 のカーネルコマンドラインでカスタム初期コマンド(*init=* パラメーターを使用)を指定することにより、パスワードをバイパスできます。Red Hat Enterprise Linux 7 System Administrator's Guide の Protecting GRUB 2 with a Password の章に記載されているように、GRUB 2 ブートローダーをパスワードで保護することが推奨されています。

2. **GRUB 2 コンソールへのアクセスの防止 - マシンがブートローダーとして GRUB 2 を使用する場合、攻撃者は GRUB 2 エディターインターフェイスを使用して設定を変更したり、cat コマンドを使用して情報を収集したりできます。**

3. **安全でないオペレーティングシステムへのアクセスの防止 — デュアルブートシステムの場合、攻撃者は、たとえば DOS のような、アクセス制御およびファイルパーミッションを無視するオペレーティングシステムを起動時に選択することができます。**

Red Hat Enterprise Linux 7 には、Intel 64 および AMD64 プラットフォームに GRUB 2 ブートローダーが含まれています。GRUB 2 の詳細については、Red Hat Enterprise Linux 7 System Administrator's Guide の Working With the GRUB 2 Boot Loader の章を参照してください。

4.2.5.1. インタラクティブスタートアップの無効化

ブートシーケンスの最初に I キーを押すと、対話的にシステムを起動できます。インタラクティブな起動中に、システムは各サービスを 1 つずつ起動するように要求します。しかし、これでは、お客様のシステムに物理的にアクセスした攻撃者が、セキュリティー関連サービスを無効化し、システムにアクセスすることができる可能性があります。

root としてユーザーが対話的にシステムを起動しないようにするには、/etc/sysconfig/init ファイルの *PROMPT* パラメーターを無効にします。

```
PROMPT=no
```

4.2.6. ハードリンクおよびシンボリックリンクの保護

悪意のあるユーザーが保護されていないハードリンクやシンボリックリンクによって引き起こされる潜在的な脆弱性を悪用するのを防ぐため、Red Hat Enterprise Linux 7 には、特定の条件を満たす場合にのみリンクの作成または追跡を許可する機能が搭載されています。

ハードリンクの場合、次のいずれかが当てはまる必要があります。

- ユーザーは、リンク先のファイルを所有しています。
- ユーザーは、リンク先のファイルに対して、すでに読み取りと書き込みのアクセス権を持っています。

シンボリックリンクの場合、プロセスは、スティッキービットを持つ誰でも書き込み可能なディレクトリーの外部にある場合にのみリンクをたどることが許可されます。または、以下のいずれかが当てはまる必要があります。

- シンボリックリンクに続くプロセスが、シンボリックリンクの所有者となります。
- ディレクトリーの所有者は、シンボリックリンクの所有者と同じになります。

この保護機能は、デフォルトでオンになっています。これは、/usr/lib/sysctl.d/50-default.conf ファイルの以下のオプションによって制御されます。

```
fs.protected_hardlinks = 1
fs.protected_symlinks = 1
```

デフォルト設定を上書きして保護を無効にするには、以下の内容で /etc/sysctl .d/ ディレクトリーに 51-no-protect-links.conf などの新しい設定ファイルを作成します。

```
fs.protected_hardlinks = 0
fs.protected_symlinks = 0
```

注記

デフォルトのシステム設定を上書きするには、新しい設定ファイルに .conf 拡張子が必要で、デフォルトのシステムファイルの 後 に読み込む必要があることに注意してください（ファイルは辞書的な順序で読み取られます）。そのため、ファイル名の最初に番号が大きいファイルに含まれる設定が優先されます。

sysctl メカニズムを使用した起動時のカーネルパラメーターの設定についての詳細は、sysctl.d(5) の man ページを参照してください。

4.3. サービスのセキュア化

組織内のシステム管理者にとって、管理コントロールへのユーザーアクセスは重要な問題ですが、Linux システムを管理および運用するすべての人にとって、どのネットワークサービスがアクティブであるかを監視することは最も重要なことです。

Red Hat Enterprise Linux 7 のサービスの多くは、ネットワークサーバーです。ネットワークサービスがマシン上で実行されている場合、サーバーアプリケーション (*デーモン* と呼ばれる) は、1 つ以上のネットワークポートで接続をリッスンしています。これらの各サーバーは、潜在的な攻撃経路として扱われる必要があります。

4.3.1. サービスへのリスク

ネットワークサービスは、Linux システムに多くのリスクをもたらす可能性があります。以下は、一部の主要な問題の一覧になります。

- サービス拒否攻撃 (DoS) — サービスにリクエストをフラッディングさせると、サービスは各リクエストをログに記録して応答しようとするため、サービス拒否攻撃はシステムを使用不能にすることができます。

- 分散型サービス拒否攻撃 (DDoS) — DoS 攻撃の一種で、セキュリティーを侵害された複数のマシン (多くは数千台以上) を使用して、サービスに協調攻撃を仕掛け、リクエストをフラッディングさせて使用不能にするものです。

- スクリプト脆弱性攻撃 — Web サーバーが一般的に行うように、サーバーが、サーバーサイドのアクションを実行するためにスクリプトを使用している場合、攻撃者は不適切に記述されたスクリプトをターゲットにすることができます。これらのスクリプトの脆弱性攻撃により、バッファーオーバーフロー状態が発生したり、攻撃者がシステム上のファイルを変更したりできます。

- バッファーオーバーフロー攻撃 - ポート 1 から 1023 をリッスンするサービスは、管理者権限で開始するか、CAP_NET_BIND_SERVICE 機能を設定する必要があります。プロセスがポートにバインドされ、そのポートをリッスンするようになると、権限または機能がドロップされることがよくあります。権限や機能がドロップされず、アプリケーションに悪用可能なバッファーオーバーフローがある場合、攻撃者はデーモンを実行しているユーザーとしてシステムにアクセスできる可能性があります。悪用可能なバッファーオーバーフローが存在するため、クラッカーは自動化ツールを使って脆弱性のあるシステムを特定し、アクセス権を獲得した後は、自動化ルートキットを使ってシステムへのアクセス権を維持します。

注記

バッファーオーバーフローの脆弱性の脅威は、*ExecShield* によって Red Hat Enterprise Linux 7 で軽減されます。これは、x86 互換のユニプロセッサーおよびマルチプロセッサーカーネルでサポートされる実行可能メモリーのセグメンテーションおよび保護テクノロジーです。ExecShield は、仮想メモリーを実行可能セグメントと非実行セグメントに分離することで、バッファーオーバーフローのリスクを低減します。実行可能セグメントの外で実行しようとするプログラムコード (バッファーオーバーフローの悪用から注入された悪意のあるコードなど) は、セグメンテーションフォールトを引き起こし、終了します。

Execshield には、AMD64 プラットフォームおよび Intel® 64 システムにおける *No eXecute* (NX) テクノロジーのサポートも含まれます。これらのテクノロジーは ExecShield と連携して動作し、4KB の実行可能コードの粒度で仮想メモリーの実行可能部分で悪意のあるコードが実行されるのを防ぎ、バッファーオーバーフローの悪用による攻撃のリスクを低減します。

重要

ネットワークに対する攻撃に対する公開を制限するには、未使用のすべてのサービスをオフにする必要があります。

4.3.2. サービスの識別と設定

セキュリティーを強化するため、Red Hat Enterprise Linux 7 でインストールされるほとんどのネットワークサービスは、デフォルトでオフになっています。ただし、いくつかの注目すべき例外があります。

- cups - Red Hat Enterprise Linux 7 のデフォルトのプリントサーバー。
- cups-lpd - 代替のプリントサーバー。

- xinetd - gssftp や telnet など、さまざまな下位サーバーへの接続を制御するスーパーサーバー。

- sshd - OpenSSH サーバー。Telnet の安全な代替です。

これらのサービスを実行したままにするかどうかを決定するときは、常識的に考え、リスクを冒さないようにすることが最善策となります。たとえば、プリンターが利用できない場合は、cups を実行したままにしないでください。同じことが portreserve にも当てはまります。NFSv3 ボリュームをマウントしない、または NIS (ypbind サービス)を使用する場合は、rpcbind を無効にする必要があります。起動時にどのネットワークサービスが起動可能かを確認するだけでは十分ではありません。また、どのポートが開いていて、リッスンしているかも確認することをお勧めします。詳細は、「リッスンしているポートの確認」 を参照してください。

4.3.3. 安全でないサービス

潜在的に、どのようななネットワークサービスも安全ではありません。そのため、使っていないサービスをオフにすることはとても重要です。サービスの不正使用は定期的に確認され、パッチが適用されるため、ネットワークサービスに関連するパッケージを定期的に更新することが非常に重要です。詳細は、*3章 システムを最新の状態に保つ* を参照してください。

一部のネットワークプロトコルは、本質的に他のプロトコルよりも安全ではありません。以下の動作を実行するサービスがそれに当たります。

- 暗号化されていないネットワーク上でのユーザー名とパスワードの送信 — Telnet や FTP などの多くの古いプロトコルは、認証セッションを暗号化しないため、可能な限り避ける必要があります。

- 暗号化されていないネットワーク上での機密データの送信 — 多くのプロトコルは、暗号化されていないネットワーク上でデータを送信します。これらのプロトコルには、Telnet、FTP、HTTP、および SMTP が含まれます。NFS や SMB などの多くのネットワークファイルシステムも、暗号化されていないネットワークを介して情報を送信します。これらのプロトコルを使用する場合、ユーザーの責任において、送信されるデータの種類を制限する必要があります。

本質的に安全ではないサービスの例としては、rlogin、rsh、telnet、vsftpd などがあります。

SSH を優先して、すべてのリモートログインおよびシェルプログラム(rlogin、rsh、および telnet)を回避する必要があります。sshd の詳細は、「SSH のセキュア化」 を参照してください。

FTP は、リモートシェルほどシステムのセキュリティーに本質的に危険ではありませんが、問題を回避するために、FTP サーバーは慎重に設定し、監視する必要があります。FTP サーバーのセキュリティー保護に関する詳細は、「FTP のセキュア化」 を参照してください。

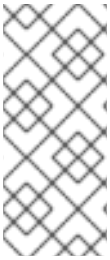
慎重に実装し、ファイアウォールの内側に置くべきサービスは以下の通りです。

- auth
- nfs-server
- smb および nbm (Samba)
- yppasswdd
- ypserv
- ypxfrd

ネットワークサービスのセキュリティーに関する詳細は、「ネットワークアクセスのセキュア化」を参照してください。

4.3.4. rpcbind のセキュア化

rpcbind サービスは、NIS や NFS などの RPC サービス用の動的ポート割り当てデーモンです。認証メカニズムが弱く、制御するサービスに幅広いポート範囲を割り当てることができます。これらの理由から、セキュリティー保護することは困難です。



注記

rpcbind のセキュリティーは、NFSv4 では不要になったため、NFSv2 および NFSv3 の実装にのみ影響します。NFSv2 または NFSv3 サーバーを実装する予定の場合は、rpcbind が必要で、以下のセクションが適用されます。

RPC サービスを実行している場合は、以下の基本的なルールにしたがってください。

4.3.4.1. TCP Wrapper による rpcbind の保護

rpcbind サービスには認証形式がないため、TCP Wrapper を使用して、rpcbind サービスにアクセスできるネットワークまたはホストを制限することが重要です。

また、サービスへのアクセスを制限する場合は、IP アドレス のみ を使用してください。ホスト名は、DNS ポイズニングなどの方法で偽造される可能性があるため、使用しないようにしてください。

4.3.4.2. firewalld による rpcbind の保護

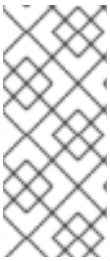
rpcbind サービスへのアクセスをさらに制限するには、サーバーに firewalld ルールを追加し、特定のネットワークへのアクセスを制限することが推奨されます。

以下は、firewalld リッチ言語コマンドの例です。1 つ目は、192.168.0.0/24 ネットワークからポート 111 (rpcbind サービスで使用される)への TCP 接続を許可します。2 つ目は、localhost から同じポートへの TCP 接続を許可します。 それ以外のパケットはすべてドロップされます。

```
~]# firewall-cmd --add-rich-rule='rule family="ipv4" port port="111" protocol="tcp" source
address="192.168.0.0/24" invert="True" drop'
~]# firewall-cmd --add-rich-rule='rule family="ipv4" port port="111" protocol="tcp" source
address="127.0.0.1" accept'
```

同様に UDP トラフィックを制限するには、次のコマンドを使用します。

```
~]# firewall-cmd --add-rich-rule='rule family="ipv4" port port="111" protocol="udp" source
address="192.168.0.0/24" invert="True" drop'
```



注記

firewalld リッチ言語コマンドに --permanent を追加して、設定を永続化します。ファイアウォールの実装に関する詳細情報は、*5章 ファイアウォールの使用* を参照してください。

4.3.5. rpc.mountd のセキュア化

rpc.mountd デーモンは、NFS バージョン 2 (RFC 1904)および NFS バージョン 3 (『RFC』1813)が使用するプロトコルである NFS MOUNT プロトコルのサーバー側を実装します。

RPC サービスを実行している場合は、以下の基本的なルールにしたがってください。

4.3.5.1. TCP ラッパーによる rpc.mountd の保護

rpc.mountd サービスには認証が組み込まれていないため、TCP Wrapper を使用して、rpc.mountd サービスにアクセスできるネットワークまたはホストを制限することが重要です。

さらに、サービスへのアクセスを制限する場合は、IP アドレス のみ を使用してください。ホスト名は DNS ポイズニングなどの方法で偽造される可能性があるため、使用しないでください。

4.3.5.2. firewalld による rpc.mountd の保護

rpc.mountd サービスへのアクセスをさらに制限するには、サーバーに firewalld リッチ言語ルールを追加し、特定のネットワークへのアクセスを制限します。

以下は、firewalld リッチ言語コマンドの例です。1 つ目は、192.168.0.0/24 ネットワークから mountd 接続を許可します。2 つ目は、ローカルホストからの mountd 接続を許可します。他のパケットはすべて遮断されます。

```
~]# firewall-cmd --add-rich-rule 'rule family="ipv4" source NOT address="192.168.0.0/24" service
name="mountd" drop'
~]# firewall-cmd --add-rich-rule 'rule family="ipv4" source address="127.0.0.1" service
name="mountd" accept'
```

注記

firewalld リッチ言語コマンドに --permanent を追加して、設定を永続化します。ファイアウォールの実装に関する詳細情報は、*5章 ファイアウォールの使用* を参照してください。

4.3.6. NIS のセキュア化

ネットワーク情報 サービス(NIS)は、ypserv と呼ばれる RPC サービスです。このサービスは、rpcbind や他の関連サービスと組み合わせて使用され、ユーザー名、パスワード、その他の機密情報を、そのドメイン内で要求するコンピューターに配布します。

NIS サーバーは、いくつかのアプリケーションで設定されています。内容は以下の通りです。

- /usr/sbin/rpc.yppasswdd - yppasswdd サービスとも呼ばれるこのデーモンにより、ユーザーは NIS パスワードを変更できます。
- /usr/sbin/rpc.ypxfrd - ypxfrd サービスとも呼ばれるこのデーモンは、ネットワークを介した NIS マップ転送を行います。
- /usr/sbin/ypserv - これは NIS サーバーデーモンです。

NIS は現在の基準からすると、やや安全性に欠けています。ホスト認証メカニズムがなく、パスワードハッシュを含むすべての情報を暗号化せずにネットワーク経由で送信します。そのため、NIS を使用するネットワークを構築する際には、細心の注意が必要です。さらに、NIS のデフォルト設定が本質的に安全でないという事実が、この問題をさらに複雑にしています。

NIS サーバーの実装を計画している人は、「rpcbind のセキュア化」 に概説されているように、最初に rpcbind サービスをセキュリティー保護し、ネットワーク計画などの以下の問題に対処することが推奨されます。

4.3.6.1. ネットワークの注意深いプランニング

NIS はネットワーク上で機密情報を暗号化せずに送信するため、サービスをファイアウォールの内側で、そしてセグメント化された安全なネットワーク上で実行することが重要となります。NIS 情報が安全でないネットワークを介して送信される場合は常に、傍受されるリスクがあります。慎重なネットワーク設計は、深刻なセキュリティー侵害を防ぐ上で役立ちます。

4.3.6.2. パスワードのような NIS ドメイン名とホスト名の使用

ユーザーが NIS サーバーの DNS ホスト名と NIS ドメイン名を知っている限り、NIS ドメイン内のすべてのマシンは、コマンドを使用して認証なしでサーバーから情報を抽出できます。

たとえば、あるユーザーがラップトップコンピューターをネットワークに接続するか、外部からネットワークに分割した場合（また内部 IP アドレスのスプーフィングを管理）、以下のコマンドで /etc/passwd マップを表示します。

```
ypcat -d <NIS_domain> -h <DNS_hostname> passwd
```

この攻撃者が root ユーザーである場合、次のコマンドを入力して /etc/shadow ファイルを取得できます。

```
ypcat -d <NIS_domain> -h <DNS_hostname> shadow
```

注記

Kerberos を使用すると、/etc/shadow ファイルは NIS マップに保存されません。

攻撃者にとって NIS マップへのアクセスを困難にするには、o7hfawtgmhwg.domain.com などの DNS ホスト名にランダムな文字列を作成します。同様に、異なるランダムな NIS ドメイン名を作成します。これにより、攻撃者が NIS サーバーにアクセスすることがより困難になります。

4.3.6.3. /var/yp/securenets ファイルの編集

/var/yp/securenets ファイルが空白であるか、存在しない場合（デフォルトのインストール後の場合）、NIS はすべてのネットワークをリッスンします。最初の作業の 1 つは、ネットマスク/ネットワークのペアを ファイルに配置して、ypserv が適切なネットワークからの要求にのみ応答するようにすることです。

以下は、/var/yp/securenets ファイルからのエントリーの例です。

```
255.255.255.0     192.168.0.0
```

警告

/var/yp/securenets ファイルを作成せずに、NIS サーバーを初めて起動しないでください。

この手法では、IP スプーフィング攻撃からの保護はできませんが、少なくとも NIS サーバーがサービスを提供するネットワークに制限を設けることができます。

4.3.6.4. 静的ポートの割り当てとリッチ言語ルールの使用

NIS に関連するサーバーはすべて、rpc.yppasswdd（ユーザーがログインパスワードを変更できるようにするデーモン）以外の特定のポートを割り当てることができます。他の 2 つの NIS サーバーデーモンである rpc.ypxfrd および ypserv にポートを割り当てると、NIS サーバーデーモンを侵入者からさらに保護するためのファイアウォールルールを作成できます。

これを行うには、以下の行を /etc/sysconfig/network に追加します。

```
YPSERV_ARGS="-p 834"
YPXFRD_ARGS="-p 835"
```

次に、以下のリッチ言語の firewalld ルールを使用して、サーバーがこれらのポートをリッスンするネットワークを適用できます。

```
~]# firewall-cmd --add-rich-rule='rule family="ipv4" source address="192.168.0.0/24" invert="True"
port port="834-835" protocol="tcp" drop'
~]# firewall-cmd --add-rich-rule='rule family="ipv4" source address="192.168.0.0/24" invert="True"
port port="834-835" protocol="udp" drop'
```

これは、リクエストが 192.168.0.0/24 ネットワークから送信された場合にのみ、サーバーがポート 834 および 835 への接続を許可することを意味します。最初のルールは TCP 用で、2 つ目は UDP のルールです。

注記

iptables コマンドによるファイアウォールの実装についての詳細は、*5章 ファイアウォールの使用* を参照してください。

4.3.6.5. Kerberos 認証の使用

NIS を認証に使用する際に考慮すべき問題の 1 つは、ユーザーがマシンにログインするたびに、/etc/shadow マップからのパスワードハッシュがネットワーク経由で送信されることです。侵入者が NIS ドメインにアクセスし、ネットワークトラフィックを盗聴した場合、侵入者はユーザー名とパスワードハッシュを収集できます。十分な時間があれば、パスワードクラッキングプログラムは弱いパスワードを推測することができ、攻撃者はネットワーク上の有効なアカウントにアクセスすることができます。

Kerberos は秘密鍵暗号を使用しているので、パスワードハッシュがネットワーク経由で送信されることはなく、システムの安全性が大幅に向上します。Kerberos の詳細については、Linux ドメイン ID、認証、およびポリシーガイドのLogging into IdM Using Kerberosの項を参照してください。

4.3.7. NFS のセキュア化

重要

NFS トラフィックは、すべてのバージョンで TCP を使用して送信することができ、それは UDP ではなく、NFSv3 で使用されるべきであり、NFSv4 を使用する場合は必須となります。NFS のすべてのバージョンは、RPCSEC_GSS カーネルモジュールの一部として Kerberos ユーザーおよびグループ認証をサポートします。Red Hat Enterprise Linux 7 は rpcbind を使用する NFSv3 をサポートしているため、rpcbind に関する情報は引き続き含まれています。

4.3.7.1. ネットワークの注意深いプランニング

NFSv2 と NFSv3 ではこれまで、データの受け渡しは安全に行われていませんでした。NFS のすべてのバージョンで、Kerberos を使用して通常のファイルシステム操作を認証する (オプションで暗号化する) 機能が追加されました。NFSv4 ではすべての操作で Kerberos を使用できますが、NFSv2 または NFSv3 では、ファイルのロックとマウントではまだ使用できません。NFSv4.0 を使用する場合で、クライアントが NAT やファイアウォールの内側にある場合は、委譲がオフになることがあります。NFSv4.1 を使用して NAT やファイアウォールを介して委譲を操作できるようにする方法の詳細については、Red Hat Enterprise Linux 7 ストレージ管理ガイドの pNFS のセクションを参照してください。

4.3.7.2. NFS マウントオプションのセキュア化

/etc/fstab ファイルでの mount コマンドの使用については、Red Hat Enterprise Linux 7 Storage Administration Guide の Using the mount command の章で説明されています。セキュリティー管理の観点からは、NFS マウントオプションは /etc/nfsmount.conf でも指定できます。これは、カスタムのデフォルトオプションを設定するために使用できます。

4.3.7.2.1. NFS サーバーのレビュー

警告

ファイルシステム全体のみをエクスポートします。ファイルシステムのサブディレクトリーをエクスポートすると、セキュリティー上の問題が発生することがあります。クライアントがファイルシステムのエクスポートされた部分から抜け出して、エクスポートされていない部分を取得する場合があります(exports (5) man ページのサブツリーチェックに関するセクションを参照してください)。

ro オプションを使用し、可能な限りファイルシステムを読み取り専用としてエクスポートし、マウントされたファイルシステムに書き込むことができるユーザー数を削減します。特に必要な場合にのみ rw オプションを使用してください。詳細は、exports (5) の man ページを参照してください。書き込

みアクセスを許可すると、たとえばシンボリックリンク攻撃によるリスクが高まります。これには、/tmp や /usr /tmp などの一時ディレクトリーが含まれます。

ディレクトリーを rw オプションでマウントする必要がある場合は、リスクを減らすために、可能な限りディレクトリーを誰でも書き込み可能にすることは避けてください。一部のアプリケーションでは、パスワードをクリアテキストまたは弱い暗号化形式で保存するため、ホームディレクトリーのエクスポートもリスクがあると見なされます。このリスクは、アプリケーションコードの見直しや改善によって減少しています。SSH 鍵にパスワードを設定しないユーザーもいるので、これもホームディレクトリーのリスクとなることを意味します。パスワードの使用を強制するか、Kerberos を使用すると、このリスクが軽減されます。

エクスポートはアクセスを必要とするクライアントのみとしてください。NFS サーバーで showmount -e コマンドを使用して、サーバーのエクスポート内容を確認します。特に必要のないものはエクスポートしないでください。

no_root_squash オプションを使用しないでください。また、既存のインストールを確認し、使用されていないことを確認してください。詳細は、「no_root_squash オプションを使用しないでください」 を参照してください。

secure オプションは、「予約済み」 ポートへのエクスポートを制限するために使用されるサーバー側のエクスポートオプションです。デフォルトでは、サーバーは 「予約済み」 ポート (1024 未満の番号のポート) からのクライアント通信のみを許可します。これは、従来からクライアントは、「信頼できる」 コード (カーネル内の NFS クライアントなど) のみがこれらのポートを使用するように許可していたためです。しかし、多くのネットワークでは、一部のクライアントで root ユーザーになることは難しくないので、予約されたポートからの通信が権限付きであるとサーバーが想定することは安全ではありません。そのため、予約ポートの制限は効果が限定的です。Kerberos、ファイアウォール、および特定クライアントへのエクスポートを制限することに依存すると良いでしょう。

現在でもほとんどのクライアントは、可能な限り予約済みポートを使用します。ただし、予約済みポートは限られたリソースであるため、クライアント (特に、NFS マウントの数が多いクライアント) は、より大きな番号のポートを使用することも選択できます。Linux クライアントは、「noresvport」マウントオプションを使用して、これを行うことができます。エクスポート時にこれを許可したい場合は、「Insecure」エクスポートオプションで許可することができます。

ユーザーのサーバーへのログインを許可しないことが推奨されます。NFS サーバーで上記の設定を確認する場合、誰および何がサーバーにアクセスできるかを確認します。

4.3.7.2.2. NFS クライアントのレビュー

nosuid オプションを使用して、setuid プログラムの使用を無効にします。nosuid オプションは、set-user-identifier ビットまたは set-group-identifier ビットを無効にします。これにより、リ

モートユーザーは、setuid プログラムを実行してより高い権限を取得できなくなります。このオプションは、クライアント側とサーバー側で使用します。

noexec オプションは、クライアント上のすべての実行可能ファイルを無効にします。これを使用して、ユーザーが共有されているファイルシステムに置かれているファイルを誤って実行しないようにします。nosuid および noexec オプションは、すべてではないにしても、ほとんどのファイルシステムの標準オプションです。

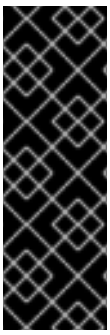
nodev オプションを使用して、「デバイスファイル」 がクライアントによってハードウェアデバイスとして処理されないようにします。

resvport オプションはクライアント側のマウントオプションで、secure は対応するサーバー側のエクスポートオプションです (上記の説明を参照してください)。これは、通信を予約済みポートに制限します。予約済みポートまたは既知のポートは、root ユーザーなどの特権ユーザーおよびプロセス用に予約されています。このオプションを設定すると、クライアントは予約済みソースポートを使用してサーバーと通信します。

NFS のすべてのバージョンで、Kerberos 認証を使用したマウントがサポートされるようになりました。これを有効にするためのマウントオプションは、sec=krb5 です。

NFSv4 では、整合性にkrb5i、プライバシー保護にkrb5pを用いた Kerberos によるマウントをサポートしています。これらは、sec=krb5でマウントする際に使用されますが、NFS サーバーで設定する必要があります。詳細は、exports の man ページを参照してください(man 5 exports)。

NFS の man ページ(man 5 nfs)には、NFSv4 のセキュリティー強化を説明する 「SECURITY CONSIDERATIONS」 セクションがあり、NFS 固有のマウントオプションがすべて含まれています。



重要

krb5-libs パッケージが提供する MIT Kerberos ライブラリーは、新しいデプロイメントで Data Encryption Standard (DES) アルゴリズムに対応しなくなりました。セキュリティーおよび互換性上の理由から、DES は非推奨となり、Kerberos ライブラリーではデフォルトで無効になっています。お使いの環境がより新しく安全なアルゴリズムをサポートしていない場合、互換性の理由に限り DES を使用してください。

4.3.7.3. 構文エラーに注意

NFS サーバーは、/etc/exports ファイルを使って、エクスポートするファイルシステムと、これらのディレクトリーをエクスポートするホストを決定します。このファイルを編集する際には、余計なス

ペースを加えないように注意してください。

たとえば、/etc/exports ファイルの次の行は、ディレクトリー /tmp/nfs/ を、読み取り/書き込みパーミッションでホスト bob.example.com と共有します。

```
/tmp/nfs/     bob.example.com(rw)
```

一方、/etc/exports ファイルの次の行は、同じディレクトリーを読み取り専用パーミッションでホスト bob.example.com と共有し、ホスト名の後の 1 つのスペース文字が原因で読み取り/書き込みパーミッションで ワールド と共有します。

```
/tmp/nfs/     bob.example.com (rw)
```

showmount コマンドを使用して、設定されている NFS 共有を確認し、共有しているものを確認することをお勧めします。

```
showmount -e <hostname>
```

4.3.7.4. no_root_squash オプションを使用しないでください

デフォルトでは、NFS 共有は root ユーザーを非特権ユーザーアカウントである nfsnobody ユーザーに変更します。これにより、root で作成されたすべてのファイルの所有者が nfsnobody に変更され、setuid ビットが設定されたプログラムのアップロードができなくなります。

no_root_squash を使用すると、リモートの root ユーザーは共有ファイルシステム上の任意のファイルを変更し、他のユーザーが誤って実行できるようにアプリケーションが Trojans に感染した状態にすることができます。

4.3.7.5. NFS ファイアウォールの設定

NFSv4 は、NFS for Red Hat Enterprise Linux 7 のデフォルトバージョンで、TCP 用にポート 2049 が開かれていることだけが必要です。NFSv3 を使用する場合は、以下で説明するように 4 つの追加ポートが必要です。

NFSv3 用のポート設定

NFS に使用されるポートは rpcbind サービスによって動的に割り当てられるため、ファイアウォールルールの作成時に問題が発生する可能性があります。このプロセスを簡素化するには、/etc/sysconfig/nfs ファイルを使用して、使用するポートを指定します。

- MOUNTD_PORT — mountd (rpc.mountd) 用の TCP および UDP ポート
- STATD_PORT — ステータス (rpc.statd) 用の TCP および UDP ポート

Red Hat Enterprise Linux 7 では、/etc/modprobe.d/lockd.conf ファイルに NFS ロックマネージャー(nlockmgr)の TCP および UDP ポートを設定します。

- nlm_tcpport — nlockmgr (rpc.lockd) 用の TCP ポート
- nlm_udpport — UDP ポート nlockmgr (rpc.lockd)

指定されたポート番号は、他のサービスでは使用しないでください。指定されたポート番号と、TCP および UDP のポート 2049(NFS) を許可するようにファイアウォールを設定します。カスタマイズ可能な追加の NFS ロックマネージャーパラメーターの説明は、/etc/modprobe.d/lockd.conf を参照してください。

NFS サーバーで rpcinfo -p コマンドを実行して、使用されているポートと RPC プログラムを確認します。

4.3.7.6. Red Hat Identity Management による NFS のセキュリティー保護

Red Hat Enterprise Linux に含まれる Red Hat Identity Management を使用している環境では、Kerberos 対応の NFS セットアップを大幅に簡素化できます。

Red Hat Enterprise Linux 7 Linux Domain Identity, Authentication, and Policy Guide、特にSetting up a Kerberos-aware NFS Serverを参照して、Red Hat Identity Management を使用する際に Kerberos で NFS を保護する方法を確認してください。

4.3.8. HTTP サーバーのセキュリティー保護

4.3.8.1. Apache HTTP Server のセキュリティー保護

Apache HTTP Server は、Red Hat Enterprise Linux 7 で最も安定かつ安全なサービスの 1 つです。Apache HTTP Server を保護するための多数のオプションと手法を利用することができます。数が

多いため、ここでは詳細な説明はしません。以下のセクションでは、Apache HTTP Server を実行する際のグッドプラクティスについて簡単に説明します。

システム上で実行されているスクリプトは、実稼働させる 前に 必ず意図したとおりに動作することを確認してください。また、root ユーザーのみがスクリプトまたは CGI を含むディレクトリーへの書き込み権限を持っていることを確認してください。これを行うには、root ユーザーで以下のコマンドを入力します。

```
chown root <directory_name>
```

```
chmod 755 <directory_name>
```

システム管理者は、次の設定オプション(/etc/httpd/conf/httpd.confで設定)を使用する場合は注意が必要です。

FollowSymLinks

このディレクティブはデフォルトで有効になっていますので、Web サーバーのドキュメント root へのシンボリックリンクを作成する場合は注意が必要です。たとえば、/ へのシンボリックリンクを提供することは適切ではありません。

Indexes

このディレクティブはデフォルトで有効になっていますが、望ましくない場合もあります。訪問者がサーバー上のファイルを閲覧できないようにするには、このディレクティブを削除してください。

UserDir

UserDir ディレクティブは、システム上にユーザーアカウントが存在することを確認できるため、デフォルトでは無効になっています。サーバーでユーザーディレクトリーの閲覧を可能にするには、以下のディレクティブを使用します。

```
UserDir enabled
    UserDir disabled root
```

これらのディレクティブは、/root/ 以外のすべてのユーザーディレクトリーの閲覧を有効にします。無効化されたアカウントのリストにユーザーを追加するには、UserDir disabled 行にスペースで区切られたユーザーのリストを追加します。

ServerTokens

ServerTokens ディレクティブは、クライアントに送り返されるサーバー応答ヘッダーフィー

ルドを制御します。これには、以下のパラメーターを使用してカスタマイズできるさまざまな情報が含まれています。

- **ServerTokens Full (デフォルトオプション) — 利用可能なすべての情報 (OS タイプや使用モジュール) を提供します。以下に例を示します。**

  ```
  Apache/2.0.41 (Unix) PHP/4.2.2 MyMod/1.2
  ```

- **ServerTokens Prod または ServerTokens ProductOnly — 以下の情報を提供します。**

  ```
  Apache
  ```

- **ServerTokens Major — 以下の情報を提供します。**

  ```
  Apache/2
  ```

- **ServerTokens Minor — 以下の情報を提供します。**

  ```
  Apache/2.0
  ```

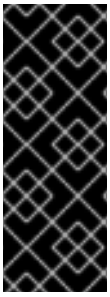
- **ServerTokens Min または ServerTokens Minimal — 以下の情報を提供します。**

  ```
  Apache/2.0.41
  ```

- **ServerTokens OS — 以下の情報を提供します。**

  ```
  Apache/2.0.41 (Unix)
  ```

攻撃者がシステムに関する重要な情報を得ることがないように、ServerTokens Prod オプションを使用することをお勧めします。



重要

IncludesNoExec ディレクティブを削除しないでください。デフォルトでは、*Server-Side Includes* (SSI) モジュールは、コマンドを実行できません。この設定は、攻撃者がシステム上でコマンドを実行できるようになる可能性があるため、絶対に必要な場合を除き、変更しないことを推奨します。

httpd モジュールの削除

特定のシナリオでは、HTTP サーバーの機能を制限するために特定の httpd モジュールを削除することが有益です。これを行うには、/etc/httpd/conf.modules.d ディレクトリーの設定ファイルを編集します。たとえば、プロキシーモジュールを削除するためには、以下のコマンドを実行します。

```
echo '# All proxy modules disabled' > /etc/httpd/conf.modules.d/00-proxy.conf
```

/etc/httpd/conf.d/ ディレクトリーには、モジュールの読み込みにも使用される設定ファイルが含まれていることに注意してください。

httpd および SELinux

詳細は、Red Hat Enterprise Linux 7 の SELinux User's Guide および Administrator's Guide の The Apache HTTP Server and SELinux の章を参照してください。

4.3.8.2. NGINX のセキュリティー保護

NGINX は、高性能の HTTP およびプロキシーサーバーです。本セクションでは、NGINX 設定を強化する追加の手順を簡単に説明します。NGINX 設定ファイルの server セクションで、以下の設定変更をすべて実行します。

バージョン文字列の無効化

攻撃者がサーバー上で動作している NGINX のバージョンを知ることを防ぐために、次の設定オプションを使用します。

```
server_tokens       off;
```

これには、バージョン番号を削除し、NGINX が提供するすべてのリクエストで文字列 nginx を単に報告するという効果があります。

```
$ curl -sI http://localhost | grep Server
Server: nginx
```

追加のセキュリティー関連ヘッダーを含む

NGINX が提供する各リクエストには、特定の既知の Web アプリケーションの脆弱性を緩和する追加の HTTP ヘッダーを含めることができます。

- add_header X-Frame-Options SAMEORIGIN; - このオプションは、ドメイン外のページが NGINX が提供するコンテンツをフレーム化するように拒否し、クリックジャッキング攻撃を効果的に軽減します。

- add_header X-Content-Type-Options nosniff; - このオプションは、特定の古いブラウザーで MIME タイプのスニッフィングを防ぎます。

- add_header X-XSS-Protection "1; mode=block"; - このオプションは、クロスサイトスクリプティング(XSS)フィルターリングを有効にします。これにより、ブラウザーは NGINX の応答に含まれる潜在的に悪意のあるコンテンツをレンダリングできなくなります。

潜在的に有害な HTTP メソッドを無効にする

一部の HTTP メソッドを有効にすると、開発者が Web アプリケーションをテストするために設計された Web サーバー上で、攻撃者がアクションを実行する可能性があります。たとえば、TRACE メソッドはクロスサイトトレース (XST) を許可することが知られています。

NGINX サーバーは、許可されるべきものだけをホワイトリストに登録することで、これらの有害な HTTP メソッドや任意のメソッドを拒否することができます。以下に例を示します。

```
# Allow GET, PUT, POST; return "405 Method Not Allowed" for all others.
if ( $request_method !~ ^(GET|PUT|POST)$ ) {
    return 405;
}
```

SSL の設定

NGINX Web サーバーが提供するデータを保護するには、HTTPS でのみサービスを提供することを検討してください。NGINX サーバーで SSL を有効にするための安全な設定プロファイルを生成するには、Mozilla SSL Configuration Generator を参照してください。生成された設定により、既知の脆弱なプロトコル (SSLv2 や SSLv3 など)、暗号、ハッシュアルゴリズム (3DES や MD5 など) が確実に無効化されます。

また、SSL サーバーテスト を使用して、設定した内容が最新のセキュリティー要件を満たしていることを確認できます。

4.3.9. FTP のセキュア化

ファイル転送プロトコル (FTP) は、ネットワーク上でファイルを転送するために設計された古い TCP プロトコルです。ユーザー認証を含むサーバーとのすべてのトランザクションが暗号化されていないため、安全でないプロトコルとみなされ、慎重に設定される必要があります。

Red Hat Enterprise Linux 7 は 2 つの FTP サーバーを提供します。

- Red Hat Content Accelerator (tux): FTP 機能を持つカーネルスペースの Web サーバー。
- vsftpd: スタンドアロンの、セキュリティー指向の FTP サービスの実装。

vsftpd FTP サービスを設定するためのセキュリティーガイドラインを以下に示します。

4.3.9.1. FTP グリーティングバナー

ユーザー名とパスワードを送信する前に、すべてのユーザーにグリーティングバナーが表示されます。デフォルトでは、このバナーには、システムの弱点を特定しようとするクラッカーに有用なバージョン情報が含まれています。

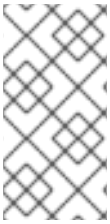
vsftpd のグリーティングバナーを変更するには、以下のディレクティブを /etc/vsftpd/vsftpd.conf ファイルに追加します。

```
ftpd_banner=<insert_greeting_here>
```

上記のディレクティブの *<insert_greeting_here>* をグリーティングメッセージのテキストで置き換えます。

複数行バナーの場合は、バナーファイルを使用することが推奨されます。複数のバナーの管理を簡素化するには、すべてのバナーを /etc/banners/ という新しいディレクトリーに配置します。この例の FTP 接続のバナーファイルは /etc/banners/ftp.msg です。以下は、このようなファイルの例です。

```
########## Hello, all activity on ftp.example.com is logged. ##########
```



注記

「TCP Wrapper と xinetd を使用したサービスの保護」 で指定されているように、ファイルの各行を 220 で始める必要はありません。

vsftpd のこのグリーティングバナーファイルを参照するには、以下のディレクティブを /etc/vsftpd/vsftpd.conf ファイルに追加します。

```
banner_file=/etc/banners/ftp.msg
```

また、「TCP Wrapper と接続バナー」 で説明されているように、TCP Wrappers を使用して着信接続に追加のバナーを送信することも可能です。

4.3.9.2. 匿名アクセス

/var/ftp/ ディレクトリーが存在すると、匿名アカウントが有効になります。

このディレクトリーを作成する最も簡単な方法は、vsftpd パッケージをインストールすることです。本パッケージは、匿名ユーザーのためのディレクトリーツリーを構築し、匿名ユーザーのためにディレクトリーのパーミッションを読み取り専用に設定します。

デフォルトでは、匿名ユーザーはどのディレクトリーにも書き込むことができません。

警告

FTP サーバーへの匿名アクセスを可能にする場合、機密データが保存される場所に注意してください。

4.3.9.2.1. 匿名のアップロード

匿名ユーザーがファイルをアップロードできるようにするには、/var/ftp/pub/ 内に書き込み専用ディレクトリーを作成することが推奨されます。root で以下のコマンドを実行します。

```
~]# mkdir /var/ftp/pub/upload
```

次に、匿名ユーザーがディレクトリーの内容を閲覧できないように、パーミッションを変更します。

```
~]# chmod 730 /var/ftp/pub/upload
```

- ディレクトリーの長い形式のリストは、次のようになります。

```
~]# ls -ld /var/ftp/pub/upload
drwx-wx---. 2 root ftp 4096 Nov 14 22:57 /var/ftp/pub/upload
```

匿名ユーザーにディレクトリーの読み取り/書き込みを許可すると、サーバーが盗まれたソフトウェアのリポジトリーになる可能性があります。

また、vsftpd で、以下の行を /etc/vsftpd/vsftpd.conf ファイルに追加します。

```
anon_upload_enable=YES
```

4.3.9.3. ユーザーアカウント

FTP は、認証用に非セキュアなネットワークで暗号化されていないユーザー名とパスワードを送信するため、ユーザーアカウントからサーバーへのシステムユーザーアクセスを拒否することが推奨されます。

vsftpd ですべてのユーザーアカウントを無効にするには、以下のディレクティブを /etc/vsftpd/vsftpd.conf に追加します。

```
local_enable=NO
```

4.3.9.3.1. ユーザーアカウントの制限

特定のアカウントまたは特定のアカウントグループ(root ユーザーや、sudo 権限を持つユーザーなど)の FTP アクセスを無効にするには、「Root アクセスの拒否」 で説明されているように PAM リストファイルを使用するのが最も簡単な方法です。vsftpd の PAM 設定ファイルは /etc/pam.d/vsftpd です。

また、各サービスでユーザーアカウントを直接無効にすることもできます。

vsftpd で特定のユーザーアカウントを無効にするには、ユーザー名を /etc/vsftpd/ftpusersに追加します。

4.3.9.4. TCP Wrapper を使用してアクセスを制御する

TCP Wrappers を使用して、「TCP Wrapper と xinetd を使用したサービスの保護」 で概説されているように、いずれかの FTP デーモンへのアクセスを制御します。

4.3.10. Postfix のセキュア化

Postfix は、Simple Mail Transfer Protocol (SMTP) を使用して他の MTA 間で電子メッセージを配信したり、クライアントや配信エージェントに電子メールを送信したりするメール転送エージェント (MTA) です。多くの MTA は相互にトラフィックを暗号化することが可能ですが、ほとんどの場合は暗号化しません。そのため、パブリックネットワークを介して電子メールを送信することは、本質的に安全でない通信形式と見なされます。Red Hat Enterprise Linux 7 では、Sendmail に代わり、Postfix がデフォルト MTA となっています。

Postfix サーバーの実装を計画している人は、以下の問題に対処することが推奨されます。

4.3.10.1. サービス拒否攻撃を制限する

電子メールの性質上、断固とした攻撃者は、サーバーを非常に簡単にメールで溢れさせ、サービス拒否を引き起こすことができます。このような攻撃の有効性は、/etc/postfix/main.cf ファイルのディレクティブの制限を設定することで制限できます。すでにあるディレクティブの値を変更したり、必要なディレクティブを以下のような形式で好きな値で追加したりすることができます。

```
<directive> = <value>
```

.サービス拒否攻撃を制限するために使用できるディレクティブの一覧を以下に示します。

- smtpd_client_connection_rate_limit - 時間単位ごとにクライアントがこのサービスに対して実行できる接続試行の最大数（以下で説明）。デフォルト値は 0 です。これは、クライアントが時間単位で Postfix が受け入れることができる数と同じ数の接続を行うことができることを意味します。デフォルトでは、信頼できるネットワーク内のクライアントは除外されます。

- anvil_rate_time_unit: この時間単位は、レート制限の計算に使用されます。デフォルト値は、60 秒です。

- smtpd_client_event_limit_exceptions - 接続およびレート制限コマンドから除外されるクライアント。デフォルトでは、信頼できるネットワーク内のクライアントは除外されます。

- smtpd_client_message_rate_limit - クライアントが時間単位ごとに要求できるメッセージ配信の最大数(Postfix が実際にこれらのメッセージを受け入れるかどうかに関係なく)。

- default_process_limit: 指定されたサービスを提供する Postfix 子プロセスのデフォルトの最大数。この制限は、master.cf ファイルの特定サービスに対してオーバーライドできます。デフォルト値は 100 です。

- queue_minfree - メールを受信するために必要なキューファイルシステムの空き領域の最小空き容量（バイト単位）。これは現在、Postfix SMTP サーバーがメールをまったく受け取らないかどうかを決定するために使用されています。デフォルトでは、Postfix SMTP サーバーは、空き容量が message_size_limit の 1.5 倍未満の場合、MAIL FROM コマンドを拒否します。空き容量の最小値をこれよりも高く指定するには、message_size_limit の 1.5 倍以上の queue_minfree 値を指定します。デフォルトの queue_minfree 値は 0 です。

- header_size_limit: メッセージヘッダーを保存するためのメモリーの最大量（バイト単位）。ヘッダーの方が大きい場合、超過分は破棄されます。デフォルト値は 102400 です。

- message_size_limit: エンベロープ情報を含むメッセージの最大サイズ（バイト単位）。デフォルト値は 10240000 です。

4.3.10.2. NFS と Postfix

メールスプールディレクトリー /var/spool/postfix/ を NFS 共有ボリュームに配置しないでください。NFSv2 および NFSv3 では、ユーザー ID およびグループ ID の管理を行わないため、2 人以上のユーザーが同じ UID を持ち、互いのメールを受信および読み取ることができます。

注記

Kerberos を使用する NFSv4 では、SECRPC_GSS カーネルモジュールは UID ベースの認証を使用しないため、これは当てはまりません。しかし、メールスプールディレクトリーを NFS 共有ボリュームに 置かない ことは、引き続きグッドプラクティスとされています。

4.3.10.3. メール専用ユーザー

Postfix サーバーのローカルユーザーによる不正使用を防止するために、メールユーザーは電子メールプログラムを使用してのみ Postfix サーバーにアクセスすることが最善とされます。メールサーバーのシェルアカウントは許可しないでください。/etc/passwd ファイルのすべてのユーザーシェルは /sbin/nologin に設定する必要があります(root ユーザーを除く可能性があります)。

4.3.10.4. Postfix ネットワークリスニングの無効化

デフォルトでは、Postfix はローカルのループバックアドレスのみをリッスンするように設定されています。これを確認するには、/etc/postfix/main.cf ファイルを表示します。

/etc/postfix/main.cf ファイルを表示して、以下の *inet_interfaces* 行のみが表示されることを確認します。

```
inet_interfaces = localhost
```

これにより、Postfix はネットワークからではなく、ローカルシステムからのメールメッセージ (cron ジョブのレポートなど) のみを受け入れるようになります。これはデフォルトの設定で、Postfix をネットワーク攻撃から保護します。

localhost の制限を削除し、Postfix がすべてのインターフェイスをリッスンできるようにするには、*inet_interfaces = all* 設定を使用できます。

4.3.10.5. Postfix が SASL を使用する設定

Red Hat Enterprise Linux 7 バージョンの Postfix は、*SMTP 認証（または SMTPAUTH ）*に Dovecot または Cyrus SASL 実装を使用できます。SMTP 認証は Simple Mail Transfer Protocol の拡張機能です。有効にすると、SMTP クライアントは、サーバーとクライアントの両方でサポートおよび許可される認証方法を使用して SMTP サーバーに対して認証する必要があります。本セクションでは、Dovecot SASL 実装を利用するように Postfix を設定する方法について説明します。

Dovecot POP/IMAP サーバーをインストールし、システムで Dovecot SASL 実装を利用できるようにするには、root ユーザーとして次のコマンドを発行します。

```
~]# yum install dovecot
```

Postfix SMTP サーバーは、*UNIX ドメインソケットまたは TCP ソケット* のいずれかを使用して、Dovecot SASL 実装と通信できます。後者の方法は、Postfix と Dovecot のアプリケーションが別々のマシンで実行されている場合にのみ必要です。このガイドでは、プライバシーを強化する UNIX ドメインソケット方式を優先しています。

Postfix に Dovecot SASL 実装を使用するように指示するには、両方のアプリケーションに対して多くの設定変更を実行する必要があります。以下の手順に従ってください。

Dovecot のセットアップ

1. メインの Dovecot 設定ファイル /etc/dovecot/conf.d/10-master.conf を変更して、次の行を含めます（デフォルトの設定ファイルにほとんどの関連セクションがすでに含まれているため、その行はコメント解除する必要があります）。

```
service auth {
  unix_listener /var/spool/postfix/private/auth {
    mode = 0660
    user = postfix
    group = postfix
  }
}
```

上記の例では、Postfix と Dovecot の間の通信に UNIX ドメインソケットを使用することを前提としています。また、/var/spool/postfix/ ディレクトリーにあるメールキューを含む Postfix SMTP サーバーのデフォルト設定と、postfix ユーザーおよびグループで実行されているアプリケーションを想定しています。このようにして、読み取りおよび書き込みのパーミッションは postfix ユーザーおよびグループに制限されます。

または、以下の設定を使用して、TCP を介して Postfix 認証要求をリッスンするように Dovecot を設定できます。

```
service auth {
  inet_listener {
    port = 12345
  }
}
```

上記の例では、12345 を使用するポートの数に置き換えます。

2. /etc/dovecot/conf.d/10-auth.conf 設定ファイルを編集して、Postfix SMTP サーバーに plain および login 認証メカニズムを提供するように Dovecot に指示します。

```
auth_mechanisms = plain login
```

Postfix のセットアップ

Postfix の場合、メインの設定ファイル /etc/postfix/main.cf のみを変更する必要があります。以下の設定ディレクティブを設定できます。

1. Postfix SMTP サーバーで SMTP 認証を有効にします。

```
smtpd_sasl_auth_enable = yes
```

2. **SMTP 認証に Dovecot SASL 実装を使用するように Postfix に指示します。**

```
smtpd_sasl_type = dovecot
```

3. **Postfix キューディレクトリーに相対的な認証パスを指定します（相対パスを使用すると、Postfix サーバーが chroot で実行されているかどうかに関係なく設定が機能することが保証されます）。**

```
smtpd_sasl_path = private/auth
```

この手順では、Postfix と Dovecot の間の通信に UNIX ドメインソケットを使用することを前提としています。通信に TCP ソケットを使用する場合に、別のマシンで Dovecot を検索するように Postfix を設定するには、以下のような設定値を使用します。

```
smtpd_sasl_path = inet:127.0.0.1:12345
```

上記の例では、127.0.0.1 を Dovecot マシンの IP アドレスに置き換え、12345 を Dovecot の/etc/dovecot/conf.d/10-master.conf 設定ファイルで指定されたポートに置き換える必要があります。

4. **Postfix SMTP サーバーがクライアントに提供する SASL メカニズムを指定します。暗号化されたセッションと暗号化されていないセッションで、異なるメカニズムを指定できることに注意してください。**

```
smtpd_sasl_security_options = noanonymous, noplaintext
smtpd_sasl_tls_security_options = noanonymous
```

上記の例では、暗号化されていないセッションの間、匿名認証は許可されず、暗号化されていないユーザー名やパスワードを送信するメカニズムも許可されないことを指定しています。暗号化されたセッション(TLSを使用)の場合、非匿名認証メカニズムのみが許可されます。

許可される SASL メカニズムを制限するためにサポートされるすべてのポリシーの一覧は、http://www.postfix.org/SASL_README.html#smtpd_sasl_security_options を参照してください。

関連情報

以下のオンラインリソースは、SASL による Postfix SMTP 認証の設定に役立つ追加情報を提供します。

- http://wiki2.dovecot.org/HowTo/PostfixAndDovecotSASL - SMTP 認証に Dovecot SASL 実装を使用するように Postfix を設定する方法に関する情報が含まれています。

- http://www.postfix.org/SASL_README.html#server_sasl - SMTP 認証用の Dovecot または Cyrus SASL 実装を使用するように Postfix を設定する方法に関する情報が含まれています。

4.3.11. SSH のセキュア化

Secure Shell(SSH) は、他のシステムと安全なチャンネルで通信するために使用される強力なネットワークプロトコルです。SSH を介した送信は暗号化され、傍受から保護されます。SSH プロトコルおよび Red Hat Enterprise Linux 7 での SSH サービスの使用に関する一般的な情報は、Red Hat Enterprise Linux 7 システム管理者のガイドの OpenSSH の章を参照してください。

重要

このセクションでは、SSH 設定を保護するための最も一般的な方法について説明します。この提案された対策のリストは、完璧または決定的なものと見なされるべきではありません。sshd デーモンの動作を変更するために利用できるすべての設定ディレクティブの説明については、sshd _config (5)を参照してください。また、SSH の基本的な概念については ssh (1) を参照してください。

4.3.11.1. 暗号化ログイン

SSH は、コンピューターにログインするための暗号鍵の使用をサポートしています。これは、パスワードのみを使用するよりもはるかに安全です。この方法を他の認証方法と組み合わせると、多要素認証と見なすことができます。複数の認証方法を使用する場合の詳細は、「複数の認証方法」 を参照してください。

認証に暗号化キーを使用できるようにするには、/etc/ssh/sshd_config ファイルの PubkeyAuthentication 設定ディレクティブを yes に設定する必要があります。これがデフォルト設定であることに注意してください。PasswordAuthentication ディレクティブを no に設定して、ログインにパスワードを使用する可能性を無効にします。

SSH キーは、ssh-keygen コマンドを使用して生成できます。追加の引数なしで呼び出されると、2048 ビットの RSA キーセットが作成されます。鍵はデフォルトで ~/.ssh/ ディレクトリーに保存されます。-b スイッチを利用すると、キーのビット強度を変更できます。通常、2048 ビットのキーを使用

するだけで十分です。Red Hat Enterprise Linux 7 システム管理者ガイドの Configuring OpenSSH の章には、キーペアの生成に関する詳細な情報が記載されています。

~/.ssh/ ディレクトリーに 2 つのキーが表示されるはずです。ssh-keygen コマンドの実行時にデフォルトを受け入れた場合、生成されたファイルの名前は id_rsa と id_rsa.pub で、それぞれ秘密鍵と公開鍵が含まれます。秘密鍵は、ファイルの所有者以外には読めないようにして、常に漏洩から保護する必要があります。ただし、公開鍵は、ログインするシステムに転送する必要があります。ssh-copy-id コマンドを使用して、鍵をサーバーに転送できます。

```
~]$ ssh-copy-id -i [user@]server
```

また、このコマンドは公開鍵を *サーバー* の ~/.ssh/authorized_keys ファイルに自動的に追加します。sshd デーモンは、サーバーにログインしようとするとこのファイルをチェックします。

パスワードやその他の認証メカニズムと同様に、SSH キーを定期的に変更する必要があります。その場合は、authorized_keys ファイルから未使用のキーが削除されていることを確認してください。

4.3.11.2. 複数の認証方法

複数の認証方法または多要素認証を使用すると、不正アクセスに対する保護レベルが向上するため、システムを強化してシステムが危険にさらされるのを防ぐ場合は、考慮する必要があります。多要素認証を使用するシステムにログインしようとするユーザーは、アクセスを許可されるために、指定されたすべての認証方法を正常に完了する必要があります。

/etc/ssh/sshd_config ファイルの AuthenticationMethods 設定ディレクティブを使用して、使用する認証方法を指定します。このディレクティブを使用して、必要な認証方法のリストを複数定義できることに注意してください。その場合、ユーザーは少なくとも 1 つのリストのすべてのメソッドを完了する必要があります。リストは空白で区切る必要があり、リスト内の個々の認証方法名はコンマで区切る必要があります以下に例を示します。

```
AuthenticationMethods publickey,gssapi-with-mic publickey,keyboard-interactive
```

上記の AuthenticationMethods ディレクティブを使用して設定された sshd デーモンは、ログインしようとしているユーザーが publickey 認証の後に gssapi-with-mic 認証または keyboard-interactive 認証のいずれかが正常に完了した場合にのみアクセスを許可します。要求された各認証方法は、/etc/ssh/sshd_config ファイルの対応する設定ディレクティブ(PubkeyAuthenticationなど)を使用して明示的に有効にする必要があります。利用可能な認証方法の一般的な一覧は、ssh (1) の『AUTHENTICATION』 セクションを参照してください。

4.3.11.3. SSH の他のセキュア化

プロトコルのバージョン

Red Hat Enterprise Linux 7 で提供される SSH プロトコルの実装は、SSH クライアント用の SSH-1 バージョンと SSH-2 バージョンの両方をサポートしますが、可能な限り後者のみを使用する必要があります。SSH-2 バージョンには、古い SSH-1 に比べて多くの改良が含まれており、高度な設定オプションの大部分は、SSH-2 を使用している場合にのみ使用できます。

Red Hat は、SSH プロトコルが使用される認証と通信を保護する範囲を最大化するために、SSH - 2 を使用することを推奨します。sshd デーモンがサポートするプロトコルのバージョンは、/etc/ssh/sshd_config ファイルの Protocol 設定ディレクティブを使用して指定できます。デフォルト設定は 2 です。SSH-2 バージョンは、Red Hat Enterprise Linux 7 SSH サーバーでサポートされている唯一のバージョンであることに注意してください。

鍵のタイプ

ssh-keygen コマンドは、デフォルトで SSH-2 RSA 鍵のペアを生成しますが、-t オプションを使用して、DSA 鍵または ECDSA 鍵を生成するように指示することもできます。ECDSA (Elliptic Curve Digital Signature Algorithm) は、同等の対称鍵長で RSA よりも優れたパフォーマンスを提供します。また、短いキーも生成します。

デフォルト以外のポート

デフォルトでは、sshd デーモンは TCP ポート 22 をリッスンします。ポートを変更すると、自動ネットワークスキャンに基づく攻撃にシステムがさらされる可能性が減るため、あいまいさによりセキュリティーが向上します。ポートは、/etc/ssh/sshd_config 設定ファイルの Port ディレクティブを使用して指定できます。また、デフォルト以外のポートを使用できるようにするためには、デフォルトの SELinux ポリシーを変更する必要があることに注意してください。これを行うには、root で以下のコマンドを入力して、ssh_port_t SELinux タイプを変更します。

```
~]# semanage -a -t ssh_port_t -p tcp port_number
```

上記のコマンドで、*port_number* を、Port ディレクティブを使用して指定された新しいポート番号に置き換えます。

root ログインなし

特定のユースケースで root ユーザーとしてログインする必要がない場合は、/etc/ssh/sshd_config ファイルで PermitRootLogin 設定ディレクティブを no に設定することを検討してください。root ユーザーとしてログインする可能性を無効にすることで、管理者は、通常のユーザーとしてログインし、root 権限を取得した後に、どのユーザーがどの特権コマンドを実行するかを監査できます。

X セキュリティー拡張機能の使用

Red Hat Enterprise Linux 7 クライアントの X サーバーは、X Security 拡張を提供しません。そのため、クライアントは X11 転送を使用して信頼できない SSH サーバーに接続するときに別のセキュリティー層を要求できません。ほとんどのアプリケーションは、この拡張機能を有効にして実行できませんでした。デフォルトでは、/etc/ssh/ssh_config ファイルの ForwardX11Trusted オプションが yes

に設定され、**ssh -X remote_machine**（信頼できないホスト）コマンドと **ssh -Y *remote_machine***（信頼できるホスト）コマンドには違いがありません。

警告

Red Hat では、信頼できないホストへの接続時は X11 転送を使用しないことを推奨しています。

4.3.12. PostgreSQL のセキュリティー確保

PostgreSQLは、オブジェクトリレーショナルデータベース管理システム (DBMS) です。**Red Hat Enterprise Linux 7** では、**postgresql-server** パッケージは **PostgreSQL** を提供します。インストールされていない場合は、**root** ユーザーになり、以下のコマンドを入力してインストールします。

```
~]# yum install postgresql-server
```

PostgreSQL の使用を開始する前に、ディスク上のデータベースストレージ領域を初期化する必要があります。これは、データベースのクラスターと呼ばれます。データベースクラスターを初期化するには、**PostgreSQL** と共にインストールされる **initdb** コマンドを使用します。データベースクラスターの希望するファイルシステムの場所は、**-D** オプションで示されます。以下に例を示します。

```
~]$ initdb -D /home/postgresql/db1
```

initdb コマンドは、指定したディレクトリーがまだ存在しない場合は作成しようとします。今回の例では、***/home/postgresql/db1*** という名前を使用します。***/home/postgresql/db1*** ディレクトリーには、データベースに保存されているすべてのデータと、クライアント認証設定ファイルが含まれています。

```
~]$ cat pg_hba.conf
# PostgreSQL Client Authentication Configuration File
# This file controls: which hosts are allowed to connect, how clients
# are authenticated, which PostgreSQL user names they can use, which
# databases they can access.  Records take one of these forms:
#
# local      DATABASE  USER  METHOD  [OPTIONS]
# host       DATABASE  USER  ADDRESS  METHOD  [OPTIONS]
# hostssl    DATABASE  USER  ADDRESS  METHOD  [OPTIONS]
# hostnossl  DATABASE  USER  ADDRESS  METHOD  [OPTIONS]
```

pg_hba.conf ファイルの以下の行により、認証されたローカルユーザーはユーザー名を使用してデータベースにアクセスできます。

```
local   all             all                                     trust
```

これは、データベースユーザーを作成し、ローカルユーザーを作成しないレイヤー型アプリケーションを使用する場合に、問題となることがあります。システム上のすべてのユーザー名を明示的に制御しない場合は、pg_hba.conf ファイルからこの行を削除します。

4.3.13. Docker のセキュリティー確保

*Docker*は、Linux コンテナー内でのアプリケーションのデプロイメントを自動化するオープンソースプロジェクトで、アプリケーションとそのランタイム依存関係をコンテナーにパッケージ化する機能を提供しています。Docker ワークフローをより安全にするには、Red Hat Enterprise Linux Atomic Host 7 Container Security Guide の手順に従います。

4.3.14. DDoS 攻撃からの memcached の保護

Memcached は、オープンソースの高性能分散メモリーオブジェクトキャッシングシステムです。これは、本来は汎用的なものですが、データベースの負荷を軽減することで、動的 Web アプリケーションのパフォーマンスを向上させるために主に使用されます。

Memcached は、データベース呼び出し、API 呼び出し、またはページレンダリングの結果から、文字列やオブジェクトなどの任意のデータの小さなチャンクを格納するメモリー内のキーと値のストアです。Memcached を使用すると、アプリケーションは、システムが必要以上にメモリーを搭載している部分からメモリーを取り出し、アプリケーションが必要未満のメモリーしか搭載していない領域へアクセスできるようにします。

memcached の脆弱性

2018 年に、パブリックインターネットに公開されている memcached サーバーを悪用することによる DDoS 増幅攻撃の脆弱性が発見されました。これらの攻撃は、トランスポートに UDP プロトコルを使用する memcached 通信を利用します。増幅率が高いため、攻撃は効果的です。数百バイトのサイズの要求は、数メガバイトまたは数百メガバイトのサイズの応答を生成することができます。この問題には CVE-2018-1000115 が割り当てられています。

ほとんどの場合、memcached サービスはパブリックインターネットに公開する必要はありません。このような露出には、リモートの攻撃者が memcached に保存されている情報を漏洩または変更できるなど、独自のセキュリティー問題があります。

memcached の強化

セキュリティーリスクを軽減するために、以下の手順のうち、お使いの設定に該当するものをでき

るだけ多く実行してください。

- LAN にファイアウォールを設定してください。ローカルネットワーク内からのみ、memcached サーバーにアクセスできるようにする必要がある場合は、memcached が使用するポートへの外部トラフィックを許可しないでください。たとえば、許可されているポートのリストから、デフォルトで memcached によって使用されるポート 11211 を削除します。

```
~]# firewall-cmd --remove-port=11211/udp
~]# firewall-cmd --runtime-to-permanent
```

  特定の IP 範囲にポート 11211 の使用を許可する firewalld コマンドについては、「ゾーンを使用し、ソースに応じた着信トラフィックの管理」 を参照してください。

- クライアントが本当にこのプロトコルを必要としない限り、/etc/sysconfig/memcached ファイルの OPTIONS 変数に -U 0 -p 11211 値を追加して UDP を無効にします。

```
OPTIONS="-U 0 -p 11211"
```

- アプリケーションと同じマシンで単一の memcached サーバーを使用する場合、ローカルホストトラフィックのみをリッスンするように memcached を設定します。-l 127.0.0.1,::1 の値を /etc/sysconfig/memcached の OPTIONS に追加します。

```
OPTIONS="-l 127.0.0.1,::1"
```

- 認証の変更が可能な場合は、SASL (Simple Authentication and Security Layer) 認証を有効にしてください。

  1. /etc/sasl2/memcached.conf ファイルで を変更または追加します。

```
sasldb_path: /path.to/memcached.sasldb
```

  2. SASL データベースにアカウントを追加します。

```
~]# saslpasswd2 -a memcached -c cacheuser -f /path.to/memcached.sasldb
```

3. **memcached のユーザーとグループがデータベースにアクセスできることを確認します。**

```
~]# chown memcached:memcached /path.to/memcached.sasldb
```

4. **/etc/sysconfig/memcached に -S 値を OPTIONS に追加して、memcached で SASL サポートを有効にします。**

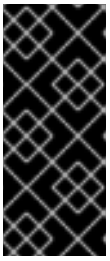
```
OPTIONS="-S"
```

5. **memcached サーバーを再起動して、変更を適用します。**

6. **SASL データベースで作成したユーザー名とパスワードを、お使いのアプリケーションの memcached クライアント設定に追加します。**

- **memcached クライアントとサーバー間の通信を stunnel で暗号化します。memcached は TLS をサポートしていないため、回避策は、memcached プロトコルの上に TLS を提供する stunnel などのプロキシーを使用することです。**

  PSK (Pre Shared Keys)を使用するか、ユーザー証明書を使用するように stunnel を設定できます。証明書を使用する場合、認証されたユーザーのみがお使いの memcached サーバーにアクセスでき、トラフィックは暗号化されます。



重要

トンネルを使用して memcached にアクセスする場合は、サービスがローカルホストでのみリッスンしているか、ファイアウォールがネットワークから memcached ポートへのアクセスを阻止しているかを確認してください。

詳細は、「stunnel の使用」 を参照してください。

4.4. ネットワークアクセスのセキュア化

4.4.1. TCP Wrapper と xinetd を使用したサービスの保護

TCP Wrapper の機能は、サービスへのアクセスを拒否するだけではありません。このセクションでは、これらを使用して接続バナーを送信し、特定のホストからの攻撃を警告し、ロギング機能を強化する方法を説明します。TCP Wrapper の機能および制御言語に関する詳細は、hosts_options(5) の man ページを参照してください。利用可能なフラグ (サービスに適用できるオプションとして機能) については、xinetd.conf(5) の man ページを参照してください。

4.4.1.1. TCP Wrapper と接続バナー

ユーザーがサービスに接続する際に適切なバナーを表示することは、潜在的な攻撃者に対して、システム管理者が警戒していることを知らせる良い方法です。システムに関するどの情報をユーザーに表示するかを制御することもできます。サービスに TCP Wrapper バナーを実装するには、banner オプションを使用します。

この例では、vsftpd にバナーを実装します。最初にバナーファイルを作成します。これは、システム上のどこにあってもかまいませんが、デーモンと同じ名前である必要があります。この例では、ファイルは /etc/banners/vsftpd と呼ばれ、以下の行が含まれます。

```
220-Hello, %c
220-All activity on ftp.example.com is logged.
220-Inappropriate use will result in your access privileges being removed.
```

%c トークンは、ユーザー名やホスト名、ユーザー名および IP アドレスなどのさまざまなクライアント情報を提供し、接続がより困難になります。

このバナーを受信接続に表示するには、以下の行を /etc/hosts.allow ファイルに追加します。

```
vsftpd : ALL : banners /etc/banners/
```

4.4.1.2. TCP Wrapper と攻撃警告

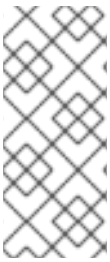
特定のホストまたはネットワークがサーバーの攻撃を検出した場合、TCP Wrapper を使用して、spawn ディレクティブを使用して、そのホストまたはネットワークからの後続の攻撃について管理者に警告できます。

この例では、206.182.68.0/24 ネットワークからサーバーを攻撃しようとするクラッカーが検出されたと仮定します。/etc/hosts.deny ファイルに以下の行を追加して、そのネットワークからの接続試行を拒否し、その試行を特別なファイルに記録します。

```
ALL : 206.182.68.0 : spawn /bin/echo `date` %c %d >> /var/log/intruder_alert
```

%d トークンは、攻撃者がアクセスしようとしたサービスの名前を提供します。

接続を許可し、ログに記録するには、spawn ディレクティブを /etc/hosts.allow ファイルに配置します。



注記

spawn ディレクティブはシェルコマンドを実行するため、特定のクライアントがサーバーへの接続を試行した場合に、管理者に通知するか、一連のコマンドを実行するための特別なスクリプトを作成することが推奨されます。

4.4.1.3. TCP Wrapper とロギングの強化

特定のタイプの接続が他の接続よりも懸念される場合は、severity オプションを使用してそのサービスのログレベルを上げることができます。

この例では、FTP サーバーのポート 23 (Telnet ポート) に接続しようとしている人はクラッカーであると想定します。これを示すには、ログファイルにデフォルトのフラグ info の代わりに emerg フラグを配置し、接続を拒否します。

これを行うには、以下の行を /etc/hosts.deny に配置します。

```
in.telnetd : ALL : severity emerg
```

これはデフォルトの authpriv ロギング機能を使用しますが、優先度をデフォルト値の info から emerg に上げます。これにより、ログメッセージがコンソールに直接投稿されます。

4.4.2. リッスンしているポートの確認

使用されていないポートは、攻撃の可能性を避けるために閉じることが重要です。リスニング状態にある予期せぬポートについては、侵入の可能性がないか調査する必要があります。

開いているポートスキャンでの netstat の使用

root で以下のコマンドを入力して、ネットワークからの接続をリッスンしているポートを確認します。

```
~]# netstat -pan -A inet,inet6 | grep -v ESTABLISHED
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address       Foreign Address    State     PID/Program name
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN      1/systemd
tcp        0      0 192.168.124.1:53        0.0.0.0:*               LISTEN      1829/dnsmasq
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      1176/sshd
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN      1177/cupsd
tcp6       0      0 :::111                  :::*                    LISTEN      1/systemd
tcp6       0      0 ::1:25                  :::*                    LISTEN      1664/master
sctp               0.0.0.0:2500                                      LISTEN   20985/sctp_darn
udp        0      0 192.168.124.1:53        0.0.0.0:*                           1829/dnsmasq
udp        0      0 0.0.0.0:67              0.0.0.0:*                           977/dhclient
...
```

netstat コマンドの -l オプションを使用して、リッスンしているサーバーソケットのみを表示します。

```
~]# netstat -tlnw
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN
tcp        0      0 192.168.124.1:53        0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
tcp6       0      0 :::111                  :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 ::1:631                 :::*                    LISTEN
tcp6       0      0 ::1:25                  :::*                    LISTEN
raw6       0      0 :::58                   :::*                    7
```

開いているポートスキャンでの ss の使用

または、ss ユーティリティーを使用して、リスニング状態で開いているポートを一覧表示します。netstat よりも多くの TCP および状態情報を表示できます。

```
~]# ss -tlw
etid State      Recv-Q Send-Q     Local Address:Port                 Peer Address:Port
udp   UNCONN     0      0                 :::ipv6-icmp                       :::*
tcp   LISTEN     0      128               *:sunrpc                          *:*
tcp   LISTEN     0      5       192.168.124.1:domain                            *:*
tcp   LISTEN     0      128               *:ssh                             *:*
tcp   LISTEN     0      128         127.0.0.1:ipp                            *:*
tcp   LISTEN     0      100         127.0.0.1:smtp                            *:*
tcp   LISTEN     0      128               :::sunrpc                         :::*
tcp   LISTEN     0      128               :::ssh                            :::*
tcp   LISTEN     0      128               ::1:ipp                           :::*
tcp   LISTEN     0      100               ::1:smtp                           :::*
```

```
~]# ss -plno -A tcp,udp,sctp
Netid State      Recv-Q Send-Q      Local Address:Port                  Peer Address:Port
udp   UNCONN     0      0         192.168.124.1:53                         *:*                  users:
(("dnsmasq",pid=1829,fd=5))
udp   UNCONN     0      0            *%virbr0:67                           *:*                  users:
(("dnsmasq",pid=1829,fd=3))
udp   UNCONN     0      0                 *:68                            *:*                  users:
(("dhclient",pid=977,fd=6))
...
tcp   LISTEN     0      5         192.168.124.1:53                         *:*                  users:
(("dnsmasq",pid=1829,fd=6))
tcp   LISTEN     0      128               *:22                            *:*                  users:
(("sshd",pid=1176,fd=3))
tcp   LISTEN     0      128         127.0.0.1:631                          *:*                  users:
(("cupsd",pid=1177,fd=12))
tcp   LISTEN     0      100         127.0.0.1:25                           *:*                  users:
(("master",pid=1664,fd=13))
...
sctp  LISTEN     0      5                 *:2500                          *:*                  users:
(("sctp_darn",pid=20985,fd=3))
```

UNCONN 状態は、UDP リスニングモードのポートを示します。

外部システムから、ss 出力に示されるすべての IP アドレス(localhost 127.0.0.0 または ::1 範囲を除く)に対してスキャンを行います。IPv6 アドレスをスキャンするには、-6 オプションを使用します。

次に、ネットワーク経由で最初のシステムに接続している別のリモートマシンから nmap ツールを使用して外部チェックを行います。これは、firewalld のルールを検証するために使用できます。次に、TCP 接続をリッスンしているポートを判別する例を示します。

```
~]# nmap -sT -O 192.168.122.65
   Starting Nmap 6.40 ( http://nmap.org ) at 2017-03-27 09:30 CEST
   Nmap scan report for 192.168.122.65
   Host is up (0.00032s latency).
   Not shown: 998 closed ports
   PORT    STATE SERVICE
```

```
22/tcp  open  ssh
111/tcp open  rpcbind
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.7 - 3.9
Network Distance: 0 hops

OS detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.79 seconds
```

TCP 接続スキャン (-sT) は、TCP SYN スキャン (-sS) がオプションでない場合のデフォルトの TCP スキャンタイプです。-O オプションは、ホストのオペレーティングシステムを検出します。

netstat と ss を使用して開いている SCTP ポートのスキャン

netstat ユーティリティーは、Linux ネットワークサブシステムに関する情報を出力します。開いている Stream Control Transmission Protocol (SCTP)ポートのプロトコル統計を表示するには、root で以下のコマンドを入力します。

```
~]# netstat -plnS
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address   Foreign Address  State    PID/Program name
sctp              127.0.0.1:250                 LISTEN   4125/sctp_darn
sctp       0      0 127.0.0.1:260   127.0.0.1:250    CLOSE    4250/sctp_darn
sctp       0      0 127.0.0.1:250   127.0.0.1:260    LISTEN   4125/sctp_darn
```

```
~]# netstat -nl -A inet,inet6 | grep 2500
sctp              0.0.0.0:2500                              LISTEN
```

ss ユーティリティーは、SCTP のオープンポートを表示することもできます。

```
~]# ss -an | grep 2500
sctp   LISTEN     0      5         *:2500                  *:*
```

詳細は、ss(8)、netstat(8)、nmap(1)、および services(5) の man ページを参照してください。

4.4.3. ソースルーティングの無効化

ソースルーティングは、IP パケットが情報 (アドレスのリスト) を伝送できるようにするインターネットプロトコルメカニズムであり、パケットがたどる必要のあるパスをルーターに通知します。ルートが通過するときにホップを記録するオプションもあります。取得されたホップのリストである "route record" は、宛先に送信元へのリターンパスを提供します。これにより、送信元 (送信ホスト) は、一部またはすべてのルーターのルーティングテーブルを無視して、大まかにまたは厳密にルートを指定でき

ます。これにより、ユーザーは悪意のある目的でネットワークトラフィックをリダイレクトできます。このため、ソースベースのルーティングは無効にする必要があります。

accept_source_route オプションを使用すると、ネットワークインターフェイスが *Strict Source Routing* (SSR) または *Loose Source Routing* (LSR) のオプションが設定されたパケットを受け入れます。ソースルーティングされたパケットの受け入れは、sysctl の設定によって制御されます。SSR または LSR オプションが設定されたパケットをドロップするために、root で次のコマンドを発行します。

```
~]# /sbin/sysctl -w net.ipv4.conf.all.accept_source_route=0
```

パケットの転送を無効にすることも、可能な限り上記と合わせて行ってください (転送を無効にすると、仮想化に支障をきたす場合があります)。以下のコマンドを root で発行します。

これらのコマンドは、すべてのインターフェイスで IPv4 および IPv6 パケットの転送を無効化します。

```
~]# /sbin/sysctl -w net.ipv4.conf.all.forwarding=0
```

```
~]# /sbin/sysctl -w net.ipv6.conf.all.forwarding=0
```

これらのコマンドは、すべてのインターフェイスですべてのマルチキャストパケットの転送を無効にします。

```
~]# /sbin/sysctl -w net.ipv4.conf.all.mc_forwarding=0
```

```
~]# /sbin/sysctl -w net.ipv6.conf.all.mc_forwarding=0
```

ICMP リダイレクトの受け入れには、正当な用途はほとんどありません。特に必要な場合を除き、ICMP リダイレクトパケットの受け入れと送信を無効にしてください。

これらのコマンドは、すべてのインターフェイスですべての ICMP リダイレクトパケットの受け入れを無効にします。

```
~]# /sbin/sysctl -w net.ipv4.conf.all.accept_redirects=0
```

```
~]# /sbin/sysctl -w net.ipv6.conf.all.accept_redirects=0
```

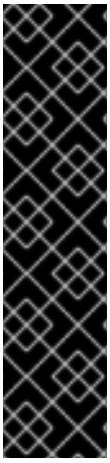
このコマンドは、すべてのインターフェイスでセキュアな ICMP リダイレクトパケットの受け入れ

を無効にします。

```
~]# /sbin/sysctl -w net.ipv4.conf.all.secure_redirects=0
```

このコマンドは、すべてのインターフェイスで IPv4 ICMP リダイレクトパケットの受け入れを無効にします。

```
~]# /sbin/sysctl -w net.ipv4.conf.all.send_redirects=0
```

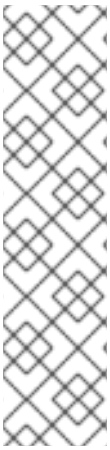


重要

少なくとも、net.ipv4.conf.all.send_redirects オプションまたは net.ipv4.conf.*interface*.send_redirects オプションのいずれか一方が有効になっていると、ICMP リダイレクトの送信はアクティブなままとなります。すべてのインターフェイスの net.ipv4.conf. *interface* .send_redirects オプションを 0 の値に設定するようにしてください。新しいインターフェイスを追加するたびに ICMP リクエストの送信を自動的に無効にするには、次のコマンドを入力します。

```
~]# /sbin/sysctl -w net.ipv4.conf.default.send_redirects=0
```

IPv4 リダイレクトパケットの送信を無効にするディレクティブは 1 つのみです。このような IPv4 と IPv6 の違いを生んだ「IPv6 Node Requirements」については、RFC4294を参照してください。



注記

これらの設定を再起動後も保持するには、/etc/sysctl.conf ファイルを変更します。たとえば、すべてのインターフェイスですべての IPv4 ICMP リダイレクトパケットの受け入れを無効にするには、root ユーザーとして実行しているエディターで /etc/sysctl.conf ファイルを開き、以下の行を追加します。

```
net.ipv4.conf.all.send_redirects=0
```

詳細は、sysctl の man ページの sysctl (8) を参照してください。ソースベースのルーティングとそのバリアントに関連するインターネットオプションの説明については、RFC791 を参照してください。

警告

イーサネットネットワークは、ARP または MAC アドレスのスプーフィング、承認されていない DHCP サーバー、IPv6 ルーターまたはネイバーアドバタイズメントなど、トラフィックをリダイレクトする方法をさらに提供します。また、ユニキャストトラフィックがブロードキャストされることもあり、情報漏えいの原因となります。これらの弱点は、ネットワークオペレーターが実装する具体的な対策によってのみ対処することができます。ホストベースの対策は十分に効果的ではありません。

4.4.3.1. 逆方向パス転送

逆方向パス転送は、あるインターフェイスを介して到達したパケットが、別のインターフェイスを介して出ることを防ぐために使用されます。発信経路と着信経路が異なる場合、*非対称ルーティング*と呼ばれることがあります。ルーターは多くの場合、この方法でパケットをルーティングしますが、ほとんどのホストはこれを行う必要はありません。ただし、あるリンクでトラフィックを送信し、別のリンクで別のサービスプロバイダーからトラフィックを受信するようなアプリケーションは例外となります。たとえば、専用回線と xDSL の組み合わせや、衛星回線と 3G モデムの併用などです。このようなシナリオに該当する場合は、受信インターフェイスで逆方向パス転送をオフにする必要があります。つまり、ユーザーがローカルサブネットから IP アドレスのスプーフィングを防ぎ、DDoS 攻撃の可能性を低減するため、必要であると分からない限り、有効にすることが最適です。

注記

Red Hat Enterprise Linux 7 はデフォルトで *Strict Reverse Path Forwarding* を使用します。これは、RFC 3704, Ingress Filtering for Multihomed Networks の厳密な逆方向パスに関する推奨事項に準拠します。

警告

転送が有効になっている場合、逆方向パス転送は、（たとえば iptables ルールなど）ソースアドレス検証の他の手段がある場合にのみ無効にする必要があります。

rp_filter

逆方向パス転送は、rp_filterディレクティブによって有効になります。sysctl ユーティリ

ティーを使用すると、実行中のシステムに変更を加えることができます。永続的な変更は、行を /etc/sysctl.conf ファイルに追加することで実行できます。rp_filter オプションは、カーネルに 3 つのモードのうち 1 つを選択するよう指示するために使用されます。

一時的なグローバル変更を行うには、root で以下のコマンドを入力します。

```
sysctl -w  net.ipv4.conf.default.rp_filter=integer
sysctl -w net.ipv4.conf.all.rp_filter=integer
```

ここで、*integer* は、以下のいずれかになります。

- 0 - ソースの検証がありません。
- 1 - RFC 3704 で定義された厳密なモード。
- 2 - RFC 3704 で定義された疎結合モード。

この設定は、以下のように net.ipv4.conf. interface .rp_filter コマンドを使用して、ネットワーク*インターフェイス* ごとに上書きできます。

```
sysctl -w net.ipv4.conf.interface.rp_filter=integer
```

注記

これらの設定を再起動後も保持するには、/etc/sysctl.conf ファイルを変更します。たとえば、すべてのインターフェイスのモードを変更するには、root ユーザーとして実行しているエディターで /etc/sysctl.conf ファイルを開き、以下の行を追加します。

```
net.ipv4.conf.all.rp_filter=2
```

IPv6_rpfilter

IPv6 プロトコルの場合、firewalld デーモンはデフォルトで逆方向パス転送に適用されます。この設定は、/etc/firewalld/firewalld.conf ファイルで確認できます。IPv6_rpfilter オプションを設

定することで、firewalld の動作を変更できます。

逆方向パス転送のカスタム設定が必要な場合は、以下のように ip6tables コマンドを使用して、firewalld デーモン なし でこれを実行できます。

```
ip6tables -t raw -I PREROUTING -m rpfilter --invert -j DROP
```

このルールは、特にステートフルマッチングルールの前にすべてのトラフィックに適用されるように、raw/PREROUTING チェーンの先頭の近くに挿入される必要があります。iptables サービスおよび ip6tables サービスの詳細は、「iptablesを使用した IP セットの設定および制御」 を参照してください。

パケット転送の有効化

システム外部から到着したパケットを別の外部ホストに転送できるようにするには、カーネルで IP フォワーディングを有効にする必要があります。root としてログインし、/etc/sysctl.conf ファイルの net.ipv4.ip_forward = 0 を読み取る行を以下のように変更します。

```
net.ipv4.ip_forward = 1
```

/etc/sysctl.conf ファイルから変更を読み込むには、以下のコマンドを入力します。

```
/sbin/sysctl -p
```

IP 転送が有効になっているかどうかを確認するには、root で以下のコマンドを実行します。

```
/sbin/sysctl net.ipv4.ip_forward
```

上記のコマンドが 1 を返すと、IP 転送が有効になります。0 を返す場合は、以下のコマンドを使用して手動でオンにできます。

```
/sbin/sysctl -w net.ipv4.ip_forward=1
```

4.4.3.2. 関連情報

逆方向パス転送の詳細については、以下の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

  /usr/share/doc/kernel-doc-*バージョン*/Documentation/networking/ip-sysctl.txt: このファイルには、ディレクトリーで利用可能なファイルとオプションの完全なリストが含まれています。カーネルのドキュメントに初めてアクセスする前に、**root** で以下のコマンドを入力します。

```
~]# yum install kernel-doc
```

- オンラインドキュメント

  マルチホームネットワークの **Ingress** フィルターリングの説明については、**RFC 3704** を参照してください。

4.5. DNSSEC を使用した DNS トラフィックのセキュア化

4.5.1. DNS の概要

DNSSEC は、**DNSSEC(** ***Domain Name System Security Extensions*** **)**のセットで、**DNS** クライアントが **DNS** ネームサーバーからの応答の整合性を認証およびチェックし、発信元を検証し、転送中に改ざんされているかどうかを判断できるようにします。

4.5.2. DNSSEC について

インターネット経由で接続するために、**Web** サイトの数が増え、**HTTPS** を使用して安全に接続する機能が提供されるようになりました。ただし、**HTTPS Web** サーバーに接続する前に、**IP** アドレスを直接入力しない限り、**DNS** ルックアップを実行する必要があります。これらの **DNS** ルックアップは安全*ではなく*、*認証がないため*、*中間者* 攻撃の対象となります。つまり、**DNS** クライアントは、特定の **DNS** ネームサーバーから送られるように見える応答が本物であり、改ざんされていないことを確信できません。さらに重要なことに、再帰的ネームサーバーは、他のネームサーバーから取得したレコードが本物であることを確認できません。**DNS** プロトコルは、クライアントが中間者攻撃を受けないようにするためのメカニズムを提供していませんでした。**DNSSEC** は、**DNS** を使用してドメイン名を解決する際に認証および整合性チェックの欠如に対処するために導入されました。**DNSSEC** は、機密性の問題には対処しません。

DNSSEC 情報の公開には、**DNS** リソースレコードのデジタル署名や、**DNS** リゾルバーが信頼の階層チェーンを構築できるようにするなどの方法で公開鍵を配布することが含まれます。すべての **DNS** リソースレコードのデジタル署名が生成され、デジタル署名リソースレコード(**RRSIG**)としてゾーンに追加されます。ゾーンの公開鍵は、**DNSKEY** リソースレコードとして追加されます。階層的な連鎖を構築するために、**DNSKEY** のハッシュを**DS(*Delegation of Signing*)** リソースレコードとして親ゾーンで公開します。消極的事実の証明を容易にするために、***NextSECure*** **(NSEC)** および **NSEC3** リソース

レコードが使用されます。DNSSEC 署名付きゾーンでは、各 *リソースレコードセット* (RRset) には対応する RRSIG リソースレコードがあります。子ゾーンへの委任に使用されるレコード (NS レコードおよび glue レコード) は署名されないことに注意してください。これらのレコードは子ゾーンに表示され、そこに署名されています。

DNSSEC 情報の処理は、root ゾーン公開鍵が設定されたリゾルバーによって実行されます。この鍵を使用して、リゾルバーは root ゾーンで使用される署名を検証できます。たとえば、ルートゾーンは .com の DS レコードに署名しています。ルートゾーンは、.com ネームサーバーの NS レコードおよび glue レコードも提供します。リゾルバーはこの委譲に従い、これらの委譲されたネームサーバーを使用して .com の DNSKEY レコードをクエリーします。取得した DNSKEY レコードのハッシュは、root ゾーンの DS レコードと一致する必要があります。その場合、リゾルバーは .com の取得した DNSKEY を信頼します。 .com ゾーンでは、RRSIG レコードは .com DNSKEY によって作成されます。このプロセスは、redhat.com など、.com 内の委譲についても同様に繰り返されます。この方法を使用すると、検証用 DNS リゾルバーは、通常の操作中に世界中から多くの DNSKEY を収集する間、1 つのルートキーでのみ設定する必要があります。暗号チェックに失敗した場合、リゾルバーはアプリケーションに SERVFAIL を返します。

DNSSEC は、DNSSEC をサポートしていないアプリケーションからは完全に見えないように設計されています。DNSSEC 非対応のアプリケーションが DNSSEC 対応リゾルバーにクエリーした場合、RRSIG などのこれらの新しいリソースレコードタイプがなくても応答を受け取ります。ただし、DNSSEC 対応のリゾルバーは引き続きすべての暗号化チェックを実行し、悪意のある DNS 応答を検出すると、アプリケーションに引き続き SERVFAIL エラーを返します。DNSSEC は、DNS サーバー（権威および再帰的）間のデータの整合性を保護します。これは、アプリケーションとリゾルバー間のセキュリティーを提供しません。したがって、アプリケーションにリゾルバーへの安全なトランスポートを提供することが重要です。これを実現する最も簡単な方法は、DNSSEC 対応のリゾルバーを localhost で実行し、/etc/resolv.conf で 127.0.0.1 を使用することです。または、リモート DNS サーバーへの VPN 接続を使用することもできます。

ホットスポットの問題について

Wi-Fi Hotspots または VPN は、Wi-Fi サービスの認証（または支払い）が必要なページにユーザーをリダイレクトするために、DNS を乗っ取る傾向があります。「」VPN に接続するユーザーは、企業ネットワーク外に存在しないリソースを見つけるために、「内部のみ」 の DNS サーバーを使用することがよくあります。これには、ソフトウェアによる追加の処理が必要です。たとえば、dnssec-trigger を使用して、ホットスポットが DNS クエリーを乗っ取っているかどうかを検出でき、unbound は DNSSEC クエリーを処理するプロキシーネームサーバーとして機能できます。

DNSSEC 対応の再帰的リゾルバーの選択

DNSSEC 対応の再帰リゾルバーをデプロイするには、BIND または unbound のいずれかを使用できます。いずれもデフォルトで DNSSEC を有効にし、DNSSEC root キーで設定されています。サーバーで DNSSEC を有効にするには、どちらのサーバーも機能しますが、ノートブックなどのモバイルデバイスでは、ローカルユーザーが動的に DNSSEC オーバーライドを再設定できるため、dnssec-trigger を使用する際にローカルユーザーが Hotspot に必要な DNSSEC オーバーライドを動的に再設定し、Libreswan を使用する場合は VPN 向けに を使用することが推奨されます。unbound デーモンは、サーバーとモバイルデバイスの両方で役立つ etc/unbound/*.d/ ディレクトリーにリストされている DNSSEC 例外のデプロイメントをさらにサポートします。

4.5.3. Dnssec-trigger について

unbound が /etc/resolv.conf にインストールされ、設定されると、アプリケーションからの DNS クエリーはすべて unbound によって処理されます。dnssec-trigger は、トリガーされた場合にのみ unbound リゾルバーを再設定します。これは、ノートパソコンなど、異なる Wi-Fi ネットワークに接続するローミングクライアントマシンに主に適用されます。プロセスは以下のようになります。

- NetworkManager は、新しい DNS サーバーが DHCP 経由で取得されたときに dnssec-trigger を 「トリガー」 します。
- dnssec-trigger は、サーバーに対して多くのテストを実行し、DNSSEC を適切にサポートしているかどうかを判断します。
- その場合、dnssec-trigger は unbound を再設定し、その DNS サーバーをすべてのクエリーのフォワーダーとして使用します。
- テストが失敗した場合、dnssec-trigger は新しい DNS サーバーを無視し、いくつかの利用可能なフォールバック方法を試します。
- 無制限のポート 53 (UDP および TCP)が使用可能であると判断した場合、フォワーダーを使用せずに完全な再帰 DNS サーバーになるように unbound に指示します。
- これが不可能な場合（たとえば、ポート 53 がネットワークの DNS サーバー自体に到達する場合を除きすべてについてファイアウォールによってブロックされている場合）は、DNS を使用してポート 80 に到達するか、または TLS でポート 443 にカプセル化された DNS を使用しようとします。ポート 80 および 443 で DNS を実行しているサーバーは、/etc/dnssec-trigger/dnssec-trigger.conf で設定できます。コメントアウトされた例は、デフォルトの設定ファイルで利用できるはずです。
- これらのフォールバック方法も失敗すると、dnssec-trigger は、DNSSEC を完全にバイパスする安全でない方法で動作するか、新しい DNS クエリーを試行しないが、キャッシュ内にすでに存在するすべてのものに応答する 「cache only」 モードで実行することを提案します。

Wi-Fi ホットスポットは、インターネットへのアクセスを許可する前に、ユーザーをサインオンページにリダイレクトすることが多くなっています。上記のプロービングシーケンス中にリダイレクトが検出された場合、インターネットにアクセスするためにログインが必要かどうかを尋ねるプロンプトが

ユーザーに表示されます。dnssec-trigger デーモンは、10 秒ごとに DNSSEC リゾルバーのプローブを続行します。dnssec-trigger グラフィカルユーティリティーの使用方法は、「Dnssec-trigger の使用」 を参照してください。

4.5.4. VPN が提供されるドメインサーバーおよびネームサーバー

VPN 接続のタイプによっては、VPN トンネル設定の一環として、ドメインとそのドメインに使用するネームサーバーのリストを伝達できます。Red Hat Enterprise Linux では、これは NetworkManager でサポートされています。つまり、unbound なdnssec-trigger と NetworkManager の組み合わせは、VPN ソフトウェアが提供するドメインおよびネームサーバーを適切にサポートできます。VPN トンネルが起動すると、受信したドメイン名のすべてのエントリーに対してローカルの unbound キャッシュがフラッシュされるため、ドメイン名内の名前に対するクエリーは VPN を使用して到達した内部ネームサーバーから新たにフェッチされます。VPN トンネルが終了すると、バインド されていないキャッシュが再度フラッシュされ、ドメインに対するクエリーが以前に取得したプライベート IP アドレスではなくパブリック IP アドレスを返すようにします。「接続が提供されるドメインの DNSSEC 検証の設定」を参照してください。

4.5.5. 推奨される命名プラクティス

Red Hat は、static および transient の両方の *名前が host.example.com などの DNS 内のマシンに使用される完全修飾ドメイン* 名(FQDN)と一致することを推奨します。

Internet Corporation for Assigned Names and Numbers (ICANN)は、以前に登録解除されたトップレベルドメイン(.yourcompanyなど)をパブリックレジスターに追加することがあります。このため、Red Hat では、プライベートネットワーク上であっても委任されていないドメイン名を使用しないことを強く推奨しています。その結果、ネットワークリソースは利用できなくなります。また、委任されていないドメイン名を使うと、DNSSEC の実装および維持がより困難になります。これは、ドメイン名の競合が DNSSEC 検証の有効化に手動の設定ペナルティーを必要とするためです。この問題の詳細は、ドメイン名の衝突に関する ICANN のよくある質問を参照してください。

4.5.6. トラストアンカーについて

階層暗号化システムでは、*トラストアンカー*は信頼できると想定される、信頼できるエンティティーです。たとえば、X.509 アーキテクチャーでは、ルート証明書がトラストチェーンの元となるトラストアンカーとなっています。トラストアンカーは、パスの検証ができるように、事前に信頼できる団体が所有しておく必要があります。

DNSSEC のコンテキストでは、トラストアンカーは、その名前に関連付けられた DNS 名と公開鍵（または公開鍵のハッシュ）で設定されます。これは、base 64 でエンコードされたキーとして表されます。DNS レコードの検証および認証に使用できる公開鍵を含む情報を交換する手段である点で、証明書と似ています。RFC 4033 は、トラストアンカーを設定された DNSKEY RR または DNSKEY RR の DS RR ハッシュと定義しています。検証用セキュリティー対応リゾルバーは、この公開鍵またはハッシュを、署名された DNS 応答への認証チェーンを構築するための開始点として使用します。一般に、検証用リゾルバーは、DNS プロトコルの外部にある安全な手段または信頼できる手段を介してト

ラストアンカーの初期値を取得する必要があります。トラストアンカーの存在はまた、リゾルバーがトラストアンカーが指すゾーンが署名されていることを想定すべきことを意味します。

4.5.7. DNSSEC のインストール

4.5.7.1. unbound のインストール

マシン上でローカルに DNSSEC を使用して DNS を検証するには、DNS リゾルバーを unbound（または bind ）にインストールする必要があります。モバイルデバイスに dnssec-trigger をインストールするだけで済みます。サーバーの場合は、サーバーが置かれている場所(LAN またはインターネット)によってはローカルドメインの転送設定が必要になる場合がありますが、unbound で十分です。dnssec-trigger は現在、グローバルパブリック DNS ゾーンでのみ役立ちます。NetworkManager、dhclient、VPN アプリケーションは、多くの場合、ドメインリスト（およびネームサーバーリスト）を自動的に収集できますが、dnssec-trigger や unbound は収集できません。

unbound をインストールするには、root ユーザーとして次のコマンドを入力します。

```
~]# yum install unbound
```

4.5.7.2. unbound の稼働確認

unbound デーモンが実行中かどうかを確認するには、次のコマンドを入力します。

```
~]$ systemctl status unbound
unbound.service - Unbound recursive Domain Name Server
  Loaded: loaded (/usr/lib/systemd/system/unbound.service; disabled)
  Active: active (running) since Wed 2013-03-13 01:19:30 CET; 6h ago
```

systemctl status コマンドは、バインド されていないサービスが実行されていない場合に、unbound を Active: inactive (dead) として報告します。

4.5.7.3. unbound の起動

現在のセッションで unbound デーモンを起動するには、root ユーザーとして次のコマンドを入力します。

```
~]# systemctl start unbound
```

systemctl enable コマンドを実行して、システムが起動するたびに unbound が起動するようにします。

```
~]# systemctl enable unbound
```

unbound デーモンは、以下のディレクトリーを使用したローカルデータまたはオーバーライドの設定を許可します。

- /etc/unbound/conf.d ディレクトリーは、特定のドメイン名の設定を追加するために使用されます。これは、ドメイン名のクエリーを特定の DNS サーバーにリダイレクトするために使用されます。これは、企業の WAN 内にのみ存在するサブドメインによく使われます。

- /etc/unbound/keys.d ディレクトリーは、特定のドメイン名のトラストアンカーを追加するために使用されます。これは、内部のみの名前が DNSSEC 署名されているが、信頼のパスを構築するための公開されている DS レコードがない場合に必要です。もう一つのユースケースは、あるドメインの内部バージョンが、企業 WAN の外で一般に利用可能な名前とは異なる DNSKEY を使用して署名されている場合になります。

- /etc/unbound/local.d ディレクトリーは、特定の DNS データをローカルオーバーライドとして追加するために使用されます。これは、ブラックリストを作成したり、手動オーバーライドを作成したりするために使用できます。このデータは unbound によってクライアントに返されますが、DNSSEC 署名としてマークされません。

NetworkManager と一部の VPN ソフトウェアは、設定を動的に変更することがあります。これらのコンフィギュレーションディレクトリーには、コメントアウトされたサンプルエントリーが含まれています。詳細は、unbound.conf (5) の man ページを参照してください。

4.5.7.4. Dnssec-trigger のインストール

dnssec-trigger アプリケーションはデーモン dnssec-triggerd として実行されます。dnssec-trigger をインストールするには、root ユーザーとして次のコマンドを入力します。

```
~]# yum install dnssec-trigger
```

4.5.7.5. dnssec-trigger デーモンが動作しているかどうかの確認

dnssec-triggerd が実行されているかどうかを確認するには、次のコマンドを入力します。

```
~]$ systemctl status dnssec-triggerd
systemctl status dnssec-triggerd.service
dnssec-triggerd.service - Reconfigure local DNS(SEC) resolver on network change
   Loaded: loaded (/usr/lib/systemd/system/dnssec-triggerd.service; enabled)
   Active: active (running) since Wed 2013-03-13 06:10:44 CET; 1h 41min ago
```

dnssec-triggerd デーモンが実行していない場合は、systemctl status コマンドは dnssec-triggerd を Active: inactive (dead) として報告します。現行セッションで起動するには、root ユーザーとして次のコマンドを入力します。

```
~]# systemctl start dnssec-triggerd
```

systemctl enable コマンドを実行して、システムが起動するたびに dnssec-triggerd が起動するようにします。

```
~]# systemctl enable dnssec-triggerd
```

4.5.8. Dnssec-trigger の使用

dnssec-trigger アプリケーションには、DNSSEC プローブ結果を表示し、DNSSEC プローブ要求をオンデマンドで実行するための GNOME パネルユーティリティーがあります。ユーティリティーを起動するには、Super キーを押してアクティビティーの概要に入り、DNSSEC と 入力 して Enter を押します。画面下のメッセージトレイに、船の錨に似たアイコンが追加されます。画面の右下にある青い丸い通知アイコンを押して表示します。錨アイコンを右クリックすると、ポップアップメニューが表示されます。

通常の操作では、unbound はローカルでネームサーバーとして使用され、resolv.conf は 127.0.0.1 を指します。Hotspot Sign-On パネルで OK をクリックすると、これが変更されます。DNS サーバーは NetworkManager からクエリーされ、resolv.conf に配置されます。これで、Hotspot のサインオ

ンページで認証ができるようになりました。アンカーアイコンには、DNS クエリーが安全ではないことを警告するために、大きな赤い感嘆符が表示されます。認証されると、dnssec-trigger は自動的にこれを検出し、セキュアモードに戻す必要がありますが、場合によってはユーザーではなく、ユーザーは Reprobe を選択して手動でこれを行う必要があります。

dnssec-trigger は通常、ユーザーの対話を必要としません。一度起動するとバックグラウンドで動作し、問題が発生した場合は、ポップアップのテキストボックスでユーザーに通知します。また、resolv.conf ファイルへの変更について unbound に通知します。

4.5.9. DNSSEC における dig の使用

DNSSEC が機能しているかどうかを確認するために、さまざまなコマンドラインツールを使用することができます。最適なツールは、bind-utils パッケージの dig コマンドです。ldns パッケージからの drill と unbound パッケージの unbound-host に役立つその他のツール。古い DNS ユーティリティー nslookup および host は廃止されているため、使用しないでください。

dig を使用して DNSSEC データを要求するクエリーを送信するには、オプション +dnssec がコマンドに追加されます。以下に例を示します。

```
~]$ dig +dnssec whitehouse.gov
; <<>> DiG 9.9.3-rl.13207.22-P2-RedHat-9.9.3-4.P2.el7 <<>> +dnssec whitehouse.gov
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21388
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;whitehouse.gov.   IN A

;; ANSWER SECTION:
whitehouse.gov.  20 IN A 72.246.36.110
whitehouse.gov.  20 IN RRSIG A 7 2 20 20130825124016 20130822114016 8399
whitehouse.gov. BB8VHWEklaKpaLprt3hq1GkjDROvkmjYTBxiGhuki/BJn3PolGyrftxR
HH0377l0Lsybj/uZv5hL4UwWd/lw6Gn8GPikqhztAkgMxddMQ2lARP6p
wbMOKbSUuV6NGUT1WWwpbi+LelFMqQcAq3Se66iyH0Jem7HtgPEUE1Zc 3ol=

;; Query time: 227 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Aug 22 22:01:52 EDT 2013
;; MSG SIZE  rcvd: 233
```

A レコードに加えて、DNSSEC 署名、および署名の開始時刻と有効期限を含む RRSIG レコードが返されます。unbound サーバーは、上部の flags: セクションの ad ビットを返して、データが DNSSEC 認証されていることを示しています。

DNSSEC の検証に失敗すると、dig コマンドは SERVFAIL エラーを返します。

```
~]$ dig badsign-a.test.dnssec-tools.org
; <<>> DiG 9.9.3-rl.156.01-P1-RedHat-9.9.3-3.P1.el7 <<>> badsign-a.test.dnssec-tools.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 1010
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;badsign-a.test.dnssec-tools.org. IN A

;; Query time: 1284 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Aug 22 22:04:52 EDT 2013
;; MSG SIZE  rcvd: 60]
```

障害に関する詳細情報を要求するには、dig コマンドに +cd オプションを指定して DNSSEC チェックを無効にすることができます。

```
~]$ dig +cd +dnssec badsign-a.test.dnssec-tools.org
; <<>> DiG 9.9.3-rl.156.01-P1-RedHat-9.9.3-3.P1.el7 <<>> +cd +dnssec badsign-a.test.dnssec-
tools.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26065
;; flags: qr rd ra cd; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;badsign-a.test.dnssec-tools.org. IN A

;; ANSWER SECTION:
badsign-a.test.dnssec-tools.org. 49 IN A 75.119.216.33
badsign-a.test.dnssec-tools.org. 49 IN RRSIG A 5 4 86400 20130919183720 20130820173720
19442 test.dnssec-tools.org.
E572dLKMvYB4cgTRyAHlKKEvdOP7tockQb7hXFNZKVbfXbZJOlDREJrr
zCgAfJ2hykfY0yJHAlnuQvM0s6xOnNBSvc2xLlybJdfTaN6kSR0YFdYZ
n2NpPctn2kUBn5UR1BJRin3Gqy20LZlZx2KD7cZBtieMsU/lunyhCSc0 kYw=

;; Query time: 1 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Aug 22 22:06:31 EDT 2013
;; MSG SIZE  rcvd: 257
```

多くの場合、DNSSEC の間違いは、開始時間や有効期限が不適切であることから明らかになりますが、この例では、www.dnssec-tools.org の人々がこの RRSIG 署名を故意に理解不能にしており、この出力を手動で確認しても検出することはできません。このエラーは systemctl status unbound の出力に表示され、unbound デーモンはこれらのエラーを以下のように syslog に記録します。

```
Aug 22 22:04:52 laptop unbound: [3065:0] info: validation failure badsign-a.test.dnssec-tools.org. A IN
```

unbound-host を使用する例：

```
~]$ unbound-host -C /etc/unbound/unbound.conf -v whitehouse.gov
whitehouse.gov has address 184.25.196.110 (secure)
whitehouse.gov has IPv6 address 2600:1417:11:2:8800::fc4 (secure)
whitehouse.gov has IPv6 address 2600:1417:11:2:8000::fc4 (secure)
whitehouse.gov mail is handled by 105 mail1.eop.gov. (secure)
whitehouse.gov mail is handled by 110 mail5.eop.gov. (secure)
whitehouse.gov mail is handled by 105 mail4.eop.gov. (secure)
whitehouse.gov mail is handled by 110 mail6.eop.gov. (secure)
whitehouse.gov mail is handled by 105 mail2.eop.gov. (secure)
whitehouse.gov mail is handled by 105 mail3.eop.gov. (secure)
```

4.5.10. Dnssec-trigger の Hotspot 検出インフラストラクチャーのセットアップ

ネットワークに接続すると、dnssec-trigger は Hotspot を検出しようとします。Hotspot は通常、ユーザーがネットワークリソースを使用する前に、Web ページとのユーザーインタラクションを強制するデバイスです。検出は、既知のコンテンツを含む特定の固定 Web ページのダウンロードを試みることによって行われます。Hotspot がある場合、受信したコンテンツは想定どおりではありません。

dnssec-trigger が Hotspot を検出するために使用できる既知のコンテンツを含む固定 Web ページを設定するには、次の手順に従います。

1. インターネット上で公開されているマシンに Web サーバーをセットアップします。Red Hat Enterprise Linux 7 システム管理者ガイドの Web Servers の章を参照してください。

2. サーバーを実行したら、既知のコンテンツを含む static ページを公開します。ページは有効な HTML ページである必要はありません。たとえば、文字列 OK のみを含む hotspot.txt と

いう名前のプレーンテキストファイルを使用できます。サーバーが example.com にあり、Web サーバーの *document_root*/static/ サブディレクトリーに hotspot.txt ファイルをパブリッシュした場合、静的 Web ページのアドレスは example.com/static/hotspot.txt になります。Red Hat Enterprise Linux 7 システム管理者のガイドの Web サーバー の章の DocumentRoot ディレクティブを参照してください。

3. /etc/dnssec-trigger/dnssec-trigger.conf ファイルに以下の行を追加します。

```
url: "http://example.com/static/hotspot.txt OK"
```

このコマンドは、HTTP （ポート 80）を使用してプローブされる URL を追加します。最初の部分は、解決される URL とダウンロードされるページです。コマンドの 2 番目の部分は、ダウンロードされた Web ページが含むと予想されるテキスト文字列です。

設定オプションの詳細は、man ページの dnssec-trigger.conf (8)を参照してください。

4.5.11. 接続が提供されるドメインの DNSSEC 検証の設定

デフォルトでは、適切なネームサーバーを持つ正引きゾーンは、NetworkManager を介した Wi-Fi 接続を除き、すべての接続で dnssec-trigger によって unbound に自動的に追加されます。デフォルトでは、unbound に追加されたすべての転送ゾーンは DNSSEC 検証済みです。

転送ゾーンの検証に関するデフォルトの動作は、デフォルトではすべての転送ゾーンが DNSSEC で検証されない ように、変更することができます。これを行うには、dnssec-trigger 設定ファイル /etc/dnssec.conf の validate_connection_provided_zones 変数を変更します。root ユーザーとして、以下のように行を開いて編集します：

```
validate_connection_provided_zones=no
```

この変更は、既存の正引きゾーンではなく、将来の正引きゾーンに対してのみ行われます。したがって、現在提供されているドメインの DNSSEC を無効にする場合は、再接続する必要があります。

4.5.11.1. Wi-Fi 提供ドメインの DNSSEC 検証の設定

Wi-Fi 提供ゾーンの転送ゾーンの追加を有効にすることができます。これを行うには、dnssec-trigger 設定ファイル /etc/dnssec.conf の add_wifi_provided_zones 変数を変更します。root ユーザーとして、以下のように行を開いて編集します：

```
add_wifi_provided_zones=yes
```

この変更は、既存の正引きゾーンではなく、将来の正引きゾーンに対してのみ行われます。したがって、現在の Wi-Fi 提供ドメインで DNSSEC を有効にする場合は、Wi-Fi 接続を再接続 (再起動) する必要があります。

警告

Wi-Fi 提供ドメインを転送ゾーンとして unbound に追加をオンにすると、以下のようなセキュリティー 上 の影響が生じる可能性があります。

1. Wi-Fi アクセスポイントは、権限を持たない DHCP を介してドメインを意図的に提供し、すべての DNS クエリーを DNS サーバーにルーティングできます。

2. 正引きゾーンの DNSSEC 検証が オフ になっている場合、Wi-Fi が提供する DNS サーバーは、知らなくても提供されたドメインからドメイン名の IP アドレスをスプーフィングできます。

4.5.12. 関連情報

DNSSEC の詳細については、以下を参照してください。

4.5.12.1. インストールされているドキュメント

- dnssec-trigger (8) man ページ - dnssec-triggerd、dnssec-trigger-control、dnssec-trigger-panel のコマンドオプションを説明しています。
- dnssec-trigger.conf (8) man ページ - dnssec-triggerd の設定オプションを説明しています。
- unbound (8) man ページ： unbound のコマンドオプションを説明しています。これは、DNS 検証リゾルバーです。
- unbound.conf (5) man ページ - unbound の設定方法に関する情報が含まれています。
- resolv.conf (5) man ページ - リゾルバールーチンが読み込む情報が含まれています。

4.5.12.2. オンラインドキュメント

http://tools.ietf.org/html/rfc4033

RFC 4033 DNS Security Introduction and Requirements.

http://www.dnssec.net/

DNSSEC に関する多くのリソースへのリンクがある Web サイト。

http://www.dnssec-deployment.org/

米国国土安全保障省が後援する DNSSEC デプロイメントイニシアチブには、多くの DNSSEC 情報が含まれており、DNSSEC のデプロイメントに関する問題を議論するためのメーリングリストもあります。

http://www.internetsociety.org/deploy360/dnssec/community/

DNSSEC のデプロイメントを刺激し、調整するための Internet Society の 「Deploy 360」 イニシアチブは、世界中のコミュニティーと DNSSEC 活動を見つけるための優れたリソースです。

http://www.unbound.net/

本書には、バインドされていない DNS サービスに関する一般的な情報が記載されています。

http://www.nlnetlabs.nl/projects/dnssec-trigger/

本書には、dnssec-trigger に関する一般的な情報が記載されています。

4.6. LIBRESWAN を使った仮想プライベートネットワーク (VPN) の保護

Red Hat Enterprise Linux 7 では、Libreswan アプリケーションでサポートされている IPsec プロトコルを使用して、*仮想プライベートネットワーク* (VPN)を設定できます。Libreswan は、Openswan アプリケーションの継続であり、Openswan ドキュメントの多くの例は Libreswan と相互に置き換え可能です。NetworkManager IPsec プラグインは、NetworkManager-libreswan と呼ばれます。GNOME Shell をお使いの方は、NetworkManager-libreswan を依存関係に持つ NetworkManager-libreswan-gnome パッケージをインストールする必要があります。NetworkManager-libreswan-gnome パッケージは、オプションチャネルからのみ利用可能である

ことに注意してください。Enabling Supplementary and Optional Repositoriesを参照してください。

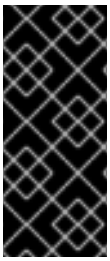
VPN の IPsec プロトコル自体は、*Internet Key Exchange* (IKE)プロトコルを使用して設定されます。IPsec と IKE は同義語です。IPsec VPN は、IKE VPN、IKEv2 VPN、XAUTH VPN、Cisco VPN、または IKE/IPsec VPN とも呼ばれます。*Level 2 Tunneling Protocol* (L2TP)も使用する IPsec VPN のバリアントは、通常 L2TP/IPsec VPN と呼ばれます。これには、Optional チャンネル xl2tpd アプリケーションが必要です。

Libreswan は、Red Hat Enterprise Linux 7 で利用可能なオープンソースのユーザー空間の IKE 実装です。IKE バージョン 1 および 2 は、ユーザーレベルのデーモンとして実装されます。IKE プロトコル自体も暗号化されています。IPsec プロトコルは Linux カーネルで実装され、Libreswan は、VPN トンネル設定を追加および削除するようにカーネルを設定します。

IKE プロトコルは UDP ポート 500 および 4500 を使用します。IPsec プロトコルは、プロトコル番号 50 を持つ *Encapsulated Security Payload* (ESP)とプロトコル番号 51 の *Authenticated Header* (AH)の 2 つの異なるプロトコルで設定されます。AH プロトコルの使用は推奨されません。AH のユーザーは、null 暗号化で ESP に移行することが推奨されます。

IPsec プロトコルには、Tunnel Mode （デフォルト）と Transport Mode の 2 つの異なる操作モードがあります。IKE を使用せずに IPsec を使用してカーネルを設定できます。これは 手動キー設定 と呼ば れます。ip xfrm コマンドを使用して手動キーを設定することは可能ですが、セキュリティー上の理由から、これは強く推奨されません。Libreswan は、netlink を使用して Linux カーネルとインターフェイスします。Linux カーネルでパケットの暗号化と復号が行われます。

Libreswan は、NSS(*Network Security Services*)暗号化ライブラリーを使用します。libreswan および NSS はともに、*連邦情報処理標準* (FIPS) の公開文書 140-2 での使用が認定されています。



重要

Libreswan および Linux カーネルが実装する IKE/IPsec VPN は、Red Hat Enterprise Linux 7 での使用が推奨される唯一の VPN 技術です。その他の VPN 技術は、そのリスクを理解せずに使用しないでください。

4.6.1. Libreswan のインストール

Libreswan をインストールするには、root で以下のコマンドを入力します。

```
~]# yum install libreswan
```

Libreswan がインストールされていることを確認するには、次のコマンドを実行します。

```
~]$ yum info libreswan
```

Libreswan を新規インストールしたら、インストールプロセスの一部として NSS データベースを初期化する必要があります。新しいデータベースを開始する前に、次のように古いデータベースを削除します。

```
~]# systemctl stop ipsec
~]# rm /etc/ipsec.d/*db
```

次に、新しい NSS データベースを初期化するには、root で以下のコマンドを入力します。

```
~]# ipsec initnss
Initializing NSS database
```

FIPS モードで運用する場合のみ、NSS データベースをパスワードで保護する必要があります。FIPS モード用にデータベースを初期化する場合は、前のコマンドの代わりに以下を使用します。

```
~]# certutil -N -d sql:/etc/ipsec.d
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
Re-enter password:
```

Libreswan が提供する ipsec デーモンを起動するには、root で以下のコマンドを実行します。

```
~]# systemctl start ipsec
```

デーモンが実行中であることを確認するには、次のようにします。

```
~]$ systemctl status ipsec
* ipsec.service - Internet Key Exchange (IKE) Protocol Daemon for IPsec
   Loaded: loaded (/usr/lib/systemd/system/ipsec.service; disabled; vendor preset: disabled)
   Active: active (running) since Sun 2018-03-18 18:44:43 EDT; 3s ago
     Docs: man:ipsec(8)
           man:pluto(8)
           man:ipsec.conf(5)
  Process: 20358 ExecStopPost=/usr/sbin/ipsec --stopnflog (code=exited, status=0/SUCCESS)
  Process: 20355 ExecStopPost=/sbin/ip xfrm state flush (code=exited, status=0/SUCCESS)
```

```
  Process: 20352 ExecStopPost=/sbin/ip xfrm policy flush (code=exited, status=0/SUCCESS)
  Process: 20347 ExecStop=/usr/libexec/ipsec/whack --shutdown (code=exited, status=0/SUCCESS)
  Process: 20634 ExecStartPre=/usr/sbin/ipsec --checknflog (code=exited, status=0/SUCCESS)
  Process: 20631 ExecStartPre=/usr/sbin/ipsec --checknss (code=exited, status=0/SUCCESS)
  Process: 20369 ExecStartPre=/usr/libexec/ipsec/_stackmanager start (code=exited,
status=0/SUCCESS)
  Process: 20366 ExecStartPre=/usr/libexec/ipsec/addconn --config /etc/ipsec.conf --checkconfig
(code=exited, status=0/SUCCESS)
 Main PID: 20646 (pluto)
   Status: "Startup completed."
   CGroup: /system.slice/ipsec.service
           └─20646 /usr/libexec/ipsec/pluto --leak-detective --config /etc/ipsec.conf --nofork
```

システムの起動時に Libreswan が起動するようにするには、root で以下のコマンドを実行します。

```
~]# systemctl enable ipsec
```

中間およびホストベースのファイアウォールを設定して、ipsec サービスを許可します。ファイアウォールと特定のサービスの通過許可についての詳細は、5章 *ファイアウォールの使用* を参照してください。Libreswan では、ファイアウォールで以下のパケットを許可する必要があります。

- **Internet Key Exchange (IKE)プロトコルの UDP ポート 500 および 4500**
- **Encapsulated Security Payload (ESP) IPsec パケットのプロトコル 50**
- **Authenticated Header (AH) IPsec パケットのプロトコル 51 （一般的ではありません）**

Libreswan を使用して IPsec VPN を設定する 3 つの例を示します。最初の例は、2 つのホストを接続し、安全に通信するためのものです。2 つ目の例では、2 つのサイトを接続して 1 つのネットワークを形成します。3 番目の例は、このコンテキストでは *ロードウォリアー* と呼ばれるリモートユーザーをサポートすることです。

4.6.2. Libreswan を使用した VPN 設定の作成

IKE/IPsec はピアツーピアプロトコルであるため、Libreswan は 「送信 『元』 および宛先」 または 「サーバー」 および 「クライアント」 という用語を使用しません。終了点 (ホスト) を参照する場合は、代わりに「左」と「右」という用語を使用します。これにより、ほとんどの場合、両方のエンドポイントで同じ設定を使用できますが、多くの管理者は、常にローカルホストに 「左」 を、リモートホストに 「右」 を使用することを選択します。

エンドポイントの認証には、一般的に 4 つの方法があります。

- *Pre-Shared Keys* (PSK) は、最も簡単な認証メソッドです。PSK はランダムな文字で設定されており、長さが 20 文字以上になります。FIPS モードでは、PSK が、使用する整合性アルゴリズムにより、最低強度の要件を満たす必要があります。PSK の値は 64 文字以上にすることが推奨されます。

- Raw RSA 鍵は、静的なホスト間またはサブネット間の IPsec 設定で一般的に使用されます。ホストは、相互の公開 RSA 鍵を使用して手動で設定します。この方法は、数十以上のホストがすべて互いに IPsec トンネルを設定する必要がある場合、適切にスケーリングされません。

- X.509 証明書は、共通の IPsec ゲートウェイに接続する必要があるホストが多数ある大規模なデプロイメントで一般的に使用されます。中央の *認証局* (CA) は、ホストまたはユーザーの RSA 証明書の署名に使用されます。この中央 CA は、個別のホストまたはユーザーの取り消しを含む、信頼のリレーを行います。

- NULL 認証は、認証なしでメッシュの暗号化を取得するために使用されます。これは、パッシブ攻撃は防ぎますが、アクティブ攻撃は防ぎません。ただし、IKEv2 は非対称認証メソッドを許可するため、NULL 認証は、インターネットスケール Opportunistic IPsec にも使用できます。この場合、クライアントはサーバーを認証しますが、サーバーはクライアントを認証しません。このモデルは、TLS (https:// websites としても知られている)を使用してセキュアな Web サイトと似ています。

これらの認証方法に加えて、量子コンピューターからの考えられる攻撃から保護するために、認証を追加することができます。この追加認証方式は、*Postquantum Preshared Keys* (PPK と呼ばれます。個々のクライアントまたはクライアントグループは、帯域幅を設定した事前共有鍵に対応する (PPKID) を指定して、独自の PPK を使用できます。「量子コンピューターに対する保護の使用」を参照してください。

4.6.3. Libreswan を使用したホスト間 VPN の作成

Libreswan が、左と右と呼ばれる 2 つのホスト間に、ホスト間の IPsec VPN を作成するように設定するには、両ホスト(「『左』」 および 「『右』」 の)で以下のコマンドを root として入力し、新しい生の RSA 鍵ペアを作成します。

```
~]# ipsec newhostkey --output /etc/ipsec.d/hostkey.secrets
Generated RSA key pair with CKAID 14936e48e756eb107fa1438e25a345b46d80433f was stored in
the NSS database
```

これでホストの RSA 鍵のペアが生成されます。RSA キーを生成するプロセスは、特にエントロ

ピーの低い仮想マシンでは、数分かかる場合があります。

ホストの公開鍵を表示して、「左側」 の設定で指定できるようにするには、「newhostkey」 コマンドで返された CKAID を使用して、新しいホストキーが追加されたホストで以下のコマンドを root として発行します。

```
~]# ipsec showhostkey --left --ckaid 14936e48e756eb107fa1438e25a345b46d80433f
 # rsakey AQPFKElpV

leftrsasigkey=0sAQPFKElpV2GdCF0Ux9Kqhcap53Kaa+uCgduoT2I3x6LkRK8N+GiVGkRH4Xg+WMrz
Rb94kDDD8m/BO/Md+A30u0NjDk724jWuUU215rnpwvbdAob8pxYc4ReSgjQ/DkqQvsemoeF4kimMU1
OBPNU7lBw4hTBFzu+iVUYMELwQSXpremLXHBNIamUbe5R1+ibgxO19l/PAbZwxyGX/ueBMBvSQ+H
0UqdGKbq7UgSEQTFa4/gqdYZDDzx55tpZk2Z3es+EWdURwJOgGiiiIFuBagasHFpeu9Teb1VzRyytny
NiJCBVhWVqsB4h6eaQ9RpAMmqBdBeNHfXwb6/hg+JIKJgjidXvGtgWBYNDpG40fEFh9USaFlSdiHO+
dmGyZQ74Rg9sWLtiVdlH1YEBUtQb8f8FVry9wSn6AZqPlpGgUdtkTYUCaaifsYH4hoIA0nku4Fy/Ugej8
9ZdrSN7Lt+igns4FysMmBOl9Wi9+LWnfl+dm4Nc6UNgLE8kZc+8vMJGkLi4SYjk2/MFYgqGX/COxSCPB
FUZFiNK7Wda0kWea/FqE1heem7rvKAPIiqMymjSmytZl9hhkCD16pCdgrO3fJXsfAUChYYSPyPQClkav
vBL/wNK9zlaOwssTaKTj4Xn90SrZaxTEjpqUeQ==
```

このキーは、以下に説明するように、両方のホストの設定ファイルに追加するために必要です。CKAID を忘れてしまった場合は、以下を使用してマシン上のすべてのホスト鍵のリストを取得することができます。

```
~]# ipsec showhostkey --list
< 1 >  RSA keyid: AQPFKElpV ckaid: 14936e48e756eb107fa1438e25a345b46d80433f
```

キーペアのシークレット部分は、/etc/ipsec.d/*.db にある 「NSS データベース」 内に保存されます。

このホスト間トンネルの設定ファイルを作成するには、上記の leftrsasigkey= と rightrsasigkey= の行を /etc/ipsec.d/ ディレクトリーにあるカスタム設定ファイルに追加します。

root で実行されているエディターを使用して、以下の形式で適切な名前でファイルを作成します。

/etc/ipsec.d/my_host-to-host.conf

以下のようにファイルを編集します。

```
conn mytunnel
    leftid=@west.example.com
    left=192.1.2.23
    leftrsasigkey=0sAQOrlo+hOafUZDlCQmXFrje/oZm [...] W2n417C/4urYHQkCvuIQ==
```

```
    rightid=@east.example.com
    right=192.1.2.45
    rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
    authby=rsasig
    # load and initiate automatically
    auto=start
```

また、公開鍵は RSAID ではなく、CKAID で設定することも可能です。この場合、「leftrsasigkey=」 の代わりに 「leftckaid=」 を使用します。

左のホストと右のホストの両方で、同一の設定ファイルを使用することができます。Libreswan は、指定された IP アドレスやホストネームをもとに、自動的に 「左」 か 「右」 かを検出します。ホストの 1 つがモバイルホストであり、その IP アドレスが事前に認識されていないことを意味する場合には、モバイルクライアントでは %defaultroute を IP アドレスとして使用します。これにより、動的 IP アドレスが自動的に選択されます。受信モバイルホストからの接続を受け入れる静的サーバーホストで、IP アドレスに %any を使用してモバイルホストを指定します。

leftrsasigkey 値が 「左」 のホストから取得され、rightrsasigkey 値が 「右」 のホストから取得されていることを確認します。leftckaid と rightckaid を使用する場合も同じことが当てはまります。

ipsec を再起動して、新しい設定を読み取り、システムの起動時に開始するように設定されている場合には、トンネルが確立されていることを確認します。

```
~]# systemctl restart ipsec
```

auto=start オプションを使用する場合は、IPsec トンネルを数秒以内に確立する必要があります。root で以下のコマンドを入力して、トンネルを手動でロードおよび起動できます。

```
~]# ipsec auto --add mytunnel
~]# ipsec auto --up mytunnel
```

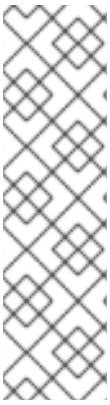
4.6.3.1. Libreswan を使用したホスト間 VPN の検証

IKE ネゴシエーションは UDP ポート 500 および 4500 で行われます。IPsec パケットは、Encapsulated Security Payload (ESP)パケットとして表示されます。ESP プロトコルにはポートがありません。VPN 接続が NAT ルーターを通過する必要がある場合、ESP パケットはポート 4500 の UDP パケットにカプセル化されます。

パケットが VPN トンネルを介して送信されていることを確認するには、root で以下の形式のコマンドを実行します。

```
~]# tcpdump -n -i interface esp or udp port 500 or udp port 4500
00:32:32.632165 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1a), length 132
00:32:32.632592 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1a), length 132
00:32:32.632592 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 7, length 64
00:32:33.632221 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1b), length 132
00:32:33.632731 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1b), length 132
00:32:33.632731 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 8, length 64
00:32:34.632183 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1c), length 132
00:32:34.632607 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1c), length 132
00:32:34.632607 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 9, length 64
00:32:35.632233 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1d), length 132
00:32:35.632685 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1d), length 132
00:32:35.632685 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 10, length 64
```

ここで、*interface* は、トラフィックを伝送することがわかっているインターフェイスです。**tcpdump** でキャプチャーを終了するには、**Ctrl+C** を押します。



注記

tcpdump コマンドは、**IPsec** と予期せず対話します。送信される暗号化されたパケットのみが表示され、送信されるプレーンテキストパケットは表示されません。暗号化された着信パケットと、復号化された着信パケットは表示されます。可能な場合は、エンドポイント自体ではなく、2 つのマシン間のルーターで **tcpdump** を実行します。**Virtual Tunnel Interface (VTI)**を使用する場合、物理インターフェイスの **tcpdump** は **ESP** パケットを表示し、**VTI** インターフェイスの **tcpdump** にはクリアテキストトラフィックが表示されます。

トンネルが正常に確立されたことを確認し、さらにトンネルを通過したトラフィックの量を確認するには、**root** で以下のコマンドを入力します。

```
~]# ipsec whack --trafficstatus
006 #2: "mytunnel", type=ESP, add_time=1234567890, inBytes=336, outBytes=336, id='@east'
```

4.6.4. Libreswan を使用したサイト間の VPN の設定

Libreswan がサイト間 **IPsec VPN** を作成し、2 つのネットワークを結合するために、1 つ以上のサブネットからのトラフィックが通過できるように設定された 2 つのホスト、エンドポイントの間に **IPsec** トンネルが作成されます。したがって、これらはネットワークのリモート部分へのゲートウェイと見なすことができます。サイト間の **VPN** の設定は、設定ファイル内で複数のネットワークまたはサブネットを指定する必要がある点のみが、ホスト間の **VPN** とは異なります。

サイト間 **IPsec VPN** を作成するように **Libreswan** を設定するには、最初に 「Libreswan を使用したホスト間 VPN の作成」 の説明に従ってホスト間 **IPsec VPN** を設定してから、ファイルを **/etc/ipsec.d/my_site-to-site.conf** などの適切な名前のファイルにコピーまたは移動します。**root** でエ

ディターを使用して、以下のようにカスタム設定ファイル /etc/ipsec.d/my_site-to-site.conf を編集します。

```
conn mysubnet
    also=mytunnel
    leftsubnet=192.0.1.0/24
    rightsubnet=192.0.2.0/24
    auto=start

conn mysubnet6
    also=mytunnel
    connaddrfamily=ipv6
    leftsubnet=2001:db8:0:1::/64
    rightsubnet=2001:db8:0:2::/64
    auto=start

conn mytunnel
    leftid=@west.example.com
    left=192.1.2.23
    leftrsasigkey=0sAQOrlo+hOafUZDlCQmXFrje/oZm [...] W2n417C/4urYHQkCvuIQ==
    rightid=@east.example.com
    right=192.1.2.45
    rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
    authby=rsasig
```

トンネルを起動するには、Libreswan を再起動するか、root で以下のコマンドを使用して、すべての接続を手動でロードして開始します。

```
~]# ipsec auto --add mysubnet
```

```
~]# ipsec auto --add mysubnet6
```

```
~]# ipsec auto --up mysubnet
104 "mysubnet" #1: STATE_MAIN_I1: initiate
003 "mysubnet" #1: received Vendor ID payload [Dead Peer Detection]
003 "mytunnel" #1: received Vendor ID payload [FRAGMENTATION]
106 "mysubnet" #1: STATE_MAIN_I2: sent MI2, expecting MR2
108 "mysubnet" #1: STATE_MAIN_I3: sent MI3, expecting MR3
003 "mysubnet" #1: received Vendor ID payload [CAN-IKEv2]
004 "mysubnet" #1: STATE_MAIN_I4: ISAKMP SA established {auth=OAKLEY_RSA_SIG
cipher=aes_128 prf=oakley_sha group=modp2048}
117 "mysubnet" #2: STATE_QUICK_I1: initiate
004 "mysubnet" #2: STATE_QUICK_I2: sent QI2, IPsec SA established tunnel mode
{ESP=>0x9414a615 <0x1a8eb4ef xfrm=AES_128-HMAC_SHA1 NATOA=none NATD=none
DPD=none}
```

```
~]# ipsec auto --up mysubnet6
003 "mytunnel" #1: received Vendor ID payload [FRAGMENTATION]
117 "mysubnet" #2: STATE_QUICK_I1: initiate
```

```
004 "mysubnet" #2: STATE_QUICK_I2: sent QI2, IPsec SA established tunnel mode
{ESP=>0x06fe2099 <0x75eaa862 xfrm=AES_128-HMAC_SHA1 NATOA=none NATD=none
DPD=none}
```

4.6.4.1. Libreswan を使用したサイト間 VPN の検証

VPN トンネルを通してパケットが送信されていることを確認する方法は、「Libreswan を使用したホスト間 VPN の検証」 で説明した手順と同じです。

4.6.5. Libreswan を使ったサイト間シングルトンネル VPN の設定

多くの場合、サイト間トンネルが構築されると、ゲートウェイはパブリック IP アドレスではなく内部 IP アドレスを使用して相互に通信する必要があります。これは、単一のトンネルを使用して実現することができます。ホスト名 west の左側のホストに、内部 IP アドレスが 192.0.1.254 で、ホスト名が east の正しいホストの内部 IP アドレスが 192.0. 2.254 の場合には、単一のトンネルを使用して以下の設定を両方のサーバーの /etc/ipsec.d/myvpn.conf ファイルに保存します。

```
conn mysubnet
    leftid=@west.example.com
    leftrsasigkey=0sAQOrlo+hOafUZDlCQmXFrje/oZm [...] W2n417C/4urYHQkCvuIQ==
    left=192.1.2.23
    leftsourceip=192.0.1.254
    leftsubnet=192.0.1.0/24
    rightid=@east.example.com
    rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
    right=192.1.2.45
    rightsourceip=192.0.2.254
    rightsubnet=192.0.2.0/24
    auto=start
    authby=rsasig
```

4.6.6. Libreswan を使用したサブネット押し出しの設定

多くの場合、IPsec はハブアンドスポークアーキテクチャーにデプロイされます。各リーフノードには、より大きな範囲の一部である IP 範囲があります。リーフはハブを介して相互に通信します。これは *サブネットの押出* と呼ばれます。

例4.2 単純なサブネット押し出しセットアップの設定

以下の例では、10.0.0.0/ 8 と、より小さな /24 サブネットを使用する 2 つのブランチを使用して、ヘッドオフィスを設定します。

本社では以下のようになります。

```
conn branch1
    left=1.2.3.4
    leftid=@headoffice
    leftsubnet=0.0.0.0/0
    leftrsasigkey=0sA[...]
    #
    right=5.6.7.8
    rightid=@branch1
    rightsubnet=10.0.1.0/24
    rightrsasigkey=0sAXXXX[...]
    #
    auto=start
    authby=rsasig

conn branch2
    left=1.2.3.4
    leftid=@headoffice
    leftsubnet=0.0.0.0/0
    leftrsasigkey=0sA[...]
    #
    right=10.11.12.13
    rightid=@branch2
    rightsubnet=10.0.2.0/24
    rightrsasigkey=0sAYYYY[...]
    #
    auto=start
    authby=rsasig
```

「branch1」 オフィスでは、同一の接続を使用します。さらに、パススルー接続を使用して、ローカル LAN トラフィックがトンネルを介して送信されないようにします。

```
conn branch1
    left=1.2.3.4
    leftid=@headoffice
    leftsubnet=0.0.0.0/0
    leftrsasigkey=0sA[...]
    #
    right=10.11.12.13
    rightid=@branch2
    rightsubnet=10.0.1.0/24
    rightrsasigkey=0sAYYYY[...]
    #
    auto=start
    authby=rsasig

conn passthrough
    left=1.2.3.4
    right=0.0.0.0
    leftsubnet=10.0.1.0/24
    rightsubnet=10.0.1.0/24
    authby=never
    type=passthrough
    auto=route
```

4.6.7. IKEv2 リモートアクセス VPN Libreswan の設定

ロードウォーリアーは、ラップトップなどの IP アドレスを動的に割り当てられたモバイルクライアントを持つ移動ユーザーです。これらは証明書を使用して認証されます。古い IKEv1 XAUTH プロトコルを使用する必要がないように、次の例では IKEv2 を使用しています。

サーバーの場合:

```
conn roadwarriors
    ikev2=insist
    # Support (roaming) MOBIKE clients (RFC 4555)
    mobike=yes
    fragmentation=yes
    left=1.2.3.4
    # if access to the LAN is given, enable this, otherwise use 0.0.0.0/0
    # leftsubnet=10.10.0.0/16
    leftsubnet=0.0.0.0/0
    leftcert=vpn-server.example.com
    leftid=%fromcert
    leftxauthserver=yes
    leftmodecfgserver=yes
    right=%any
    # trust our own Certificate Agency
    rightca=%same
    # pick an IP address pool to assign to remote users
    # 100.64.0.0/16 prevents RFC1918 clashes when remote users are behind NAT
    rightaddresspool=100.64.13.100-100.64.13.254
    # if you want remote clients to use some local DNS zones and servers
    modecfgdns="1.2.3.4, 5.6.7.8"
    modecfgdomains="internal.company.com, corp"
    rightxauthclient=yes
    rightmodecfgclient=yes
    authby=rsasig
    # optionally, run the client X.509 ID through pam to allow/deny client
    # pam-authorize=yes
    # load connection, don't initiate
    auto=add
    # kill vanished roadwarriors
    dpddelay=1m
    dpdtimeout=5m
    dpdaction=%clear
```

詳細は以下のようになります。

left=*1.2.3.4*

1.2.3.4 の値は、サーバーの実際の IP アドレスまたはホスト名を指定します。

leftcert=vpn-server.example.com

このオプションは、証明書をインポートする際に使用されたそのフレンドリーネームまたはニックネームを参照する証明書を指定します。通常、名前は .p12 ファイルの形式で PKCS #12 証明書バンドルの一部として生成されます。詳細は、pkcs12 (1) および pk12util (1) の man ページを参照してください。

ロードウォーリアーのデバイスであるモバイルクライアントでは、上記の設定に多少変更を加えて使用します。

```
conn to-vpn-server
    ikev2=insist
    # pick up our dynamic IP
    left=%defaultroute
    leftsubnet=0.0.0.0/0
    leftcert=myname.example.com
    leftid=%fromcert
    leftmodecfgclient=yes
    # right can also be a DNS hostname
    right=1.2.3.4
    # if access to the remote LAN is required, enable this, otherwise use 0.0.0.0/0
    # rightsubnet=10.10.0.0/16
    rightsubnet=0.0.0.0/0
    # trust our own Certificate Agency
    rightca=%same
    authby=rsasig
    # allow narrowing to the server's suggested assigned IP and remote subnet
    narrowing=yes
    # Support (roaming) MOBIKE clients (RFC 4555)
    mobike=yes
    # Initiate connection
    auto=start
```

詳細は以下のようになります。

auto=start

このオプションを使用すると、ユーザーは ipsec システムサービスが開始されるたびに VPN に接続できるようになります。後で接続を確立する場合は、auto=add に置き換えます。

4.6.8. X.509 を使用した IKEv1 リモートアクセス VPN Libreswan および XAUTH の設定

Libreswan は、XAUTH IPsec 拡張機能を使用して接続を確立する際に、ローミング VPN クライアントに IP アドレスと DNS 情報をネイティブに割り当てる方法を提供します。拡張認証 (XAUTH) は、PSK または X.509 証明書を使用してデプロイすることができます。X.509 を使用したデプロイの方がより安全です。クライアント証明書は、証明書失効リストまたは *オンライン証明書ステータスプロトコル* (OCSP) により取り消すことができます。X.509 証明書を使用すると、個々のクライアントがサーバーになりすますことはできません。グループパスワードとも呼ばれる PSK を使用すると、これは理論的には可能です。

XAUTH では、VPN クライアントがユーザー名とパスワードで自身を追加で識別する必要があります。Google 認証システムや RSA SecureID トークンなどのワンタイムパスワード (OTP) の場合、ワンタイムトークンがユーザーパスワードに追加されます。

XAUTH には 3 つの可能なバックエンドがあります。

xauthby=pam

これは、/etc/pam.d/pluto の設定を使用してユーザーを認証します。Pluggable Authentication Modules (PAM) は、さまざまなバックエンドを単独で使用するように設定できます。システムアカウントのユーザーパスワードスキーム、LDAP ディレクトリー、RADIUS サーバー、またはカスタムパスワード認証モジュールを使用できます。詳細は、Using Pluggable Authentication Modules (PAM) の章を参照してください。

xauthby=file

これは /etc/ipsec.d/passwd 設定ファイルを使用します(/etc/ipsec.d/nsspassword ファイルと混同しないでください)。このファイルのフォーマットは Apache .htpasswd ファイルと似ており、Apache htpasswd コマンドを使用してこのファイルのエントリーを作成できます。ただし、ユーザー名とパスワードの後に、使用される IPsec 接続の接続名を指定して 3 番目の列が必要になります。たとえば、conn remoteusers を使用してユーザーを削除するために VPN を提供する場合、パスワードファイルのエントリーは以下のようになります。

```
user1:$apr1$MIwQ3DHb$1I69LzTnZhnCT2DPQmAOK.:remoteusers
```

注記

htpasswd コマンドを使用する場合、各行の *user:password* 部分の後に接続名を手動で追加する必要があります。

xauthby=alwaysok

サーバーは常に、XAUTH ユーザーとパスワードの組み合わせが正しいふりをします。クライアントは引き続き、ユーザー名とパスワードを指定しなければなりませんが、サーバーはこれらを

無視します。これは、ユーザーがすでに X.509 証明書によって識別されている場合、または XAUTH バックエンドを必要とせずに VPN をテストする場合にのみ使用されるべきものです。

X.509 証明書を使用したサーバーの設定例:

```
conn xauth-rsa
    ikev2=never
    auto=add
    authby=rsasig
    pfs=no
    rekey=no
    left=ServerIP
    leftcert=vpn.example.com
    #leftid=%fromcert
    leftid=vpn.example.com
    leftsendcert=always
    leftsubnet=0.0.0.0/0
    rightaddresspool=10.234.123.2-10.234.123.254
    right=%any
    rightrsasigkey=%cert
    modecfgdns="1.2.3.4,8.8.8.8"
    modecfgdomains=example.com
    modecfgbanner="Authorized access is allowed"
    leftxauthserver=yes
    rightxauthclient=yes
    leftmodecfgserver=yes
    rightmodecfgclient=yes
    modecfgpull=yes
    xauthby=pam
    dpddelay=30
    dpdtimeout=120
    dpdaction=clear
    ike_frag=yes
    # for walled-garden on xauth failure
    # xauthfail=soft
    # leftupdown=/custom/_updown
```

xauthfail が hard ではなく soft に設定されている場合、認証の失敗は無視され、VPN はユーザーが適切に認証されたかのように設定されます。カスタムアップダウンスクリプトを使用して、環境変数 XAUTH_FAILED をチェックすることができます。このようなユーザーは、たとえば iptables DNAT を使用して、ウォードガーンにリダイレクトできます。ここで、管理者に問い合わせたり、サービスの有料サブスクリプションを更新したりできます。「」

VPN クライアントは、modecfgdomain 値と DNS エントリーを使用して、指定されたドメインのクエリーをこれらの指定されたネームサーバーにリダイレクトします。これにより、ローミングユーザーは内部 DNS 名を使用して内部専用リソースにアクセスすることができます。IKEv2 は modecfgdomains と modecfgdns を使用して、ドメイン名とネームサーバー IP アドレスのコンマ区切りリストをサポートしますが、IKEv1 プロトコルは 1 つのドメイン名のみをサポートし、libreswan は最大 2 つのネームサーバー IP アドレスのみをサポートすることに注意してください。オプション

で、バナーテキストを VPN クライアントに送信するには、modecfgbanner オプションを使用します。

leftsubnet が 0.0.0.0/0 ではない場合には、スプリットトンネリング設定要求はクライアントに自動的に送信されます。たとえば、leftsubnet=10.0.0.0/8 を使用する場合、VPN クライアントは VPN 経由で 10.0.0.0/8 のトラフィックのみを送信します。

クライアントでは、ユーザーはユーザーパスワードを入力する必要がありますが、これは使用するバックエンドに依存します。以下に例を示します。

xauthby=file

管理者がパスワードを生成し、/etc/ipsec.d/passwd ファイルに保存しました。

xauthby=pam

パスワードは、/etc/pam.d/pluto ファイルの PAM 設定で指定された場所で取得されます。

xauthby=alwaysok

パスワードはチェックされず、常に受け入れられます。このオプションは、テスト目的や、xauth のみのクライアントの互換性を確保したい場合に使用します。

関連情報

XAUTH の詳細については、Extended Authentication within ISAKMP/Oakley (XAUTH) インターネットドラフトのドキュメントを参照してください。

4.6.9. 量子コンピューターに対する保護の使用

事前共有鍵が設定されている IKEv1 を使用すると、量子攻撃者に対する保護が可能になります。IKEv2 の再設計は、この保護をネイティブに提供しません。Libreswan は、*Postquantum Preshared Keys* (PPK)を使用して、量子攻撃に対して IKEv2 接続を保護します。

任意の PPK 対応を有効にする場合は、接続定義に ppk=yes を追加します。PPK が必要な場合は ppk=insist を追加します。次に、各クライアントには、帯域外で通信する (および可能であれば量子攻撃に対して安全な) シークレット値を持つ PPK ID を付与できます。PPK はランダム性において非常に強力で、辞書の単語は使用しません。PPK ID および PPK データ自体は ipsec.secrets に保存されます。以下に例を示します。

```
@west @east : PPKS "user1" "thestringismeanttobearandomstr"
```

PPKS オプションは、静的な PPK を参照します。ワンタイムパッドベースのダイナミック PPK を使用する実験的な関数があります。各接続では、ワンタイムパッドの新しい部分が PPK として使用されます。これを使用すると、ファイル内の動的な PPK の部分がゼロで上書きされ、再利用を防ぐことができます。複数のタイムパッドマテリアルが残っていないと、接続は失敗します。詳細は、man ページの ipsec.secrets (5) を参照してください。

警告

動的の PPK の実装はテクノロジープレビューとして提供されており、この機能は注意して使用する必要があります。詳細は、Red Hat Enterprise Linux 7.5 Release Notes 参照してください。

4.6.10. 関連情報

以下の資料は、Libreswan および ipsec デーモンに関するその他のリソースを提供します。

4.6.10.1. インストールされているドキュメント

- man ページの ipsec (8) - ipsec のコマンドオプションが説明されています。
- man ページの ipsec.conf (5) - ipsec の設定に関する情報が記載されています。
- man ページの ipsec.secrets (5) - ipsec.secrets ファイルの形式が説明されています。
- man ページの ipsec_auto (8) - 鍵の自動交換を使用して確立された Libreswan IPsec 接続を操作する auto コマンドラインクライアントの使用方法が説明されています。
- man ページの ipsec_rsasigkey (8) - RSA 署名鍵の生成に使用されるツールが説明されています。

- /usr/share/doc/libreswan-*version*/

4.6.10.2. オンラインドキュメント

https://libreswan.org

アップストリームプロジェクトの Web サイトです。

https://libreswan.org/wiki

Libreswan プロジェクトの Wiki です。

https://libreswan.org/man/

Libreswan に関する全 man ページ

NIST Special Publication 800-77: Guide to IPsec VPNs

IPsec に基づくセキュリティーサービスの実装に関する組織への実用的なガイダンス。

4.7. OPENSSL の使用

OpenSSL は、アプリケーションに暗号化プロトコルを提供するライブラリーです。openssl コマンドラインユーティリティーは、シェルからの暗号化機能の使用を可能にします。これには、インタラクティブモードが含まれています。

openssl コマンドラインユーティリティーには、システムにインストールされている openssl のバージョンがサポートするコマンドに関する情報を提供する多数の疑似コマンドがあります。疑似コマンド list-standard-commands、list-message-digest-commands、および list-cipher-commands は、現在の openssl ユーティリティーで利用可能なすべての標準コマンド、メッセージダイジェストコマンド、または暗号コマンドの一覧を出力します。

疑似コマンド list-cipher-algorithms および list-message-digest-algorithms は、すべての暗号名とメッセージダイジェスト名を一覧表示します。擬似コマンド list-public-key-algorithms は、サポートされているすべての公開鍵アルゴリズムを一覧表示します。たとえば、サポートされている公開鍵アルゴリズムを一覧表示するには、次のコマンドを発行します。

```
~]$ openssl list-public-key-algorithms
```

疑似コマンド **no-*command-name*** は、指定された名前の ***command-name*** が使用可能かどうかをテストします。シェルスクリプトでの使用を目的としています。詳細は、**man openssl(1)** を参照してください。

4.7.1. 暗号鍵の作成および管理

OpenSSL では、公開鍵は対応する秘密鍵から派生します。したがって、アルゴリズムを決定した後の最初のステップは、秘密鍵を生成することです。以下の例では、秘密鍵を ***privkey.pem*** とします。たとえば、デフォルトのパラメーターを使用して **RSA** 秘密鍵を作成する場合は、次のコマンドを実行します。

```
~]$ openssl genpkey -algorithm RSA -out privkey.pem
```

RSA アルゴリズムは、以下のオプションをサポートしています。

- **rsa_keygen_bits:numbits — 生成されたキーのビット数。指定されていない場合は、1024 が使用されます。**
- **rsa_keygen_pubexp:value — RSA パブリック指数値。これは、大きな 10 進数値、または 0x で始まる場合は 16 進数の値になります。デフォルト値は 65537 です。**

たとえば、公開指数として **3** を使用して **2048** ビットの **RSA** 秘密鍵を作成するには、以下のコマンドを実行します。

```
~]$ openssl genpkey -algorithm RSA -out privkey.pem -pkeyopt rsa_keygen_bits:2048 \ -pkeyopt
rsa_keygen_pubexp:3
```

128 ビット **AES** とパスフレーズ **「hello」** を使用して、出力される秘密鍵を暗号化するには、次のコマンドを発行します。

```
~]$ openssl genpkey -algorithm RSA -out privkey.pem -aes-128-cbc -pass pass:hello
```

秘密鍵の生成に関する詳細は、man genpkey(1) を参照してください。

4.7.2. 証明書の生成

OpenSSL を使用して証明書を生成するには、秘密鍵が利用可能である必要があります。以下の例では、秘密鍵を ***privkey.pem*** とします。秘密鍵をまだ生成していない場合は、**「暗号鍵の作成および管**

理」 を参照してください。

認証局 (CA) に証明書に署名してもらうには、証明書を生成してから CA に送信して署名する必要があります。これは、証明書署名要求と呼ばれます。詳細は、「証明書署名要求の作成」 を参照してください。別の方法は、自己署名証明書を作成することです。詳細は、「自己署名証明書の作成」 を参照してください。

4.7.2.1. 証明書署名要求の作成

CA に送信するための証明書を作成するには、次の形式でコマンドを発行します。

```
~]$ openssl req -new -key privkey.pem -out cert.csr
```

これにより、デフォルトの PEM 形式(PEM)形式でエンコードされた cert.csr という名前の X.509 証明書が作成されます。*PEM という名前は、RFC1424 で説明されている 「Privacy Enhancement for Internet Electronic Mail」 に由来しています。代替 DER 形式で証明書ファイルを生成するには、-outform DER コマンドオプションを使用します。*

上記のコマンドを発行すると、証明書の distinguished name (DN) を作成するために、お客様や組織に関する情報を入力する画面が表示されます。以下の情報が必要になります。

- *2 文字の国コード*
- *州または県の名前*
- *市または自治体*
- *組織の名前*
- *組織内のユニット名*
- *ユーザー名もしくはシステムのホスト名*

- メールアドレス

req(1) man ページでは、PKCS# 10 証明書要求と生成ユーティリティーについて説明しています。証明書の作成プロセスで使用されるデフォルト設定は、/etc/pki/tls/openssl.cnf ファイルに含まれます。詳細は、openssl.cnf (5) の man ページを参照してください。

4.7.2.2. 自己署名証明書の作成

366 日間有効な自己署名証明書を生成するには、以下の形式でコマンドを発行します。

```
~]$ openssl req -new -x509 -key privkey.pem -out selfcert.pem -days 366
```

4.7.2.3. Makefile を使った証明書の作成

/etc/pki/tls/certs/ ディレクトリーには、make コマンドを使用して証明書を作成するのに使用できる Makefile が含まれています。使用説明書を表示するには、次のようにコマンドを発行します。

```
~]$ make -f /etc/pki/tls/certs/Makefile
```

または、ディレクトリーに移動し、以下のように make コマンドを実行します。

```
~]$ cd /etc/pki/tls/certs/
~]$ make
```

詳細は、make(1) の man ページを参照してください。

4.7.3. 証明書の確認

CA によって署名された証明書は、信頼できる証明書と呼ばれます。したがって、自己署名証明書は信頼できない証明書になります。verify ユーティリティーは、同じ SSL および S/MIME 機能を使用して、通常の操作で OpenSSL が使用する証明書を検証します。エラーが見つかった場合は報告され、他のエラーを報告するためにテストを続行しようとします。

PEM 形式の複数の個別の X.509 証明書を検証するには、次の形式のコマンドを発行します。

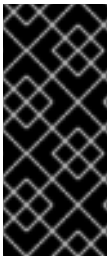
```
~]$ openssl verify cert1.pem cert2.pem
```

証明書チェーンを確認するには、リーフ証明書が cert.pem にあり、信頼しない中間証明書を

untrusted.pem に直接連結する必要があります。信頼できるルート CA 証明書は、/etc/pki/tls/certs/ca-bundle.crt または cacert.pem ファイルに一覧表示されているデフォルト CA に含まれている必要があります。次に、チェーンを確認するために、次の形式でコマンドを発行します。

```
~]$ openssl verify -untrusted untrusted.pem -CAfile cacert.pem cert.pem
```

詳細は、man verify(1) を参照してください。



重要

MD5 ハッシュアルゴリズムを使用した署名の検証は、このアルゴリズムの強度が不十分であるため、Red Hat Enterprise Linux 7 では無効になっています。SHA 256 などの強度の高いアルゴリズムを常に使用するようにしてください。

4.7.4. ファイルの暗号化および暗号化解除

OpenSSL でファイルを暗号化（および復号化）するには、pkeyutl または enc の組み込みコマンドのいずれかを使用できます。pkeyutl では、RSA キーを使用して暗号化と復号化を実行しますが、enc では対称アルゴリズムが使用されます。

RSA 鍵の使用

プレーンテキスト と呼ばれるファイルを暗号化するには、以下のコマンドを実行します。

```
~]$ openssl pkeyutl -in plaintext -out cyphertext -inkey privkey.pem
```

キーと証明書のデフォルトの形式は PEM です。必要に応じて、-keyform DER オプションを使用して DER キー形式を指定します。

暗号エンジンを指定するには、以下のように -engine オプションを使用します。

```
~]$ openssl pkeyutl -in plaintext -out cyphertext -inkey privkey.pem -engine id
```

ここで、id は、暗号化エンジンの ID です。エンジンの可用性を確認するには、次のコマンドを発行します。

```
~]$ openssl engine -t
```

プレーンテキスト と呼ばれるデータファイルに署名するには、次のようにコマンドを発行します。

```
~]$ openssl pkeyutl -sign -in plaintext -out sigtext -inkey privkey.pem
```

署名されたデータファイルを検証してデータを抽出するには、次のようにコマンドを発行します。

```
~]$ openssl pkeyutl -verifyrecover -in sig -inkey key.pem
```

DSA 鍵などを使って署名を検証するには、次のようなコマンドを発行します。

```
~]$ openssl pkeyutl -verify -in file -sigfile sig -inkey key.pem
```

pkeyutl(1) の man ページでは、公開鍵アルゴリズムユーティリティーについて説明しています。

シンメトリックアルゴリズムの使用

利用可能な対称暗号化アルゴリズムを一覧表示するには、**-l** などのサポート対象外のオプションを指定して **enc** コマンドを実行します。

```
~]$ openssl enc -l
```

アルゴリズムを指定するには、その名前をオプションとして使用します。たとえば、**aes-128-cbc** アルゴリズムを使用するには、以下の構文を使用します。

```
openssl enc -aes-128-cbc
```

a es-128-cbc アルゴリズムを使用して **plaintext** という名前のファイルを暗号化するには、以下のコマンドを入力します。

```
~]$ openssl enc -aes-128-cbc -in plaintext -out plaintext.aes-128-cbc
```

前の例で取得したファイルを復号化するには、次の例のように **-d** オプションを使用します。

```
~]$ openssl enc -aes-128-cbc -d -in plaintext.aes-128-cbc -out plaintext
```

重要

enc コマンドは AEAD 暗号を適切にサポートせず、ecb モードは安全とは見なされません。最適な結果を得るには、cbc、cfb、ofb、または ctr 以外のモードを使用しないでください。

4.7.5. メッセージダイジェストの生成

dgst コマンドは、提供されたファイルまたはファイルのメッセージダイジェストを 16 進数形式で生成します。また、このコマンドは、デジタル署名や検証にも使用することができます。message digest コマンドの形式は次のとおりです。

```
openssl dgst algorithm -out filename -sign private-key
```

ここで、algorithm は、md5|md4|md2|sha1|sha|mdc2|ripemd160|dss1 のいずれかになります。執筆時点では、SHA1 アルゴリズムが推奨されます。DSA を使った署名や検証が必要な場合は、-rand オプションで指定したランダムデータを含むファイルと一緒に dss1 オプションを使用する必要があります。

sha1 アルゴリズムを使用してデフォルトの Hex 形式でメッセージダイジェストを生成するには、次のコマンドを実行します。

```
~]$ openssl dgst sha1 -out digest-file
```

秘密鍵 privekey.pem を使用してダイジェストに電子署名を行うには、以下のコマンドを実行します。

```
~]$ openssl dgst sha1 -out digest-file -sign privkey.pem
```

詳細は、man dgst(1) を参照してください。

4.7.6. パスワードハッシュの生成

passwd コマンドは、パスワードのハッシュを計算します。コマンドラインでパスワードのハッシュを計算するには、次のようにコマンドを発行します。

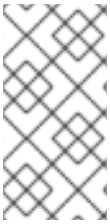
```
~]$ openssl passwd password
```

デフォルトでは、**-crypt** アルゴリズムが使用されます。

MD5 ベースの **BSD** アルゴリズム **1** を使用して、標準入力からパスワードのハッシュを計算するには、以下のコマンドを実行します。

```
~]$ openssl passwd -1 password
```

-apr1 オプションは、**BSD** アルゴリズムの **Apache** バリアントを指定します。



注記

openssl passwd -1 password コマンドは、**FIPS** モードが無効になっている場合にのみ使用してください。そうでない場合は、コマンドは機能しません。

ファイルに保存されているパスワードのハッシュを計算し、**salt xx** を使用するには、以下のコマンドを実行します。

```
~]$ openssl passwd -salt xx -in password-file
```

パスワードは標準出力に送られ、出力ファイルを指定する **-out** オプションはありません。**-table** は、対応するクリアテキストパスワードを含むパスワードハッシュのテーブルを生成します。

詳細および例は、**man sslpasswd(1)** を参照してください。

4.7.7. ランダムデータの生成

シードファイルを用いて、ランダムなデータを含むファイルを生成するには、次のコマンドを実行します。

```
~]$ openssl rand -out rand-file -rand seed-file
```

ランダムデータプロセスをシードする複数のファイルは、コロン（）をリスト区切り文字と し て使用して指定できます。

詳細は、man rand(1) を参照してください。

4.7.8. システムのベンチマーキング

特定のアルゴリズムのシステムの計算速度をテストするには、次の形式でコマンドを発行します。

```
~]$ openssl speed algorithm
```

ここで、algorithm は、使用する予定のサポートされているアルゴリズムの 1 つになります。利用可能なアルゴリズムを一覧表示するには、openssl speed と入力し、Tab を押します。

4.7.9. OpenSSL の設定

OpenSSL には、マスター設定ファイルと呼ばれる /etc/pki/tls/openssl.cnf 設定ファイルがあり、OpenSSL ライブラリーにより読み取られます。また、アプリケーションごとに個別の設定ファイルを用意することも可能です。設定ファイルには、[section_name] のようなセクション名を持つセクションが多数含まれています。最初の [section_name] までのファイルの最初の部分は、デフォルトのセクションと呼ばれることに注意してください。OpenSSL が設定ファイルで名前を検索する場合、名前付きセクションが最初に検索されます。すべての OpenSSL コマンドは、コマンド内で代替の設定ファイルを指定するためのオプションが使用されていない限り、マスター OpenSSL 設定ファイルを使用します。設定ファイルの詳細は、config (5) の man ページで説明されています。

2 つの RFC は、証明書ファイルの内容を説明しています。以下のとおりです。

- 『インターネット X.509 公開鍵インフラストラクチャー証明書および証明書失効リスト (CRL) プロファイル』
- 『インターネット X.509 公開鍵インフラストラクチャー証明書および証明書失効リスト (CRL) プロファイルへの更新』

4.8. STUNNEL の使用

stunnel プログラムは、クライアントとサーバー間の暗号化ラッパーです。設定ファイルで指定されたポートでリッスンし、クライアントとの通信を暗号化して、通常のポートでリッスンしている元のデーモンにデータを転送します。こうすることで、それ自体がどのタイプの暗号化もサポートしていないサービスを保護したり、POODLE SSL 脆弱性 (CVE-2014-3566) の影響を受ける SSL バージョン 2 および 3 など、セキュリティー上の理由から避けたいタイプの暗号化を使用しているサービスのセキュリティーを向上させたりすることができます。詳しくは

https://access.redhat.com/solutions/1234773 を参照してください。CUPS は、独自の設定で SSL を無効にする方法を提供しないコンポーネントの例です。

4.8.1. stunnel のインストール

root で以下のコマンドを入力して、stunnel パッケージをインストールします。

```
~]# yum install stunnel
```

4.8.2. stunnel を TLS Wrapper として設定する

stunnel を設定するには、次の手順に従います。

1. 使用するサービスに関係なく、stunnel の有効な証明書が必要です。適切な証明書がない場合は、認証局 に申請して取得するか、自己署名証明書を作成することができます。

警告

実稼働環境で実行されているサーバーには、常に認証局によって署名された証明書を使用してください。自己署名証明書は、テスト目的またはプライベートネットワークにのみ適しています。

認証局から付与される証明書の詳細については、「証明書署名要求の作成」 を参照してください。一方、stunnel の自己署名証明書を作成するには、/etc/pki/tls/certs/ ディレクトリーに移動し、root で以下のコマンドを入力します。

```
certs]# make stunnel.pem
```

すべての質問に答えて、プロセスを完了してください。

2. 証明書がある場合は、stunnel の設定ファイルを作成します。これは、すべての行がオプションまたはサービス定義の開始を指定するテキストファイルです。コメントと空の行をファイルに残して、コメントがセミコロンで始まる読みやすさを向上させることもできます。

stunnel RPM パッケージには、設定ファイルを保存できる /etc/stunnel/ ディレクトリーが含まれています。stunnel はファイル名やその拡張子の特別な形式を必要としませんが、/etc/stunnel/stunnel.conf を使用します。以下のコンテンツは、stunnel を TLS ラッパーとして設定します。

```
cert = /etc/pki/tls/certs/stunnel.pem
; Allow only TLS, thus avoiding SSL
sslVersion = TLSv1
chroot = /var/run/stunnel
setuid = nobody
setgid = nobody
pid = /stunnel.pid
socket = l:TCP_NODELAY=1
socket = r:TCP_NODELAY=1

[service_name]
accept = port
connect = port
TIMEOUTclose = 0
```

または、sslVersion = TLSv1 を含む行を以下の行に置き換えることで、SSL を回避することもできます。

```
options = NO_SSLv2
options = NO_SSLv3
```

オプションの目的は、以下のとおりです。

- cert — 証明書へのパス。
- sslVersion - SSL のバージョン。SSL と TLS は 2 つの独立した暗号化プロトコルですが、ここでは TLS を使用できることに注意してください。
- chroot — セキュリティーを強化するために stunnel プロセスが実行される変更された root ディレクトリー
- setuid, setgid - stunnel プロセスが実行されるユーザーおよびグループ。nobody は制限されたシステムアカウントです。
- pid - stunnel がプロセス ID を保存するファイル(chrootと相対的)

- *socket — ローカルおよびリモートのソケットオプション。この場合、ネットワーク遅延を改善するために Nagle のアルゴリズム を無効にします。*

- *[service_name] : サービス定義の先頭。この行の下で使用されるオプションは指定のサービスのみに適用されますが、上記のオプションは stunnel にグローバルに影響します。*

- *accept — リッスンするポート*

- *connect — 接続先のポート。これは、セキュリティー保護対象のサービスが使用するポートである必要があります*

- *TIMEOUTclose - クライアントからの close_notify アラートを待機する秒数。0 は stunnel にまったく待機しないように指示します。*

- *options — OpenSSL ライブラリーオプション*

> ***例4.3 CUPS のセキュア化***
>
> *stunnel を CUPS の TLS ラッパーとして設定するには、以下の値を使用します。*
>
> ```
> [cups]
> accept = 632
> connect = 631
> ```
>
> *632 の代わりに、任意の空きポートを使用できます。631 は CUPS が通常使用するポートです。*

3. *chroot ディレクトリーを作成し、setuid オプションで指定されたユーザーに、そのディレクトリーへの書き込みアクセス権を付与します。これを行うには、root で以下のコマンドを入力します。*

```
~]# mkdir /var/run/stunnel
~]# chown nobody:nobody /var/run/stunnel
```

これにより、stunnel が PID ファイルを作成できます。

4. システムが新しいポートへのアクセスを許可しないファイアウォール設定を使用している場合は、それに応じて変更してください。詳細は「GUI を使用してポートを開く」を参照してください。

5. 設定ファイルと chroot ディレクトリーを作成し、指定したポートにアクセスできることを確認したら、stunnel の使用を開始する準備が整います。

4.8.3. stunnel の起動、停止、再起動

stunnel を起動するには、root で以下のコマンドを入力します。

```
~]# stunnel /etc/stunnel/stunnel.conf
```

デフォルトでは、stunnel は /var/log/secure を使用して出力をログに記録します。

stunnel を終了するには、root で以下のコマンドを実行してプロセスを強制終了します。

```
~]# kill `cat /var/run/stunnel/stunnel.pid`
```

stunnel の実行中に設定ファイルを編集した場合は、stunnel を終了して再度起動し、変更を有効にします。

4.9. 暗号化

4.9.1. LUKS ディスクの暗号化の使用

Linux Unified Key Setup-on-disk-format (または LUKS) を使用すると、Linux コンピューター上のパーティションを暗号化できます。特に、モバイル PC やリムーバブルメディアに関しては特に重要です。LUKS を使用すれば、複数のユーザー鍵が、パーティションのバルク暗号化に使用されるマスター鍵を複号できるようになります。

LUKS の概要

LUKS の機能

-

LUKS は、ブロックデバイス全体を暗号化するため、脱着可能なストレージメディアやノート PC のディスクドライブといった、モバイルデバイスのコンテンツの保護に適しています。

- *暗号化されたブロックデバイスの基本的な内容は任意です。これにより、スワップデバイスの暗号化に役立ちます。また、とりわけデータストレージ用にフォーマットしたブロックデバイスを使用する特定のデータベースに関しても有用です。*
- *LUKS は、既存のデバイスマッパーのカーネルサブシステムを使用します。*
- *LUKS は、パラフレーズの強化を提供し、辞書攻撃から保護します。*
- *LUKS デバイスには複数のキースロットが含まれ、ユーザーはこれを使用してバックアップキーやパスフレーズを追加できます。*

LUKS が行わない こと

- *LUKS は、多くのユーザー (8 人以上) が同じデバイスへの個別のアクセスキーを持つことを必要とするシナリオには適していません。*
- *LUKS は、ファイルレベルの暗号化を必要とするアプリケーションには適していません。*

重要

LUKS などのディスク暗号化ソリューションは、システムの停止時にしかデータを保護しません。システムの電源がオンになり、LUKS がディスクを復号すると、そのディスクのファイルは、通常、そのファイルにアクセスできるすべてのユーザーが使用できます。

4.9.1.1. Red Hat Enterprise Linux における LUKS の実装

Red Hat Enterprise Linux 7 は LUKS を利用してファイルシステム暗号化を実行します。デフォルトではインストール時に、ファイルシステムを暗号化するオプションが指定されていません。ハードドライブを暗号化するオプションを選択すると、コンピューターを起動するたびに尋ねられるパスフレーズの入力を求められます。このパスフレーズは、あなたのパーティションを復号化するために使用されるバルク暗号化キーの "ロックを解除" します。デフォルトのパーティションテーブルの変更を選択すると、暗号化するパーティションを選択できます。この設定は、パーティションテーブル設定で行われます。

LUKS に使用されるデフォルトの暗号(cryptsetup --helpを参照)は aes-cbc-essiv:sha256 (ESSIV - Encrypted Salt-Sector Initialization Vector)です。インストールプログラム Anaconda は、デフォルトで XTS モード(aes-xts-plain64)を使用することに注意してください。LUKS のデフォルトの鍵サイズは 256 ビットです。Anaconda を使用した LUKS のデフォルトの鍵サイズ(XTS モード)は 512 ビットです。利用可能な暗号は以下のとおりです。

- AES (Advanced Encryption Standard) - FIPS PUB 197
- Twofish (128 ビットブロック暗号)
- Serpent
- cast5 - RFC 2144
- cast6 - RFC 2612

4.9.1.2. 手動でのディレクトリーの暗号化

警告

この手順を実行すると、暗号化するパーティションにあるすべてのデータが削除されます。すべての情報を失うことになります。この手順を開始する前に、必ずデータを外部ソースにバックアップしてください。

1. root でシェルプロンプトに次のように入力し、ランレベル 1 に入ります。

```
telinit 1
```

2. ***既存の /home をアンマウントします。***

```
umount /home
```

3. ***前の手順のコマンドが失敗した場合は、fuser を使用して、/home を占有しているプロセスを見つけ、それらを強制終了します。***

```
fuser -mvk /home
```

4. ***/home がマウントされていないことを確認します。***

```
grep home /proc/mounts
```

5. ***パーティションをランダムなデータで埋めます。***

```
shred -v --iterations=1 /dev/VG00/LV_home
```

このコマンドは、デバイスのシーケンシャル書き込み速度で進行し、完了するまでに時間がかかる場合があります。使用済みのデバイスに暗号化されていないデータが残らないようにし、ランダムなデータだけでなく、暗号化されたデータを含むデバイスの部分を難読化することが重要なステップとなります。

6. ***パーティションを初期化します。***

```
cryptsetup --verbose --verify-passphrase luksFormat /dev/VG00/LV_home
```

7. ***新たに暗号化したデバイスを開きます。***

```
cryptsetup luksOpen /dev/VG00/LV_home home
```

8. ***デバイスがあることを確認します。***

```
ls -l /dev/mapper | grep home
```

9. ファイルシステムを作成します。

```
mkfs.ext3 /dev/mapper/home
```

10. ファイルシステムをマウントします。

```
mount /dev/mapper/home /home
```

11. ファイルシステムが表示されていることを確認します。

```
df -h | grep home
```

12. 以下を **/etc/crypttab** ファイルに追加します。

```
home /dev/VG00/LV_home none
```

13. **/etc/fstab** ファイルを編集し、**/home** の古いエントリーを削除し、以下の行を追加します。

```
/dev/mapper/home /home ext3 defaults 1 2
```

14. **SELinux** のセキュリティーコンテキストをデフォルトに戻します。

```
/sbin/restorecon -v -R /home
```

15. マシンを再起動します。

```
shutdown -r now
```

16. **/etc/crypttab** のエントリーにより、コンピューターが起動時に **luks** のパスフレーズを要求します。

17. **root** でログインして、バックアップを復元してください。

これで、コンピューターの電源がオフのときにすべてのデータを安全に保管するための暗号化されたパーティションができました。

4.9.1.3. 既存のデバイスへの新しいパスフレーズの追加

既存のデバイスに新しいパスフレーズを追加するには、次のコマンドを使用します。

```
cryptsetup luksAddKey device
```

認証のために既存のパスフレーズのいずれかを入力するように求められた後、新しいパスフレーズを入力するように求められます。

4.9.1.4. 既存のデバイスからのパスフレーズ削除

既存のデバイスからパスフレーズを削除するには、次のコマンドを使用します。

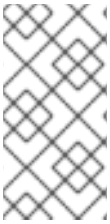
```
cryptsetup luksRemoveKey device
```

削除するパスフレーズの入力を求められ、次に認証用にに残りのパスフレーズのいずれかを入力するように求められます。

4.9.1.5. Anaconda での暗号化したブロックデバイスの作成

システムインストール時に、暗号化されたデバイスを作成することができます。これにより、暗号化されたパーティションを持つシステムを簡単に設定することができます。

ブロックデバイスの暗号化を有効にするには、自動パーティション設定を選択する際に **Encrypt System** チェックボックスをオンにするか、個別のパーティション、ソフトウェア **RAID** アレイ、または論理ボリュームを作成するときに **Encrypt** チェックボックスをオンにします。パーティション設定が完了すると、暗号化用のパスフレーズの入力が求められます。このパスフレーズは、暗号化されたデバイスにアクセスする際に必要となります。既存の **LUKS** デバイスがあり、インストール時に正しいパスフレーズを入力した場合、パスフレーズ入力ダイアログにもチェックボックスが表示されます。このチェックボックスをオンにすると、既存の各暗号化ブロックデバイスの使用可能なスロットに、新しいパスフレーズを追加する必要があることを意味します。



注記

自動パーティション設定 画面の システムの暗号化 チェックボックスを選択し、カスタムレイアウトの作成 を選択しても、ブロックデバイスは自動的に暗号化されません。

注記

キックスタート を使用して、新しい暗号化ブロックデバイスごとに個別のパスフレーズを設定できます。

4.9.1.6. 関連情報

LUKS または Red Hat Enterprise Linux 7 でのハードドライブの暗号化に関する詳細情報については、以下のリンクのいずれかを参照してください。

- LUKS ホームページ
- LUKS/cryptsetup FAQ
- LUKS - Linux Unified Key Setup (Wikipedia の記事)
- HOWTO: Creating an encrypted Physical Volume (PV) using a second hard drive and pvmove

4.9.2. GPG 鍵の作成

GPG は、あなた自身を識別し、あなたの知らない人とのコミュニケーションを含むあなたのコミュニケーションを認証するために使用されます。GPG を使用すると、GPG で署名された電子メールを読んでいる人は誰でも、その信頼性を確認できます。言い換えれば、GPG は、あなたが署名した通信が実際にあなたからのものであることを合理的に確信できるようにします。GPG は、サードパーティーがコードを変更したり、会話を傍受したり、メッセージを変更したりすることを阻止できるので便利です。

4.9.2.1. GNOME での GPG 鍵の作成

GNOME で GPG キーを作成するには、以下の手順に従います。

1. *Seahorse ユーティリティーをインストールします。これにより、GPG キーの管理が容易になります。*

```
~]# yum install seahorse
```

2. *キーを作成するには、Applications → Accessories メニューから Passwords and Encryption Keys を選択します。これにより、アプリケーションの Seahorse が起動します。*

3. *File メニューから New を選択してから PGP Key を選択します。次に、Continue をクリックします。*

4. *フルネーム、メールアドレス、および自身に関して説明するオプションのコメントを入力します (例:John C. Smith、jsmith@example.com、ソフトウェアエンジニア)。Create をクリックします。キーのパスフレーズを要求するダイアログが表示されます。強力かつ覚えやすいパスフレーズを選んでください。OK をクリックすると、キーが作成されます。*

警告

パスフレーズを忘れると、データの暗号解除ができなくなります。

GPG キー ID を見つけるには、新しく作成された鍵の横にある Key ID 列を確認します。ほとんどの場合、キー ID を求められた場合は、 0x 6789ABCD のように、キー ID の前に 0x を付けます。秘密鍵のバックアップを取り、安全な場所に保管してください。

4.9.2.2. KDE での GPG 鍵の作成

KDE で GPG キーを作成するには、以下の手順に従います。

1. *メインメニューから Applications → Utilities → Encryption Tool を選択して、KGpg プログラムを起動します。以前に KGpg を使用したことがない場合、プログラムは独自の GPG キーペアを作成するプロセスを順を追って説明します。*

2. *新しいキーペアを作成するように求めるダイアログボックスが表示されます。名前、メー*

ルアドレス、およびオプションのコメントを入力します。キーの有効期限、キーの強度 (ビット数) およびアルゴリズムを選択することもできます。

3. 次のダイアログボックスでパスフレーズを入力します。この時点で、キーが KGpg のメインウィンドウに表示されます。

警告

パスフレーズを忘れると、データの暗号解除ができなくなります。

GPG キー ID を見つけるには、新しく作成された鍵の横にある Key ID 列を確認します。ほとんどの場合、キー ID を求められた場合は、 0x 6789ABCD のように、キー ID の前に 0x を付けます。秘密鍵のバックアップを取り、安全な場所に保管してください。

4.9.2.3. コマンドラインを用いた GPG 鍵の生成

1. 次のシェルコマンドを使用します。

```
~]$ gpg2 --gen-key
```

このコマンドは、公開鍵と秘密鍵からなるキーペアを生成します。他の人はあなたの公開鍵を使用してあなたの通信を認証および復号化します。公開鍵をできるだけ広く配布します。特に、メーリングリストなど、認証された通信の受信希望者に配布します。

2. 一連のプロンプトがプロセスを指示します。必要に応じて、Enter キーを押してデフォルト値を割り当てます。最初のプロンプトでは、希望するキーの種類を選択するように求められます。

```
Please select what kind of key you want:
(1) RSA and RSA (default)
(2) DSA and Elgamal
(3) DSA (sign only)
(4) RSA (sign only)
Your selection?
```

ほとんどすべての場合、デフォルトが適切な選択となります。RSA/RSA 鍵は、通信の署

名だけでなく、ファイルの暗号化も可能にします。

3. 鍵のサイズを選択します。

```
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
```

この場合も、デフォルトの **2048** は、ほとんどすべてのユーザーにとって十分であり、非常に強力なセキュリティーレベルを表しています。

4. 鍵の有効期限を選択します。デフォルトの なし を使用する代わりに、有効期限を選択することが推奨され ます。たとえば、キーの電子メールアドレスが無効になった場合、有効期限があると、他の人にその公開キーの使用を停止するように知らせることができます。

```
Please specify how long the key should be valid.
0 = key does not expire
d = key expires in n days
w = key expires in n weeks
m = key expires in n months
y = key expires in n years
key is valid for? (0)
```

たとえば、**1y** の値を入力すると、キーは **1** 年間有効になります。(気が変わった場合は、キーが生成された後にこの有効期限を変更できます。)

5. **gpg2** アプリケーションが署名情報を要求する前に、以下のプロンプトが表示されます。

```
Is this correct (y/N)?
```

y を入力してプロセスを終了します。

6. **GPG** 鍵用に氏名と電子メールアドレスを入力します。このプロセスは、あなたが実在する人物であることを証明するためのものであることを忘れないでください。このため、本当の名前を入力してください。偽の電子メールアドレスを選択すると、他の人があなたの公開鍵を見

つけるのがより困難になります。これにより、通信の認証が困難になります。たとえば、この GPG キーをメーリングリストでの自己紹介に使用している場合は、そのリストで使用している電子メールアドレスを入力します。

コメントフィールドには、エイリアスなどの情報を記載してください。(人によっては、目的別にキーを使い分け、それぞれのキーに "Office" または "Open Source Projects" などのコメントをつけて識別しています)。

7. すべてのエントリーが正しい場合は、確認プロンプトで O を入力して続行するか、他のオプションを使用して問題を修正します。最後に、秘密鍵のパスフレーズを入力します。gpg2 プログラムでは、入力エラーが発生しないようにパスフレーズを 2 回入力するように求められます。

8. 最後に、gpg2 はランダムなデータを生成して、キーを可能な限り一意にします。この手順では、マウスを動かしたり、ランダムキーを入力したり、システムで他のタスクを実行したりして、プロセスを高速化します。この手順が終わると、キーは完成し、すぐに使えるようになります。

```
pub  1024D/1B2AFA1C 2005-03-31 John Q. Doe <jqdoe@example.com>
Key fingerprint = 117C FE83 22EA B843 3E86  6486 4320 545E 1B2A FA1C
sub  1024g/CEA4B22E 2005-03-31 [expires: 2006-03-31]
```

9. キーのフィンガープリントは、キーの省略形の "署名" です。これにより、実際の公開鍵を改ざんされることなく受け取ったことを他者に確認することができるのです。このフィンガープリントを書き留める必要はありません。いつでもフィンガープリントを表示するには、以下のコマンドを使用し、電子メールアドレスは置き換えます。

```
~]$ gpg2 --fingerprint jqdoe@example.com
```

"GPG キー ID" は、公開鍵を識別する 8 桁の 16 進数で設定されています。上記の例では、GPG キー ID は 1B2AFA1C です。ほとんどの場合、キー ID を求められた場合は、 0x6789ABCD のように、キー ID の前に 0x を付けます。

警告

パスフレーズを忘れると、キーは使用できなくなり、そのキーを使用して暗号化されたデータはすべて失われます。

4.9.2.4. 公開鍵の暗号化について

1. *Wikipedia - Public Key Cryptography*
2. *HowStuffWorks - Encryption*

4.9.3. 公開鍵暗号化における openCryptoki の使用

openCryptoki は PKCS#11 の Linux 実装で、トークンと呼ばれる暗号化デバイスへのアプリケーションプログラミングインターフェイス(API)を定義する 公開鍵暗号化標準 です。トークンは、ハードウェアまたはソフトウェアに実装することが可能です。本章では、openCryptoki システムを Red Hat Enterprise Linux 7 でインストール、設定、および使用する方法の概要を説明します。

4.9.3.1. openCryptoki のインストールとサービスの起動

テスト目的でトークンのソフトウェア実装を含む、基本的な openCryptoki パッケージをシステムにインストールするには、root で以下のコマンドを実行します。

```
~]# yum install opencryptoki
```

使用する予定のハードウェアトークンの種類によっては、特定のユースケースをサポートする追加のパッケージをインストールする必要がある場合があります。たとえば、Trusted Platform Module (TPM) デバイスのサポートを取得するには、opencryptoki-tpmtok パッケージをインストールする必要があります。

Yum パッケージマネージャーを 使用 してパッケージをインストールする方法は、Red Hat Enterprise Linux 7 システム管理者のガイドのパッケージのインストールセクションを参照してください。

openCryptoki サービスを有効にするには、pkcsslotd デーモンを実行する必要があります。root

で以下のコマンドを実行して、現行セッションで デーモンを起動します。

```
~]# systemctl start pkcsslotd
```

起動時に自動的にサービスが開始されるようにするには、次のコマンドを入力します。

```
~]# systemctl enable pkcsslotd
```

systemd ターゲットを使ってサービスを管理する方法の詳細については、Red Hat Enterprise Linux 7 システム管理者ガイド の Managing Services with systemd の章を参照してください。

4.9.3.2. openCryptoki の設定と使用

起動すると、pkcsslotd デーモンは /etc/opencryptoki/opencryptoki.conf 設定ファイルを読み取ります。このファイルは、システムで動作するように設定されたトークンとスロットに関する情報を収集します。

このファイルは、キーと値のペアを使用して個々のスロットを定義します。各スロット定義には、説明、使用するトークンライブラリーの仕様、およびスロットの製造元の ID を含めることができます。オプションで、スロットのハードウェアとファームウェアのバージョンを定義できます。ファイルのフォーマットや、個々のキーとそれに割り当てられる値の詳細については、opencryptoki.conf(5) の man ページを参照してください。

ランタイム時に pkcsslotd デーモンの動作を変更するには、pkcsconf ユーティリティーを使用します。このツールでは、デーモンの状態の表示や設定、現在設定されているスロットやトークンの一覧表示や変更を行うことができます。たとえば、トークン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(pkcsslotd デーモンと通信する必要があるすべての非 root ユーザーは、pkcs11 システムグループの一部である必要があることに注意してください)。

```
~]$ pkcsconf -t
```

pkcsconf ツールで利用可能な引数の一覧は、pkcsconf(1) の man ページを参照してください。

警告

このグループのすべてのメンバーには、openCryptoki サービスの他のユーザーが設定済みの PKCS#11 トークンにアクセスできないようにブロックする権利があるため、完全に信頼できるユーザーのみが pkcs11 グループのメンバーシップを割り当てる必要があることに注意してください。このグループのすべてのメンバーは、openCryptoki の他のユーザーの権限で任意のコードを実行することもできます。

4.9.4. OpenSSH に認証情報を提供するスマートカードの使用

スマートカードは、USB メモリー、MicroSD、または SmartCard のフォームファクターを持つ軽量のハードウェアセキュリティーモジュールです。リモートで管理可能なセキュアなキーストアを提供します。Red Hat Enterprise Linux 7 では、OpenSSH はスマートカードを使用した認証をサポートしています。

OpenSSH でスマートカードを使用するには、カードからの公開鍵を ~/.ssh/authorized_keys ファイルに保存します。opensc パッケージが提供する PKCS#11 ライブラリーをクライアントにインストールします。PKCS#11 は、トークンと呼ばれる暗号化デバイスへのアプリケーションプログラミングインターフェイス(API)を定義する公開鍵暗号化標準です。root で以下のコマンドを入力します。

```
~]# yum install opensc
```

4.9.4.1. カードからの公開鍵の取得

カードの鍵を一覧表示するには、ssh-keygen コマンドを使用します。共有ライブラリー (以下の例では OpenSC) を -D ディレクティブで指定します。

```
~]$ ssh-keygen -D /usr/lib64/pkcs11/opensc-pkcs11.so
ssh-rsa AAAAB3NzaC1yc[...]+g4Mb9
```

4.9.4.2. サーバーへの公開鍵の保存

リモートサーバーでスマートカードを使用した認証を有効にするには、公開鍵をリモートサーバーに転送します。これを行うには、取得した文字列（キー）をコピーしてリモートシェルに貼り付けるか、鍵をファイル（以下の例ではsmartcard.pub ）に保存し、ssh-copy-id コマンドを使用します。

```
~]$ ssh-copy-id -f -i smartcard.pub user@hostname
user@hostname's password:
```

```
Number of key(s) added: 1

Now try logging into the machine, with:   "ssh user@hostname"
and check to make sure that only the key(s) you wanted were added.
```

秘密鍵ファイルなしで公開鍵を保存するには、環境変数SSH_COPY_ID_LEGACY=1 または -f オプションを使用する必要があります。

4.9.4.3. スマートカードのキーを使用したサーバーへの認証

OpenSSH は、スマートカードからあなたの公開鍵を読み取り、鍵そのものを公開することなく、あなたの秘密鍵を使った操作を実行することができます。これは、秘密鍵がカードを離れないことを意味します。スマートカードを認証に使用してリモートサーバーに接続するには、次のコマンドを入力し、カードを保護する PIN を入力してください。

```
[localhost ~]$ ssh -I /usr/lib64/pkcs11/opensc-pkcs11.so hostname
Enter PIN for 'Test (UserPIN)':
[hostname ~]$
```

ホスト名 を、接続する実際のホスト名に置き換えます。

次回リモートサーバーに接続する際に不要な入力を保存するには、~/.ssh/config ファイルの PKCS#11 ライブラリーへのパスを保存します。

```
Host hostname
    PKCS11Provider /usr/lib64/pkcs11/opensc-pkcs11.so
```

追加オプションを指定せずに ssh コマンドを実行して接続します。

```
[localhost ~]$ ssh hostname
Enter PIN for 'Test (UserPIN)':
[hostname ~]$
```

4.9.4.4. ssh-agent を使用した PIN ログインの自動化

ssh-agent の使用を開始するための環境変数を設定します。ssh-agent は通常のセッションですでに実行されているため、ほとんどの場合、この手順をスキップできます。以下のコマンドを使用して、認証エージェントに接続できるかどうかを確認します。

```
~]$ ssh-add -l
Could not open a connection to your authentication agent.
~]$ eval `ssh-agent`
```

このキーを使って接続するたびに PIN を書き込まないようにするには、次のコマンドを実行してカードをエージェントに追加してください。

```
~]$ ssh-add -s /usr/lib64/pkcs11/opensc-pkcs11.so
Enter PIN for 'Test (UserPIN)':
Card added: /usr/lib64/pkcs11/opensc-pkcs11.so
```

カードを ssh-agent から削除するには、次のコマンドを使用します。

```
~]$ ssh-add -e /usr/lib64/pkcs11/opensc-pkcs11.so
Card removed: /usr/lib64/pkcs11/opensc-pkcs11.so
```

注記

FIPS 201-2 は、カードに格納されたデジタル署名鍵の使用条件として、Personal Identity Verification (PIV) カード保持者による明示的なユーザーアクションを要求します。OpenSC には、この要件が正しく適用されます。

しかし、アプリケーションによっては、カード保有者が署名のたびに PIN を入力することを要求するのは現実的ではありません。スマートカード PIN をキャッシュするには、/etc/opensc-x86_64.conf の pin_cache_ignore_user_consent = true; オプションの前に # 文字を削除します。

詳細は、Cardholder Authentication for the PIV Digital Signature Key (NISTIR 7863) レポートを参照してください。

4.9.4.5. 関連情報

ハードウェアまたはソフトウェアトークンのセットアップについては、Smart Card support in Red Hat Enterprise Linux 7 の記事で説明されています。

スマートカードや同様の PKCS# 11 セキュリティートークンを管理および使用する pkcs11-tool ユーティリティーの詳細は、pkcs11-tool (1) の man ページを参照してください。

4.9.5. 信頼できる鍵および暗号化された鍵

信頼できる鍵と暗号化された鍵は、カーネルキーリングサービスを使用するカーネルが生成する可変長の対称鍵です。キーが暗号化されていない形式でユーザースペースに表示されないという事実は、キーの整合性を検証できることを意味します。つまり、拡張検証モジュール (EVM) などで実行中のシステムの整合性を検証および確認するために使用できます。ユーザーレベルのプログラムは、暗号化された blobs の形式でのみキーにアクセスできます。

信頼できる鍵は、Trusted Platform Module (TPM) チップというハードウェアが必要になります。これは、鍵を作成し、暗号化 (保護) するために使用されます。TPM は、ストレージルートキー(SK)と呼ばれる 2048 ビットの RSA キーを使用して鍵 を保護します。

それに加えて、信頼できるキーは、TPM の プラットフォーム設定レジスター (PCR) 値の特定のセットを使用してシールすることもできます。PCR には、BIOS、ブートローダー、およびオペレーティングシステムを反映する一連の整合性管理値が含まれています。つまり、PCR でシールされたキーは、暗号化されたのとまったく同じシステム上の TPM でのみ復号化できます。ただし、PCR で保護された信頼できる鍵が読み込まれると (キーリングに追加されると)、新しいカーネルなどを起動できるように、関連する PCR 値が検証され、新しい (または今後の) PCR 値に更新されます。1 つの鍵を複数の blobs として保存することもできますが、それぞれ PCR 値が異なります。

暗号化鍵はカーネルの AES 暗号化を使用するため、TPM は必要ありません。これにより、信頼できる鍵よりも高速になります。暗号化鍵は、カーネルが生成した乱数を使用して作成され、ユーザー空間の blobs へのエクスポート時に マスターキー により暗号化されます。このマスターキーは、信頼できるキーでもユーザーキーでも構いませんが、これが主な欠点となっています。マスターキーが信頼できるキーでない場合、暗号化されたキーは、それを暗号化するために使用したユーザーキーと同等の安全性しか得られません。

4.9.5.1. 鍵を使った作業

鍵を使用して操作を実行する前に、trusted および encrypted-keys カーネルモジュールがシステムに読み込まれていることを確認してください。さまざまな RHEL カーネルアーキテクチャーでカーネルモジュールをロードするときは、次の点を考慮してください。

- x86_64 アーキテクチャーを持つ RHEL カーネルの場合、TRUSTED_KEYS および ENCRYPTED_KEYS コードはコアカーネルコードの一部として組み込まれています。その結果、x86_64 システムユーザーは、trusted モジュールおよび encrypted-keys モジュールを読み込むことなく、これらの鍵を使用できます。

- その他のアーキテクチャーでは、鍵で操作を実行する前に、trusted および encrypted-keys カーネルモジュールを読み込む必要があります。カーネルモジュールをロードするには、以下のコマンドを実行します。

```
~]# modprobe trusted encrypted-keys
```

信頼できる鍵および暗号化された鍵は、keyctl ユーティリティーを使用して作成、読み込み、エクスポート、および更新できます。keyctl の使用に関する詳細は、keyctl(1) を参照してください。

注記

TPM を使用するには (信頼できる鍵の作成および保護目的など)、これを有効かつアクティブにする必要があります。これは通常、マシンの BIOS の設定、またはユーティリティーの tpm-tools パッケージから tpm_setactive コマンドを使用して実行できます。また、TPM と通信するには、TrouSers アプリケーション(trousers パッケージ) と、TrouSers スイートの一部である tcsd デーモンをインストールする必要があります。

TPM を使用して信頼できる鍵を作成するには、次の構文で keyctl コマンドを実行します。

```
~]$ keyctl add trusted name "new keylength [options]" keyring
```

上記の構文を使用すると、コマンドの例を次のように作成できます。

```
~]$ keyctl add trusted kmk "new 32" @u
642500861
```

上記の例では、kmk と呼ばれる信頼できる鍵を作成し、長さが 32 バイト(256 ビット)で、ユーザーのキーリング(@u)に配置します。鍵の長さは 32 から 128 バイト (256 から 1024 ビット) です。show サブコマンドを使用して、カーネルキーリングの現在の構造を一覧表示します。

```
~]$ keyctl show
Session Keyring
       -3 --alswrv    500   500  keyring: _ses
 97833714 --alswrv    500    -1   \_ keyring: _uid.1000
642500861 --alswrv    500   500       \_ trusted: kmk
```

print サブコマンドは、暗号化されたキーを標準出力に出力します。キーをユーザー空間のブロブにエクスポートするには、以下のように pipe サブコマンドを使用します。

```
~]$ keyctl pipe 642500861 > kmk.blob
```

ユーザー空間のブロブから信頼できる鍵を読み込むには、ブロブを引数として add コマンドを再度

使用します。

```
~]$ keyctl add trusted kmk "load `cat kmk.blob`" @u
268728824
```

次に、TPM でシールされた信頼できる鍵を使用して、安全な暗号化された鍵を作成できます。暗号化された鍵の生成には、以下のコマンド構文を使用します。

```
~]$ keyctl add encrypted name "new [format] key-type:master-key-name keylength" keyring
```

上記の構文に基づいて、すでに作成済みの信頼できるキーを使用して暗号化されたキーを生成するコマンドは、次のように設定できます。

```
~]$ keyctl add encrypted encr-key "new trusted:kmk 32" @u
159771175
```

TPM が使用できないシステムで暗号化されたキーを作成するには、ランダムな数字のシーケンスを使用してユーザーキーを生成し、次にこれを使用して実際の暗号化されたキーをシールします。

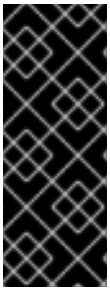
```
~]$ keyctl add user kmk-user "`dd if=/dev/urandom bs=1 count=32 2>/dev/null`" @u
427069434
```

次に、乱数ユーザーキーを使用して暗号化されたキーを生成します。

```
~]$ keyctl add encrypted encr-key "new user:kmk-user 32" @u
1012412758
```

list サブコマンドは、指定されたカーネルキーリング内のすべてのキーを一覧表示するために使用できます。

```
~]$ keyctl list @u
2 keys in keyring:
427069434: --alswrv  1000  1000 user: kmk-user
1012412758: --alswrv  1000  1000 encrypted: encr-key
```



重要

暗号化鍵がマスターの信頼できる鍵で保護されていない場合には、そのセキュリティーは、ユーザーのマスターキー (乱数キー) を使用して暗号化したものと同じでしかない点に注意してください。そのため、マスターユーザーキーはできるだけ安全に、システムの起動プロセスの早い段階で読み込む必要があります。

4.9.5.2. 関連情報

次のオフラインおよびオンラインリソースを使用して、信頼できる暗号化されたキーの使用に関する追加情報を取得できます。

インストールされているドキュメント

- *keyctl(1) : keyctl ユーティリティーとそのサブコマンドの使用を説明します。*

オンラインドキュメント

- *Red Hat Enterprise Linux 7 SELinux ユーザーおよび管理者のガイド: Red Hat Enterprise Linux 7 『の SELinux ユーザーおよび管理者のガイド』 では、SELinux の原則と、SELinux を Apache HTTP Server などのさまざまなサービスで設定および使用する方法を説明します。*

- *https://www.kernel.org/doc/Documentation/security/keys-trusted-encrypted.txt — Linux カーネルの信頼できる鍵および暗号化された鍵機能に関する公式ドキュメントです。*

関連項目

- *「高度暗号化標準 — AES」 Advanced Encryption Standard の簡潔な説明を提供します。*

- *「公開鍵の暗号化」 は、公開鍵暗号化アプローチとそれが使用するさまざまな暗号化プロトコルについて説明します。*

4.9.6. 乱数ジェネレーターの使用

簡単に破ることができない安全な暗号化キーを生成できるようにするには、乱数のソースが必要です。一般に、番号がランダムであるほど、一意のキーを取得する可能性が高くなります。乱数を生成するための エントロピー は、通常、環境の "ノイズ" を計算するか、ハードウェア 乱数ジェネレーター を使用して取得されます。

rng-tools パッケージの一部である rngd デーモンは、エントロピーを抽出するために環境ノイズとハードウェア乱数ジェネレーターの両方を使用できます。デーモンは、乱数性のソースによって供給されるデータが十分にランダムなものかどうかをチェックしてから、カーネルの乱数エントロピープールにデータを格納します。生成する乱数は、/dev/random および /dev/urandom キャラクターデバイスで利用できます。

/dev/random と /dev/urandom の違いは、前者はブロッキングデバイスであることです。つまり、エントロピーの量が適切なランダム出力を生成するのに不十分なと判断すると、番号の供給を停止します。逆に、/dev/urandom はノンブロッキングソースであり、カーネルのエントロピープールを再利用するため、エントロピーが少ない疑似乱数を無制限に提供できます。そのため、/dev/urandom は、長期暗号化キーの作成には使用しないでください。

rng-tools パッケージをインストールするには、root で以下のコマンドを発行します。

```
~]# yum install rng-tools
```

rngd デーモンを起動するには、root で以下のコマンドを実行します。

```
~]# systemctl start rngd
```

デーモンの状態を問い合わせるには、次のコマンドを使用します。

```
~]# systemctl status rngd
```

オプションのパラメーターを使用して rngd デーモンを起動するには、直接実行します。たとえば、乱数入力の代替ソース(/dev/hwrandom以外)を指定するには、以下のコマンドを使用します。

```
~]# rngd --rng-device=/dev/hwrng
```

上記のコマンドは、乱数が読み取られるデバイスとして、/dev/hwrng で rngd デーモンを起動します。同様に、-o （または --random-device）オプションを使用して、乱数出力用のカーネルデバイスを選択できます（デフォルトの /dev/random以外）。利用可能なすべてのオプションの一覧は、rngd(8) の man ページを参照してください。

特定のシステムで利用可能なエントロピーのソースを確認するには、root で以下のコマンドを実行します。

```
~]# rngd -vf
```

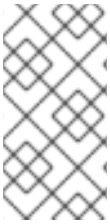
```
Unable to open file: /dev/tpm0
Available entropy sources:
 DRNG
```

注記

rngd -v コマンドを入力すると、以下のプロセスがバックグラウンドで実行を継続します。-b, --background オプション (デーモンになる) はデフォルトで適用されます。

TPM デバイスが存在しない場合は、エントロピーのソースとして Intel Digital Random Number Generator (DRNG) のみが表示されます。お使いの CPU が RDRAND プロセッサー命令をサポートしているかどうかを確認するには、以下のコマンドを入力します。

```
~]$ cat /proc/cpuinfo | grep rdrand
```



注記

詳細およびソフトウェアコードの例については、*Intel Digital Random Number Generator (DRNG) Software Implementation Guide.* を参照してください。

rng-tools パッケージには、データのランダム性を確認するために使用できる rngtest ユーティリティーも含まれています。/dev/random の出力のランダム性のレベルをテストするには、次のように rngtest ツールを使用します。

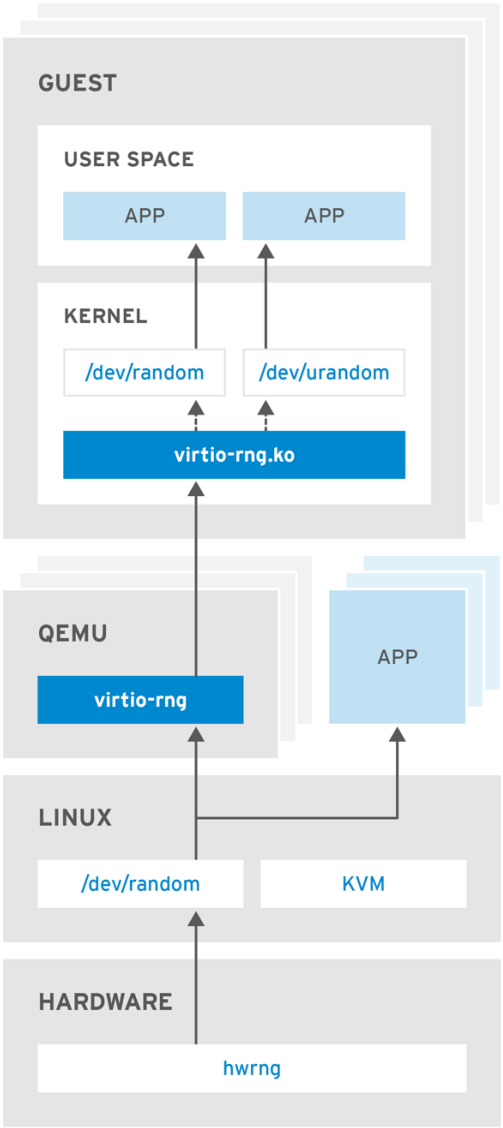
```
~]$ cat /dev/random | rngtest -c 1000
rngtest 5
Copyright (c) 2004 by Henrique de Moraes Holschuh
This is free software; see the source for copying conditions. There is NO warranty; not even for
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

rngtest: starting FIPS tests...
rngtest: bits received from input: 20000032
rngtest: FIPS 140-2 successes: 998
rngtest: FIPS 140-2 failures: 2
rngtest: FIPS 140-2(2001-10-10) Monobit: 0
rngtest: FIPS 140-2(2001-10-10) Poker: 0
rngtest: FIPS 140-2(2001-10-10) Runs: 0
rngtest: FIPS 140-2(2001-10-10) Long run: 2
rngtest: FIPS 140-2(2001-10-10) Continuous run: 0
rngtest: input channel speed: (min=1.171; avg=8.453; max=11.374)Mibits/s
rngtest: FIPS tests speed: (min=15.545; avg=143.126; max=157.632)Mibits/s
rngtest: Program run time: 2390520 microseconds
```

rngtest ツールの出力に表示される多数の障害は、テストされたデータのランダム性が不十分であ

り、依存しないことを示しています。rngtest ユーティリティーで利用可能なオプションの一覧は、rngtest(1) の man ページを参照してください。

Red Hat Enterprise Linux 7 では、ホストマシンからエントロピーにアクセスできる KVM 仮想マシンを提供する virtio RNG （乱数ジェネレーター）デバイスが導入されました。推奨される設定では、hwrng はホスト Linux カーネルのエントロピープールに(/dev/randomを介して)フィードし、QEMU はゲストが要求するエントロピーのソースとして /dev/random を使用します。

図4.1 virtio RNG デバイス



RHEL_453350_0717

[D]

以前は、Red Hat Enterprise Linux 7.0 ゲストおよび Red Hat Enterprise Linux 6 ゲストは、rngd ユーザースペースデーモンを介してホストからのエントロピーを利用できました。デーモンの設定は、Red Hat Enterprise Linux のインストールごとに手作業で行っていました。Red Hat Enterprise Linux 7.1 では、手動の手順が不要になり、プロセス全体がシームレスかつ自動化されています。rngd を使用する必要がなくなり、使用可能なエントロピーが特定のしきい値を下回ると、ゲストカーネル自体がホ

ストからエントロピーをフェッチします。これにより、ゲストカーネルは、アプリケーションが要求するとすぐに乱数を使用できるようにします。

Red Hat Enterprise Linux インストーラーである Anaconda は、インストーラーイメージに virtio-rng モジュールを提供し、Red Hat Enterprise Linux のインストール時にホストエントロピーを利用できるようにするようになりました。

重要

シナリオで使用する乱数ジェネレーターを正しく決定するには、Understanding the Red Hat Enterprise Linux random number generator interface の記事を参照してください。

4.10. ポリシーベースの復号を使用して暗号化ボリュームの自動アンロックの設定

Policy-Based Decryption (PBD) は、ユーザーパスワード、Trusted Platform Module (TPM) デバイス、スマートカードなどのシステムに接続された PKCS#11 デバイス、または特殊なネットワークサーバーなどを使用した異なる方法を用いて、物理および仮想マシンのハードドライブの暗号化された root ボリュームとセカンダリーボリュームを解除できるテクノロジーのコレクションです。

テクノロジーとしての PBD を使用すると、さまざまなロック解除方法をポリシーに組み合わせて、同じボリュームをさまざまな方法でロック解除する機能を作成できます。現在、Red Hat Enterprise Linux に実装されている PBD は、Clevis フレームワークと pins と呼ばれるプラグインで設定されています。各ピンは、個別のアンロック機能を提供します。現在のところ、TPM によるボリュームのロック解除とネットワークサーバーによるロック解除の 2 つのピンしか用意されていません。

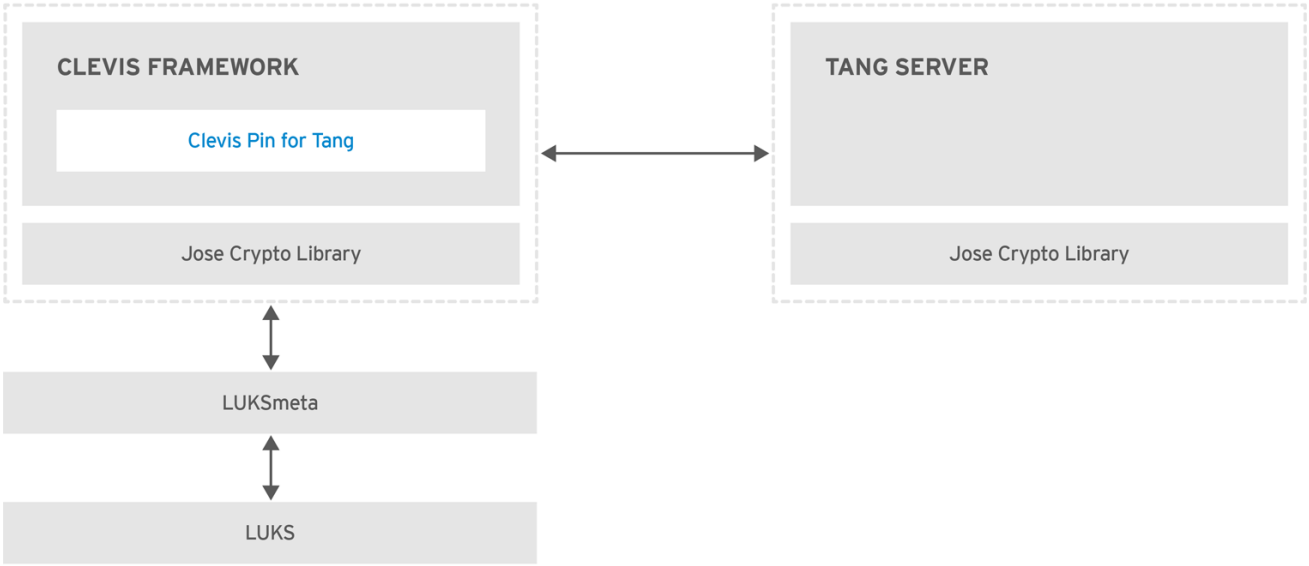
NBDE (Network Bound Disc Encryption) は、特定のネットワークサーバーに暗号化ボリュームをバインドできるようにする PBD テクノロジーのサブカテゴリーです。NBDE の現在の実装には、Tang サーバー用の Clevis ピンと Tang サーバー自体が含まれています。

4.10.1. NBDE (Network-Bound Disk Encryption)

Network Bound Disk Encryption (NBDE) を使用すると、システムの再起動時にパスワードを手動で入力することなく、物理マシンおよび仮想マシンのハードドライブの root ボリュームを暗号化できます。

Red Hat Enterprise Linux 7 では、NBDE は以下のコンポーネントおよび技術により実装されます。

図4.2 Clevis と Tang を使用した Network-Bound Disk Encryption



Tang は、ネットワークのプレゼンスにデータをバインドするためのサーバーです。セキュリティーが保護された特定のネットワークにシステムをバインドする際に利用可能なデータを含めるようにします。Tang はステートレスで、TLS または認証は必要ありません。エスクローベースのソリューション (サーバーが暗号鍵をすべて保存し、使用されたことがあるすべての鍵に関する知識を有する) とは異なり、Tang はクライアントの鍵と相互作用することはないため、クライアントから識別情報を得ることがありません。

Clevis は、自動化された復号用のプラグイン可能なフレームワークです。NBDE では、Clevis は、LUKS ボリュームの自動アンロックを提供します。clevis パッケージは、クライアントで使用される機能を提供します。

Clevis ピン は、Clevis フレームワークへのプラグインです。このようなピンの 1 つは、NBDE サーバー (Tang) との相互作用を実装するプラグインです。

Clevis および Tang は、一般的なクライアントおよびサーバーのコンポーネントで、ネットワークがバインドされた暗号化を提供します。Clevis および Tang は、Red Hat Enterprise Linux 7 で LUKS と組み合わせて、root および非 root のストレージボリュームの暗号化および複号に使用し、NBDE を実現します。

クライアントおよびサーバーのコンポーネントはともに José ライブラリーを使用して、暗号化および複号の操作を実行します。

NBDE のプロビジョニングを開始すると、Tang サーバーの Clevis ピンは、Tang サーバーの、アドバタイズされている非対称鍵の一覧を取得します。もしくは、鍵が非対称であるため、Tang の公開鍵

の一覧を帯域外に配布して、クライアントが Tang サーバーにアクセスしなくても動作できるようにします。このモードは オフラインプロビジョニング と呼ばれます。

Tang 用の Clevis ピンは、公開鍵のいずれかを使用して、固有で、暗号論的に強力な暗号鍵を生成します。この鍵を使用してデータを暗号化すると、この鍵は破棄されます。Clevis クライアントは、使いやすい場所に、このプロビジョニング操作で生成した状態を保存する必要があります。データを暗号化するこのプロセスは プロビジョニング手順 と呼ばれています。NBDE のプロビジョニング状態は、luksmeta パッケージを活用して LUKS ヘッダーに格納されます。

クライアントがそのデータにアクセスする準備ができると、プロビジョニング手順で生成したメタデータを読み込み、応答して暗号鍵を戻します。このプロセスは 復旧手順 と呼ばれます。

Clevis は、NBDE ではピンを使用して LUKS ボリュームをバインドしているため、自動的にロックが解除されます。バインドプロセスが正常に終了すると、提供されている Dracut アンロックを使用してディスクをアンロックできます。

ネットワーク接続が確立される前に起動する必要があるファイルシステムを含む /tmp ディレクトリー、/var ディレクトリー、および /usr/local/ ディレクトリーなどのすべての LUKS 暗号化デバイスは、root ボリューム とみなされます。さらに、/var/log/、var/log/audit/、または /opt など、ネットワークが起動する前に実行されるサービスが使用するすべてのマウントポイントは、ルートデバイスに切り替えた後に早期にマウントする必要があります。/etc/fstab ファイルに _netdev オプションがないため、root ボリュームを識別することもできます。

4.10.2. 暗号化クライアント (Clevis) のインストール

Clevis のプラグ可能なフレームワークとピンを暗号化されたボリューム（クライアント）を備えたマシンにインストールするには、root で以下のコマンドを入力します。

```
~]# yum install clevis
```

データを復号するには、clevis decrypt コマンドを使用して、暗号テキスト(JWE)を提供します。

```
~]$ clevis decrypt < JWE > PLAINTEXT
```

詳細は、CLI ヘルプを参照してください。

```
~]$ clevis
Usage: clevis COMMAND [OPTIONS]

  clevis decrypt      Decrypts using the policy defined at encryption time
```

```
  clevis encrypt http Encrypts using a REST HTTP escrow server policy
  clevis encrypt sss  Encrypts using a Shamir's Secret Sharing policy
  clevis encrypt tang Encrypts using a Tang binding server policy
  clevis encrypt tpm2 Encrypts using a TPM2.0 chip binding policy

~]$ clevis decrypt
Usage: clevis decrypt < JWE > PLAINTEXT

Decrypts using the policy defined at encryption time

~]$ clevis encrypt tang
Usage: clevis encrypt tang CONFIG < PLAINTEXT > JWE

Encrypts using a Tang binding server policy

This command uses the following configuration properties:

  url: <string>   The base URL of the Tang server (REQUIRED)

  thp: <string>   The thumbprint of a trusted signing key

  adv: <string>   A filename containing a trusted advertisement
  adv: <object>   A trusted advertisement (raw JSON)

Obtaining the thumbprint of a trusted signing key is easy. If you
have access to the Tang server's database directory, simply do:

    $ jose jwk thp -i $DBDIR/$SIG.jwk

Alternatively, if you have certainty that your network connection
is not compromised (not likely), you can download the advertisement
yourself using:

    $ curl -f $URL/adv > adv.jws
```

4.10.3. SELinux を Enforcing モードで有効にした Tang サーバーのデプロイメント

Red Hat Enterprise Linux 7.7 以降では、tangd_port_t SELinux タイプが提供され、Tang サーバーは、SELinux Enforcing モードで制限のあるサービスとしてデプロイできます。

前提条件

- **policycoreutils-python-utils パッケージおよび依存関係がインストールされている。**

手順

1. **tang パッケージとその依存関係をインストールするには、root で以下のコマンドを実行します。**

```
~]# yum install tang
```

2. **7500/tcp などの不要なポートを選択し、tangd サービスがそのポートにバインドできるようにします。**

```
~]# semanage port -a -t tangd_port_t -p tcp 7500
```

ポートは一度に 1 つのサービスでのみ使用できるため、すでに使用しているポートを使用しようとすると、ValueError: Port already defined というエラーメッセージが表示されます。

3. **ファイアウォールのポートを開きます。**

```
~]# firewall-cmd --add-port=7500/tcp
~]# firewall-cmd --runtime-to-permanent
```

4. **systemd を使用して tangd サービスを有効にします。**

```
~]# systemctl enable tangd.socket
Created symlink from /etc/systemd/system/multi-user.target.wants/tangd.socket to
/usr/lib/systemd/system/tangd.socket.
```

5. **オーバーライドファイルを作成します。**

```
~]# systemctl edit tangd.socket
```

6. **以下のエディター画面で、/etc/systemd/system/tangd.socket.d/ ディレクトリーにある空の override.conf ファイルを開き、以下の行を追加して Tang サーバーのデフォルトポートを、80 から以前に取得した番号に変更します。**

```
[Socket]
ListenStream=
ListenStream=7500
```

ファイルを保存して、エディターを終了します。

7. **変更した設定を再読み込みし、tangd サービスを起動します。**

```
~]# systemctl daemon-reload
```

8. 設定が機能していることを確認します。

```
~]# systemctl show tangd.socket -p Listen
Listen=[::]:7500 (Stream)
```

9. tangd サービスを起動します。

```
~]# systemctl start tangd.socket
```

tangd は systemd ソケットアクティベーションメカニズムを使用するため、サーバーは最初の接続を受け取るとすぐに起動します。最初の起動時に、一組の暗号鍵が自動的に生成されます。

キーの手動生成などの暗号化操作を実行するには、jose ユーティリティーを使用します。詳細は、jose -h コマンドを入力するか、man ページの jose (1) を参照してください。

例4.4 Tang 鍵のローテーション

鍵を定期的に変更することが重要です。鍵を変更するのに適した間隔は、アプリケーション、鍵サイズ、および組織のポリシーにより異なります。一般的な推奨事項は Cryptographic Key Length Recommendation ページを参照してください。

キーをローテーションするには、キーデータベースディレクトリー（通常は /var/db/tang ）の新しいキーの生成から開始します。たとえば、以下のコマンドを使用して、新しい署名を作成し、鍵を交換します。

```
~]# DB=/var/db/tang
~]# jose jwk gen -i '{"alg":"ES512"}' -o $DB/new_sig.jwk
~]# jose jwk gen -i '{"alg":"ECMR"}' -o $DB/new_exc.jwk
```

古いキーの名前を先頭に . に変更して、アドバタイズメントから非表示にします。以下の例のファイル名は、鍵データベースディレクトリーに実在する固有のファイル名とは異なります。

```
~]# mv $DB/old_sig.jwk $DB/.old_sig.jwk
~]# mv $DB/old_exc.jwk $DB/.old_exc.jwk
```

Tang は、直ちにすべての変更を適用します。再起動は必要ありません。

この時点で、新しいクライアントバインディングは新しい鍵を選択し、以前のクライアントは古い鍵を使用し続けます。すべてのクライアントが新しい鍵を使用することを確認すると、古い鍵を削除できます。

警告

クライアントが使用している最中に古い鍵を削除すると、データが失われる場合があります。

4.10.3.1. 高可用性システムのデプロイメント

Tang は、高可用性デプロイメントを構築する方法を 2 つ提供します。

1. クライアントの冗長性 (推奨)

   クライアントは、複数の Tang サーバーにバインドする機能を使用して設定する必要があります。この設定では、各 Tang サーバーは独自の鍵を持ち、クライアントはこれらのサーバーのサブセットに連絡することで復号化できます。Clevis は、sss プラグインを使用してこのワークフローをすでにサポートしています。

   この設定の詳細については、以下の man ページを参照してください。

   - tang (8)、High Availability セクション
   - clevis (1) の Shamir's Secret Sharing セクション
   - clevis-encrypt-sss(1)

   Red Hat は、高可用性のデプロイメントにこの方法を推奨します。

2. 鍵の共有

冗長性を確保するために、Tang のインスタンスは複数デプロイできます。2 番目以降のインスタンスを設定するには、tang パッケージをインストールし、SSH 経由で rsync を使用して鍵ディレクトリーを新規ホストにコピーします。鍵を共有すると鍵への不正アクセスのリスクが高まり、追加の自動化インフラストラクチャーが必要になるため、Red Hat はこの方法を推奨していません。

4.10.4. Tang を使用する NBDE システムへの暗号化クライアントのデプロイメント

前提条件

- Clevis フレームワークがインストールされている。「暗号化クライアント (Clevis) のインストール」 を参照
- Tang サーバーまたはそのダウンロードされた広告が利用可能です。「SELinux を Enforcing モードで有効にした Tang サーバーのデプロイメント」 を参照

手順

Clevis 暗号化クライアントを Tang サーバーにバインドするには、clevis encrypt tang サブコマンドを使用します。

```
~]$ clevis encrypt tang '{"url":"http://tang.srv"}' < PLAINTEXT > JWE
The advertisement contains the following signing keys:

_OsIk0T-E2l6qjfdDiwVmidoZjA

Do you wish to trust these keys? [ynYN] y
```

先ほどの例の http://tang.srv の URL を、tang がインストールされているサーバーの URL と一致するように変更します。JWE 出力ファイルには、暗号化された暗号文が含まれます。この暗号文は、PLAINTEXT 入力ファイルから読み込まれます。

データを復号するには、clevis decrypt コマンドを使用して、暗号テキスト(JWE)を提供します。

```
~]$ clevis decrypt < JWE > PLAINTEXT
```

詳細は、man ページの clevis-encrypt-tang (1) を参照するか、同梱の CLI ヘルプを使用します。

```
~]$ clevis
Usage: clevis COMMAND [OPTIONS]

  clevis decrypt      Decrypts using the policy defined at encryption time
  clevis encrypt http Encrypts using a REST HTTP escrow server policy
  clevis encrypt sss  Encrypts using a Shamir's Secret Sharing policy
  clevis encrypt tang Encrypts using a Tang binding server policy
  clevis luks bind    Binds a LUKSv1 device using the specified policy
  clevis luks unlock  Unlocks a LUKSv1 volume

~]$ clevis decrypt
Usage: clevis decrypt < JWE > PLAINTEXT

Decrypts using the policy defined at encryption time

~]$ clevis encrypt tang
Usage: clevis encrypt tang CONFIG < PLAINTEXT > JWE

Encrypts using a Tang binding server policy

This command uses the following configuration properties:

  url: <string>   The base URL of the Tang server (REQUIRED)

  thp: <string>   The thumbprint of a trusted signing key

  adv: <string>   A filename containing a trusted advertisement
  adv: <object>   A trusted advertisement (raw JSON)

Obtaining the thumbprint of a trusted signing key is easy. If you
have access to the Tang server's database directory, simply do:

    $ jose jwk thp -i $DBDIR/$SIG.jwk

Alternatively, if you have certainty that your network connection
is not compromised (not likely), you can download the advertisement
yourself using:

    $ curl -f $URL/adv > adv.jws
```

4.10.5. TPM 2.0 ポリシーを使用した暗号化クライアントのデプロイメント

64 ビット Intel または 64 ビットの AMD アーキテクチャーのシステムでは、Trusted Platform Module 2.0 (TPM 2.0)チップを使用して暗号化するクライアントをデプロイするには、JSON 設定オブジェクトの形式で唯一の引数を指定して clevis encrypt tpm2 サブコマンドを使用します。

```
~]$ clevis encrypt tpm2 '{}' < PLAINTEXT > JWE
```

別の階層、ハッシュ、および鍵アルゴリズムを選択するには、以下のように、設定プロパティーを指定します。

-

```
~]$ clevis encrypt tpm2 '{"hash":"sha1","key":"rsa"}' < PLAINTEXT > JWE
```

データを復号するには、暗号文 (JWE) を提供します。

```
~]$ clevis decrypt < JWE > PLAINTEXT
```

ピンは、PCR (Platform Configuration Registers) 状態へのデータのシーリングにも対応します。このように、PCP ハッシュ値が、シーリング時に使用したポリシーと一致する場合にのみ、データのシーリングを解除できます。

たとえば、SHA1 バンクに対して、インデックス 0 および 1 の PCR にデータをシールするには、以下を行います。

```
~]$ clevis encrypt tpm2 '{"pcr_bank":"sha1","pcr_ids":"0,1"}' < PLAINTEXT > JWE
```

詳細と可能な設定プロパティーの一覧は、man ページの clevis-encrypt-tpm2 (1) を参照してください。

4.10.6. root ボリュームの手動登録の設定

LUKS で暗号化した既存の root ボリュームを自動的にアンロックするには、clevis-luks サブパッケージをインストールし、clevis luks bind コマンドを使用してボリュームを Tang サーバーにバインドします。

```
~]# yum install clevis-luks
```

```
~]# clevis luks bind -d /dev/sda tang '{"url":"http://tang.srv"}'
The advertisement contains the following signing keys:

_OsIk0T-E2l6qjfdDiwVmidoZjA

Do you wish to trust these keys? [ynYN] y
You are about to initialize a LUKS device for metadata storage.
Attempting to initialize it may result in data loss if data was
already written into the LUKS header gap in a different format.
A backup is advised before initialization is performed.

Do you wish to initialize /dev/sda? [yn] y
Enter existing LUKS password:
```

このコマンドは、以下の 4 つの手順を実行します。

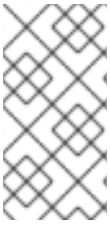
1. *LUKS マスター鍵と同じエントロピーを使用して、新しい鍵を作成します。*

2. *Clevis で新しい鍵を暗号化します。*

3. *LUKS ヘッダーに Clevis JWE オブジェクトを LUKSMeta で格納します。*

4. *LUKS を使用する新しい鍵を有効にします。*

このディスクは、Clevis ポリシーだけでなく、既存のパスワードでもロック解除ができるようになりました。詳細は、man ページの clevis-luks-bind (1) を参照してください。



注記

バインド手順では、空き LUKS パスワードスロットが少なくとも 1 つあることが前提となっています。clevis luks bind コマンドは、スロットの 1 つを取ります。

Clevis JWE オブジェクトが LUKS ヘッダーに正常に配置されていることを確認するには、luksmeta show コマンドを使用します。

```
~]# luksmeta show -d /dev/sda
0   active empty
1   active cb6e8904-81ff-40da-a84a-07ab9ab5715e
2 inactive empty
3 inactive empty
4 inactive empty
5 inactive empty
6 inactive empty
7 inactive empty
```

システムの起動プロセスの初期段階でディスクバインディングを処理するようにするには、インストール済みのシステムで次のコマンドを実行します。

```
~]# yum install clevis-dracut
~]# dracut -f --regenerate-all
```

重要

(DHCP を使用しない) 静的な IP 設定を持つクライアントに NBDE を使用するには、以下のように、手動でネットワーク設定を dracut ツールに渡します。

```
~]# dracut -f --regenerate-all --kernel-cmdline "ip=192.0.2.10 netmask=255.255.255.0
gateway=192.0.2.1 nameserver=192.0.2.45"
```

または、静的ネットワーク情報を使用して、/etc/dracut.conf.d/ ディレクトリーに .conf ファイルを作成します。以下に例を示します。

```
~]# cat /etc/dracut.conf.d/static_ip.conf
kernel_cmdline="ip=10.0.0.103 netmask=255.255.252.0 gateway=10.0.0.1
nameserver=10.0.0.1"
```

初期 RAM ディスクイメージを再生成します。

```
~]# dracut -f --regenerate-all
```

詳細は、man ページの dracut.cmdline (7) を参照してください。

4.10.7. キックスタートを使用した自動登録の設定

Clevis は、キックスタートと統合して、登録プロセスを完全に自動化にできます。

1.

一時パスワードを使用して、/boot 以外のすべてのマウントポイントで LUKS 暗号化が有効になっているディスクを分割するように、キックスタートに指示します。パスワードは、登録プロセスの手順に使用するための一時的なものです。

```
part /boot --fstype="xfs" --ondisk=vda --size=256
part / --fstype="xfs" --ondisk=vda --grow --encrypted --passphrase=temppass
```

OSPP 準拠のシステムには、より複雑な設定が必要であることに注意してください。次に例を示します。

```
part /boot --fstype="xfs" --ondisk=vda --size=256
part / --fstype="xfs" --ondisk=vda --size=2048 --encrypted --passphrase=temppass
part /var --fstype="xfs" --ondisk=vda --size=1024 --encrypted --passphrase=temppass
part /tmp --fstype="xfs" --ondisk=vda --size=1024 --encrypted --passphrase=temppass
part /home --fstype="xfs" --ondisk=vda --size=2048 --grow --encrypted --
```

```
passphrase=temppass
part /var/log --fstype="xfs" --ondisk=vda --size=1024 --encrypted --passphrase=temppass
part /var/log/audit --fstype="xfs" --ondisk=vda --size=1024 --encrypted --
passphrase=temppass
```

2. **関連する Clevis パッケージを %packages セクションに追加してインストールします。**

```
%packages
clevis-dracut
%end
```

3. **clevis luks bind を呼び出して、%post セクションでバインディングを実行します。その後、一時パスワードを削除します。**

```
%post
clevis luks bind -f -k- -d /dev/vda2 \
tang '{"url":"http://tang.srv","thp":"_OsIk0T-E2l6qjfdDiwVmidoZjA"}' \ <<< "temppass"
cryptsetup luksRemoveKey /dev/vda2 <<< "temppass"
%end
```

上記の例では、バインディングの設定で、Tang サーバーで信頼するサムプリントを指定することで、バインディングを完全に非対話にします。

Tang サーバーの代わりに TPM 2.0 ポリシーを使用する場合は、同様の手順を使用できます。

キックスタートのインストールに関する詳細は、*Red Hat Enterprise Linux 7 Installation Guide* を参照してください。Linux Unified Key Setup-on-disk-format (LUKS) の詳細は、*「LUKS ディスクの暗号化の使用」* を参照してください。

4.10.8. リムーバブルストレージデバイスの自動ロック解除の設定

USB ドライブなど、LUKS で暗号化したリムーバブルストレージデバイスを自動的にアンロックするには、clevis-udisks2 パッケージをインストールします。

```
~]# yum install clevis-udisks2
```

システムを再起動してから、*「root ボリュームの手動登録の設定」* の説明に従って、clevis luks bind コマンドを使用したバインディング手順を実行します。以下に例を示します。

```
~]# clevis luks bind -d /dev/sdb1 tang '{"url":"http://tang.srv"}'
```

LUKS で暗号化したリムーバブルデバイスは、GNOME デスクトップセッションで自動的にアンロックできるようになりました。Clevis ポリシーにバインドされているデバイスは、clevis luks unlock コマンドでロックを解除することもできます。

```
~]# clevis luks unlock -d /dev/sdb1
```

Tang サーバーの代わりに TPM 2.0 ポリシーを使用する場合は、同様の手順を使用できます。

4.10.9. ブート時に非 root ボリュームのロックを自動解除する設定

NBDE を使用して、LUKS で暗号化した非 root ボリュームをアンロックするには、以下の手順を行います。

1. clevis-systemd パッケージをインストールします。

```
~]# yum install clevis-systemd
```

2. Clevis のアンロックサービスを有効にします。

```
~]# systemctl enable clevis-luks-askpass.path
Created symlink from /etc/systemd/system/remote-fs.target.wants/clevis-luks-askpass.path to
/usr/lib/systemd/system/clevis-luks-askpass.path.
```

3. 「root ボリュームの手動登録の設定」 の説明に従って、clevis luks bind コマンドを使用したバインディング手順を実行します。

4. システムの起動時に暗号化されたブロックデバイスを設定するには、_netdev オプションを指定して対応する行を /etc/crypttab 設定ファイルに追加します。詳細は、crypttab (5) man ページを参照してください。

5. /etc/fstab ファイルでアクセス可能なファイルシステムの一覧にボリュームを追加します。この設定ファイルに _netdev オプションを使用します。詳細は、man ページの fstab (5) を参照してください。

4.10.10. NBDE ネットワークでの仮想マシンのデプロイ

clevis luks bind コマンドは、LUKS マスターキーを変更しません。これは、仮想マシンまたはクラウド環境で使用する、LUKS で暗号化したイメージを作成する場合に、このイメージを実行するすべてのインスタンスがマスター鍵を共有することを意味します。これにはセキュリティーの観点で大きな問題があるため、常に回避する必要があります。

これは、Clevis の制限ではなく、LUKS の設計原理です。クラウドに暗号化された root ボリュームが必要な場合は、クラウドの Red Hat Enterprise Linux の各インスタンスにインストールプロセスを実行できるようにする (通常はキックスタートを使用) 必要があります。このイメージは、LUKS マスター鍵を共有しなければ共有できません。

仮想化環境に自動アンロックをデプロイする場合は、キックスタートファイルを使用して lorax、virt-install などのシステムを使用すること (「キックスタートを使用した自動登録の設定」 を参照)、または暗号化した各仮想マシンに固有のマスター鍵があるようにする別の自動プロビジョニングツールを使用することを Red Hat は強く推奨します。

4.10.11. NBDE を使用してクラウド環境に自動的に登録可能な仮想マシンイメージの構築

自動登録可能な暗号化イメージをクラウド環境にデプロイすると、特有の課題が発生する可能性があります。他の仮想化環境と同様に、LUKS マスター鍵を共有しないように、1 つのイメージから起動するインスタンス数を減らすことが推奨されます。

したがって、ベストプラクティスは、どのパブリックリポジトリーでも共有されず、限られたインスタンスのデプロイメントのベースを提供するように、イメージをカスタマイズすることです。作成するインスタンスの数は、デプロイメントのセキュリティーポリシーで定義する必要があります。また、LUKS マスター鍵の攻撃ベクトルに関連するリスク許容度に基づいて決定する必要があります。

LUKS に対応する自動デプロイメントを構築するには、Lorax、virt-install などのシステムとキックスタートファイルを一緒に使用し、イメージ構築プロセス中にマスター鍵の一意性を確保する必要があります。

クラウド環境では、ここで検討する 2 つの Tang サーバーデプロイメントオプションが利用できます。まず、クラウド環境そのものに Tang サーバーをデプロイできます。もしくは、2 つのインフラストラクチャー間で VPN リンクを使用した独立したインフラストラクチャーで、クラウドの外に Tang サーバーをデプロイできます。

クラウドに Tang をネイティブにデプロイすると、簡単にデプロイできます。ただし、別のシステムの暗号文のデータ永続化層でインフラストラクチャーを共有します。Tang サーバーの秘密鍵および Clevis メタデータは、同じ物理ディスクに保存できる場合があります。この物理ディスクでは、暗号文データへのいかなる不正アクセスが可能になります。

重要

このため、Red Hat は、データを保存する場所と、Tang が実行しているシステムを、物理的に分離させることを強く推奨します。クラウドと Tang サーバーを分離することで、Tang サーバーの秘密鍵が、Clevis メタデータと誤って結合することがないようにします。さらに、これにより、クラウドインフラストラクチャーが危険にさらされている場合に、Tang サーバーのローカル制御を提供します。

4.10.12. 関連情報

ナレッジベースの記事How to set up Network Bound Disk Encryption with multiple LUKS devices (Clevis+Tang unlocking)

詳細は、以下の man ページを参照してください。

- tang(8)
- clevis(1)
- jose(1)
- clevis-luks-unlockers(1)
- tang-nagios(1)

4.11. AIDEで整合性の確認

AIDE(Advanced Intrusion Detection Environment)は、システム上でファイルのデータベースを作成し、そのデータベースを使用してファイルの整合性を確保し、システムの侵入を検出するユーティリティーです。

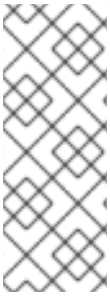
4.11.1. AIDEのインストール

aide パッケージをインストールするには、root で以下のコマンドを入力します。

```
~]# yum install aide
```

初期データベースを生成するには、root で以下のコマンドを入力します。

```
~]# aide --init

AIDE, version 0.15.1

### AIDE database at /var/lib/aide/aide.db.new.gz initialized.
```



注記

デフォルト設定では、aide --init コマンドは、/etc/aide.conf ファイルで定義されているディレクトリーとファイルのセットのみをチェックします。AIDE データベースに追加のディレクトリーまたはファイルを含め、監視パラメーターを変更するには、それに応じて /etc/aide.conf を編集します。

データベースの使用を開始するには、初期データベースファイル名から .new 部分文字列を削除します。

```
~]# mv /var/lib/aide/aide.db.new.gz /var/lib/aide/aide.db.gz
```

AIDE データベースの場所を変更するには、/etc/aide.conf ファイルを編集し、DBDIR 値を変更します。セキュリティーを強化するために、データベース、設定、および /usr/sbin/aide バイナリーファイルを読み取り専用メディアなどの安全な場所に保存します。

重要

AIDE データベースの場所を変更した後に SELinux が拒否されるのを防ぐため、SELinux ポリシーを適宜更新してください。詳細は、SELinux User's and Administrator's Guide を参照してください。

4.11.2. 整合性確認の実行

手動チェックを開始するには、root で以下のコマンドを入力します。

```
~]# aide --check
AIDE 0.15.1 found differences between database and filesystem!!
Start timestamp: 2017-03-30 14:12:56

Summary:
```

```
  Total number of files: 147173
  Added files:   1
  Removed files:  0
  Changed files:  2
...
```

少なくとも、AIDE は週次スキャンを実行するように設定する必要があります。AIDE は毎日実行する必要があります。たとえば、cron を使用して 4:05 am に AIDE の毎日の実行をスケジュールするには（システム管理者のガイドの Automating System Tasks の章を参照）、以下の行を /etc/crontab に追加します。

```
05 4 * * * root /usr/sbin/aide --check
```

4.11.3. AIDE データベースの更新

パッケージの更新や設定ファイルの調整など、システムの変更を確認したら、ベースライン AIDE データベースを更新します。

```
~]# aide --update
```

aide --update コマンドは、/var/lib/aide/aide.db.new.gz データベースファイルを作成します。整合性チェックに使用を開始するには、ファイル名から .new サブストリングを削除します。

4.11.4. 関連情報

AIDE の詳細については、以下のドキュメントを参照してください。

- aide (1) の man ページ
- aide.conf (5) の man ページ
- Guide to the Secure Configuration of Red Hat Enterprise Linux 7 (OpenSCAP Security Guide): Verify Integrity with AIDE

4.12. USBGUARDの使用

USBGuard ソフトウェアフレームワークは、デバイス属性に基づく基本的なホワイトリスト機能およびブラックリスト機能を実装することにより、侵入型 USB デバイスに対するシステム保護を提供し

ます。ユーザー定義のポリシーを適用するために、USBGuard は Linux カーネルの USB デバイス認証機能を使用します。USBGuard フレームワークは、以下のコンポーネントを提供します。

- *動的対話とポリシー施行のためのプロセス間通信 (IPC) インターフェイスを持つデーモンコンポーネント。*
- *実行中の USBGuard インスタンスと対話するコマンドラインインターフェイス。*
- *USB デバイス認証ポリシーを記述するルール言語*
- *共有ライブラリーに実装されているデーモンコンポーネントと対話する C++ API*

4.12.1. USBGuardのインストール

usbguard パッケージをインストールするには、root で以下のコマンドを入力します。

```
~]# yum install usbguard
```

初期ルールセットを作成するには、root で以下のコマンドを入力します。

```
~]# usbguard generate-policy > /etc/usbguard/rules.conf
```

注記

USBGuard ルールセットをカスタマイズするには、/etc/usbguard/rules.conf ファイルを編集します。詳細は、usbguard-rules.conf (5) の man ページを参照してください。使用例は 「ルール言語を使って独自のポリシーを作成する」 を参照してください。

USBGuard デーモンを起動するには、root で以下のコマンドを入力します。

```
~]# systemctl start usbguard.service
~]# systemctl status usbguard
● usbguard.service - USBGuard daemon
   Loaded: loaded (/usr/lib/systemd/system/usbguard.service; disabled; vendor preset: disabled)
   Active: active (running) since Tue 2017-06-06 13:29:31 CEST; 9s ago
     Docs: man:usbguard-daemon(8)
```

```
 Main PID: 4984 (usbguard-daemon)
   CGroup: /system.slice/usbguard.service
           └─4984 /usr/sbin/usbguard-daemon -k -c /etc/usbguard/usbguard-daem...
```

システムの起動時に USBGuard が自動的に開始するようにするには、root で以下のコマンドを使用します。

```
~]# systemctl enable usbguard.service
Created symlink from /etc/systemd/system/basic.target.wants/usbguard.service to
/usr/lib/systemd/system/usbguard.service.
```

USBGuard が認識するすべての USB デバイスを一覧表示するには、root で次のコマンドを実行します。

```
~]# usbguard list-devices
1: allow id 1d6b:0002 serial "0000:00:06.7" name "EHCI Host Controller" hash
"JDOb0BiktYs2ct3mSQKopnOOV2h9MGYADwhT+oUtF2s=" parent-hash
"4PHGcaDKWtPjKDwYpIRG722cB9SlGz9l9lea93+Gt9c=" via-port "usb1" with-interface 09:00:00
...
6: block id 1b1c:1ab1 serial "000024937962" name "Voyager" hash
"CrXgiaWIf2bZAU+5WkzOE7y0rdSO82XMzubn7HDb95Q=" parent-hash
"JDOb0BiktYs2ct3mSQKopnOOV2h9MGYADwhT+oUtF2s=" via-port "1-3" with-interface 08:06:50
```

デバイスがシステムと対話することを許可するには、allow-device オプションを使用します。

```
~]# usbguard allow-device 6
```

デバイスの認証を解除してシステムから削除するには、reject-device オプションを使用します。デバイスの認証を解除するには、block-device オプションを指定して usbguard コマンドを使用します。

```
~]# usbguard block-device 6
```

USBGuard では、block および reject は以下の意味で使用されます。

- **block - 今は、このデバイスとはやりとりしない**
- **reject - このデバイスは存在しないものとして無視する**

usbguard コマンドのすべてのオプションを表示するには、--help ディレクティブを指定して入力します。

```
~]$ usbguard --help
```

4.12.2. ホワイトリストおよびブラックリストの作成

usbguard-daemon.conf ファイルは、コマンドラインオプションを解析した後に usbguard デーモンによって読み込まれ、デーモンのランタイムパラメーターを設定するために使用されます。デフォルトの設定ファイル(/etc/usbguard/usbguard-daemon.conf)をオーバーライドするには、-c コマンドラインオプションを使用します。詳細は、usbguard-daemon (8)の man ページを参照してください。

ホワイトリストまたはブラックリストを作成するには、usbguard-daemon.conf ファイルを編集し、以下のオプションを使用します。

USBGuard 設定ファイル

RuleFile=<path>

usbguard デーモンは、このファイルを使用して、そこからポリシールールセットを読み込み、IPC インターフェイスを介して受信した新しいルールを書き込みます。

IPCAllowedUsers=<username> [<username> ...]

デーモンが IPC 接続を受け入れるユーザー名のスペース区切りのリスト。

IPCAllowedGroups=<groupname> [<groupname> ...]

デーモンが IPC 接続を受け入れるグループ名のスペース区切りのリスト。

IPCAccessControlFiles=<path>

IPC アクセス制御ファイルを保持するディレクトリーへのパス。

ImplicitPolicyTarget=<target>

ポリシーのどのルールにも一致しないデバイスを処理する方法。許容される値は allow、block、および reject です。

PresentDevicePolicy=<policy>

デーモンの起動時にすでに接続されているデバイスを処理する方法:

- *allow - 存在するデバイスをすべて認証する*
- *block - 存在するデバイスの認証をすべて解除する*
- *reject - 存在するデバイスをすべて削除する*
- *keep - 内部ステータスの同期してそのままにする*
- *apply-policy - 存在するすべてのデバイスに対してルールセットを評価する*

PresentControllerPolicy=<policy>

デーモンの起動時にすでに接続されている USB コントローラーを処理する方法:

- *allow - 存在するデバイスをすべて認証する*
- *block - 存在するデバイスの認証をすべて解除する*
- *reject - 存在するデバイスをすべて削除する*
- *keep - 内部ステータスの同期してそのままにする*
- *apply-policy - 存在するすべてのデバイスに対してルールセットを評価する*

例4.5 USBGuard 設定

以下の設定ファイルは、usbguard デーモンが /etc/usbguard/rules.conf ファイルからルールをロードするように順序付け、usbguard グループのユーザーのみが IPC インターフェイスを使用できるようにします。

```
RuleFile=/etc/usbguard/rules.conf
IPCAccessControlFiles=/etc/usbguard/IPCAccessControl.d/
```

IPC アクセス制御リスト(ACL)を指定するには、usbguard add-user または usbguard remove-user コマンドを使用します。詳細は usbguard (1) を参照してください。この例では、usbguard グループのユーザーが USB デバイス承認状態の変更、USB デバイスの一覧表示、例外イベントのリッスン、USB 認証ポリシーの一覧表示を許可するには、root で次のコマンドを実行します。

```
~]# usbguard add-user -g usbguard --devices=modify,list,listen --policy=list --exceptions=listen
```

重要

デーモンは、USBGuard パブリック IPC インターフェイスを提供します。Red Hat Enterprise Linux では、このインターフェイスへのアクセスはデフォルトで root ユーザーのみに制限されます。IPC インターフェイスへのアクセスを制限するには、IPCAccessControlFiles オプション (推奨)、IPCAllowedUsers オプション、および IPCAllowedGroups オプションを設定することを検討してください。ACL を未設定のままにしないでください。設定しないと、すべてのローカルユーザーに IPC インターフェイスが公開され、USB デバイスの認証状態を操作して USBGuard ポリシーを変更できるようになります。

詳細は、usbguard-daemon.conf (5) man ページの IPC Access Control セクションを参照してください。

4.12.3. ルール言語を使って独自のポリシーを作成する

usbguard デーモンは、一連のルールで定義されたポリシーに基づいて USB デバイスを承認するかどうかを決定します。USB デバイスがシステムに挿入されると、デーモンは既存のルールを順次スキャンし、一致するルールが見つかると、ルールのターゲットに基づいて、デバイスを承認 (許可)、非承認 (ブロック)、または削除 (拒否) します。一致するルールが見つからない場合、決定は暗黙のデフォルトターゲットに基づいて行われます。この暗黙のデフォルトとは、ユーザーが決定を下すまでデバイスをブロックすることです。

ルール言語の文法は以下のようになります。

```
rule ::= target device_id device_attributes conditions.

target ::= "allow" | "block" | "reject".

device_id ::= "*:*" | vendor_id ":*" | vendor_id ":" product_id.

device_attributes ::= device_attributes | attribute.
device_attributes ::= .

conditions ::= conditions | condition.
conditions ::= .
```

ターゲット、デバイス仕様、デバイス属性などのルール言語の詳細は、**usbguard-rules.conf (5)** の **man** ページを参照してください。

例4.6 USBguard のサンプルポリシー

USB 大容量ストレージデバイスを許可し、それ以外をすべてブロックする

このポリシーは、単なる大容量ストレージデバイスではないあらゆるデバイスをブロックします。**USB** フラッシュディスクにキーボードインターフェイスが隠されているデバイスはブロックされます。大容量ストレージインターフェイスが **1** つあるデバイスのみが、オペレーティングシステムと対話することを許可されます。ポリシーは、以下のような **1** つのルールで設定されます。

```
allow with-interface equals { 08:*:* }
```

ブロックルールがないため、ブロックは暗黙のうちに行われます。**USBGuard** イベントをリッスンするデスクトップアプレットは、デバイスに対して暗黙的なターゲットが選択されているかどうかをユーザーに尋ねることができるため、暗黙的なブロックはデスクトップユーザーにとって便利です。

特定の Yubikey デバイスを特定のポート経由で接続できるようにする

そのポートの他のすべてを拒否します。

```
allow 1050:0011 name "Yubico Yubikey II" serial "0001234567" via-port "1-2" hash
"044b5e168d40ee0245478416caf3d998"
reject via-port "1-2"
```

インターフェイスの組み合わせに不審な点があるデバイスは排除する

キーボードやネットワークインターフェイスを実装した **USB** フラッシュディスクは非常に

疑わしいものです。次の一連のルールは、USB フラッシュディスクを許可し、追加の疑わしいインターフェイスを備えたデバイスを明示的に拒否するポリシーを形成します。

```
allow with-interface equals { 08:*:* }
reject with-interface all-of { 08:*:* 03:00:* }
reject with-interface all-of { 08:*:* 03:01:* }
reject with-interface all-of { 08:*:* e0:*:* }
reject with-interface all-of { 08:*:* 02:*:* }
```

注記

ブラックリストは間違ったアプローチであり、あるデバイスだけをブラックリストに入れ、残りを許可するということはしないでください。上記のポリシーは、ブロッキングが暗黙のデフォルトであることを前提としています。"不良" と見なされる一連のデバイスを拒否することは、そのようなデバイスへのシステムの露出を可能な限り制限するための良いアプローチです。

キーボードのみの USB デバイスを許可

以下のルールは、キーボードインターフェイスを持つ USB 機器がすでに許可されていない場合にのみ、キーボードのみの USB 機器を許可するものです。

```
allow with-interface one-of { 03:00:01 03:01:01 } if !allowed-matches(with-interface one-of { 03:00:01 03:01:01 })
```

usbguard generate-policy コマンドを使用して最初のポリシーを生成したら、/etc/usbguard/rules.conf を編集して USBGuard ポリシールールをカスタマイズします。

```
~]$ usbguard generate-policy > rules.conf
~]$ vim rules.conf
```

更新されたポリシーをインストールし、変更を有効にするには、次のコマンドを使用します。

```
~]# install -m 0600 -o root -g root rules.conf /etc/usbguard/rules.conf
```

4.12.4. 関連情報

USBGuard の詳細は、以下のドキュメントを参照してください。

- *usbguard (1) の man ページ*
- *usbguard-rules.conf(5) man page*
- *usbguard-daemon(8) man page*
- *usbguard-daemon.conf(5) man page*
- *USBGuard ホームページ*

4.13. TLS 設定の強化

TLS (Transport Layer Security)は、ネットワーク通信のセキュリティーを保護するために使用される暗号化プロトコルです。優先する 鍵交換プロトコル、認証方法、および暗号化アルゴリズムを設定してシステムのセキュリティー設定を強化する際には、対応するクライアントの範囲が広ければ広いほど、セキュリティーのレベルが低くなることを認識しておく必要があります。反対に、セキュリティー設定を厳密にすると、クライアントとの互換性が制限され、システムからロックアウトされるユーザーが出てくる可能性もあります。可能な限り厳密な設定を目指し、互換性に必要な場合に限り、設定を緩めるようにしてください。

Red Hat Enterprise Linux 7 に含まれるライブラリーが提供するデフォルト設定は、ほとんどのデプロイメントで十分に安全である点に留意してください。TLS 実装は、可能な場合は安全なアルゴリズムを使用しますが、レガシークライアントまたはサーバーへの接続を防止しません。セキュリティーが保護されたアルゴリズムまたはプロトコルに対応しないレガシーなクライアントまたはサーバーの接続が期待できないまたは許可されない場合に、厳密なセキュリティー要件の環境で、このセクションで説明されている強化された設定を適用します。

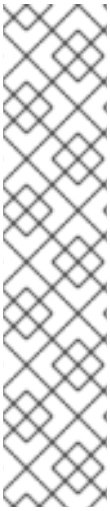
4.13.1. 有効にするアルゴリズムの選択

選択して設定する必要のあるコンポーネントがいくつかあります。以下の各項目は、結果として得られる設定の堅牢性 (および結果的にはクライアントでのサポートレベル)、またはソリューションがシステムに与える計算上の要求に直接影響します。

プロトコルのバージョン

最新バージョンの TLS は、最高のセキュリティーメカニズムを提供します。古いバージョンの TLS（または SSL）のサポートを含めるという面倒な理由がない限り、システムが最新バージョンの TLS のみを使用して接続をネゴシエートできるようにします。

SSL バージョン 2 または 3 を使用したネゴシエーションを許可しないでください。これらのバージョンは両方とも、深刻なセキュリティーの脆弱性があります。TLS バージョン 1.0 以降を使用したネゴシエーションのみを許可します。TLS の現行バージョン 1.2 が常に推奨されます。



注記

現在、TLS のすべてのバージョンのセキュリティーは、 TLS 拡張機能、特定の暗号（以下を参照）、およびその他の回避策の使用に依存することに注意してください。すべての TLS 接続ピアは、安全な再ネゴシエーション表示(RFC 5746)を実装する必要があり、圧縮をサポートしてはならず、CBCモード暗号(Lucky Thirteen 攻撃)に対するタイミング攻撃の緩和策を実装する必要があります。TLS 1.0 クライアントは、レコード分割を追加で実装する必要があります(BEAST 攻撃に対する回避策)。TLS 1.2 は、既知の問題のない AES-GCM、AES-CCM、または Camellia-GCM などの Authenticated Encryption with Associated Data (AEAD)モード暗号をサポートします。ここで述べた緩和策はすべて、Red Hat Enterprise Linux に含まれる暗号ライブラリーに実装されています。

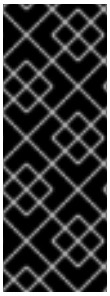
プロトコルのバージョンと推奨される使用方法の概要については、表4.6「プロトコルのバージョン」 を参照してください。

表4.6 プロトコルのバージョン

プロトコルのバージョン	推奨される使用方法
SSL v2	使用しないでください。深刻なセキュリティー上の脆弱性があります。
SSL v3	使用しないでください。深刻なセキュリティー上の脆弱性があります。

プロトコルのバージョン	推奨される使用方法
TLS 1.0	必要に応じて相互運用性の目的で使用します。相互運用性を保証する方法で緩和できない既知の問題があるため、緩和策はデフォルトで有効になっていません。最新の暗号スイートには対応しません。
TLS 1.1	必要に応じて相互運用性の目的で使用します。既知の問題はありませんが、Red Hat Enterprise Linux のすべての TLS 実装に含まれるプロトコル修正に依存します。最新の暗号スイートには対応しません。
TLS 1.2	推奨されるバージョン。最新の AEAD 暗号スイートに対応します。

Red Hat Enterprise Linux の一部のコンポーネントは、TLS 1. 1 または 1.2 のサポートを提供していても、TLS 1.0 を使用するように設定されて い ます。これは、最新バージョンの TLS をサポートしていない可能性のある外部サービスとの最高レベルの相互運用性を実現しようとする試みによって動機付けされます。相互運用性の要件に応じて、利用可能な最高バージョンの TLS を有効にします。



重要

SSL v3 の使用は推奨されません。ただし、安全ではなく、一般的な使用には適していないと考えられているにもかかわらず、SSL v3 を有効にしておく必要があります。暗号化をサポートしていないサービスを使用している場合でも、stunnel を使用して通信を安全に暗号化する方法については、「stunnel の使用」 を参照してください。

暗号化スイート

旧式で、安全ではない 暗号化スイート ではなく、最近の、より安全なものを使用してください。暗号化スイートの eNULL および aNULL は、暗号化や認証を提供しないため、常に無効にしてください。重大な欠点がある RC4 または HMAC-MD5 をベースとする暗号化スイートも可能であれば、無効

にする必要があります。いわゆるエクスポート暗号化スイートも同様です。エクスポート暗号化スイートは意図的に弱くなっているため、侵入が容易になっています。

128 ビット未満のセキュリティーしか提供しない暗号化スイートでは直ちにセキュリティーが保護されなくなるというわけではありませんが、使用できる期間が短いため考慮すべきではありません。アルゴリズムが 128 ビット以上のセキュリティーを使用している場合は、少なくとも数年間は解読不可能であることが期待されているため、強く推奨されます。3DES 暗号は 168 ビットの使用を公開しますが、実際には 112 ビットのセキュリティーを提供します。

サーバーの鍵が危険にさらされた場合でも、暗号化したデータの機密性を保証する (完全な) 前方秘匿性 (PFS) に対応する暗号スイートを常に優先します。このルールにより、高速 RSA 鍵交換は除外されますが、ECDHE および DHE を使用できます。この 2 つのうち、ECDHE が高速であるため、選択が推奨されます。

また、AES-GCM などの AEAD 暗号は、パディング oracle 攻撃に対して脆弱ではないため、CBC-mode 暗号よりも優先する必要があります。また、多くの場合、特にハードウェアに AES の暗号化アクセラレーターがある場合、AES-GCM は CBC モードの AES よりも高速です。

ECDSA 証明書で ECDHE 鍵交換を使用する場合、トランザクションは純粋な RSA 鍵交換よりもさらに高速になります。レガシークライアントのサポートを提供するために、サーバーに証明書と鍵のペアを 2 つインストールします。1 つは ECDSA 鍵（新しいクライアント用）と RSA 鍵（レガシークライアント用）です。

公開鍵の長さ

RSA 鍵を使用する場合は、少なくとも SHA-256 で署名されている 3072 ビット以上の鍵の長さを常に優先します。これは、実際の 128 ビットのセキュリティーに対して十分な大きさです。

警告

システムのセキュリティー強度は、チェーンの中の最も弱いリンクが示すものと同じになることを覚えておいてください。たとえば、強力な暗号化だけではすぐれたセキュリティーは保証されません。鍵と証明書も同様に重要で、認証機関 (CA) が鍵の署名に使用するハッシュ機能と鍵もまた重要になります。

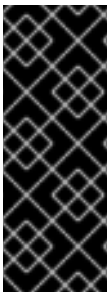
4.13.2. TLS 実装の使用

Red Hat Enterprise Linux 7 には、TLS の複数のフル機能の実装が同梱されています。このセク

ションでは、OpenSSL および GnuTLS の設定について説明します。個々のアプリケーションで TLS サポートを設定する方法については、「特定アプリケーションの設定」 を参照してください。

利用可能な TLS 実装は、TLSで保護された通信を確立および使用する際に結合されるすべての要素を定義するさまざまな 暗号スイート のサポートを提供します。

さまざまな実装に含まれるツールを使用して、「有効にするアルゴリズムの選択」 で概説されている推奨事項を考慮しながら、ユースケースに可能な限り最適なセキュリティーを提供する暗号スイートをリストアップし、指定することができます。このようにして得られた暗号スイートは、個々のアプリケーションが接続をネゴシエートして保護する方法を設定するために使用することができます。



重要

使用する TLS 実装またはその実装を利用するアプリケーションの更新またはアップグレードごとに、必ず設定を確認してください。新しいバージョンでは、有効化したくない、かつ現在の設定では無効化できない新しい暗号スイートが導入される可能性があります。

4.13.2.1. OpenSSL での暗号化スイートの使用

OpenSSL は、SSL プロトコルおよび TLS プロトコルをサポートするツールキットおよび暗号化ライブラリーです。Red Hat Enterprise Linux 7 では、設定ファイルは /etc/pki/tls/openssl.cnf で提供されます。この設定ファイルのフォーマットは、config(1) に記載されています。「OpenSSL の設定」も参照してください。

OpenSSL のインストールでサポートされているすべての暗号スイートの一覧を取得するには、以下のように openssl コマンドを ciphers サブコマンドと共に使用します。

```
~]$ openssl ciphers -v 'ALL:COMPLEMENTOFALL'
```

その他のパラメーター(OpenSSL ドキュメントの 暗号文字列 および キーワード と呼ばれます)を ciphers サブコマンドに渡して、出力を絞り込みます。特別なキーワードを使用して、特定の条件を満たすスイートのみを一覧表示できます。たとえば、HIGH グループに属すると定義されているスイートのみをリストアップするには、次のコマンドを使用します。

```
~]$ openssl ciphers -v 'HIGH'
```

利用可能なキーワードと暗号文字列の一覧は、ciphers(1) の man ページを参照してください。

「有効にするアルゴリズムの選択」 の推奨事項を満たす暗号化スイートを一覧表示するには、以下のようなコマンドを実行します。

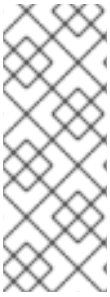
```
~]$ openssl ciphers -v 'kEECDH+aECDSA+AES:kEECDH+AES+aRSA:kEDH+aRSA+AES' |
column -t
ECDHE-ECDSA-AES256-GCM-SHA384  TLSv1.2  Kx=ECDH  Au=ECDSA  Enc=AESGCM(256)
Mac=AEAD
ECDHE-ECDSA-AES256-SHA384      TLSv1.2  Kx=ECDH  Au=ECDSA  Enc=AES(256)
Mac=SHA384
ECDHE-ECDSA-AES256-SHA         SSLv3    Kx=ECDH  Au=ECDSA  Enc=AES(256)    Mac=SHA1
ECDHE-ECDSA-AES128-GCM-SHA256  TLSv1.2  Kx=ECDH  Au=ECDSA  Enc=AESGCM(128)
Mac=AEAD
ECDHE-ECDSA-AES128-SHA256      TLSv1.2  Kx=ECDH  Au=ECDSA  Enc=AES(128)
Mac=SHA256
ECDHE-ECDSA-AES128-SHA         SSLv3    Kx=ECDH  Au=ECDSA  Enc=AES(128)    Mac=SHA1
ECDHE-RSA-AES256-GCM-SHA384    TLSv1.2  Kx=ECDH  Au=RSA    Enc=AESGCM(256)
Mac=AEAD
ECDHE-RSA-AES256-SHA384        TLSv1.2  Kx=ECDH  Au=RSA    Enc=AES(256)    Mac=SHA384
ECDHE-RSA-AES256-SHA           SSLv3    Kx=ECDH  Au=RSA    Enc=AES(256)    Mac=SHA1
ECDHE-RSA-AES128-GCM-SHA256    TLSv1.2  Kx=ECDH  Au=RSA    Enc=AESGCM(128)
Mac=AEAD
ECDHE-RSA-AES128-SHA256        TLSv1.2  Kx=ECDH  Au=RSA    Enc=AES(128)    Mac=SHA256
ECDHE-RSA-AES128-SHA           SSLv3    Kx=ECDH  Au=RSA    Enc=AES(128)    Mac=SHA1
DHE-RSA-AES256-GCM-SHA384      TLSv1.2  Kx=DH    Au=RSA    Enc=AESGCM(256)
Mac=AEAD
DHE-RSA-AES256-SHA256          TLSv1.2  Kx=DH    Au=RSA    Enc=AES(256)    Mac=SHA256
DHE-RSA-AES256-SHA             SSLv3    Kx=DH    Au=RSA    Enc=AES(256)    Mac=SHA1
DHE-RSA-AES128-GCM-SHA256      TLSv1.2  Kx=DH    Au=RSA    Enc=AESGCM(128)
Mac=AEAD
DHE-RSA-AES128-SHA256          TLSv1.2  Kx=DH    Au=RSA    Enc=AES(128)    Mac=SHA256
DHE-RSA-AES128-SHA             SSLv3    Kx=DH    Au=RSA    Enc=AES(128)    Mac=SHA1
```

*上記のコマンドは、セキュアでないすべての暗号を省略し、一時的な省略曲線 **Diffie-Hellman** 鍵交換および **ECDSA** 暗号を優先し、**RSA** 鍵交換を省略します（これにより 完全な転送秘密性が確保されます）。*

これはかなり厳密な設定であり、より広範囲のクライアントとの互換性を確保するために、実際のシナリオの条件を緩和する必要がある場合があることに注意してください。

4.13.2.2. GnuTLS での暗号化スイートの使用

GnuTLS は、SSL および TLS プロトコルおよび関連テクノロジーを実装する通信ライブラリーです。



注記

Red Hat Enterprise Linux 7 での GnuTLS インストールは、ほとんどのユースケースに十分なセキュリティーを提供する最適なデフォルト設定値を提供します。特別なセキュリティー要件を満たす必要がない限り、提供されたデフォルトを使用することが推奨されます。

-l（または --list）オプションを指定して gnutls-cli コマンドを使用して、サポートされているすべての暗号スイートを一覧表示します。

```
~]$ gnutls-cli -l
```

-l オプションによって表示される暗号スイートのリストを絞り込むには、1 つ以上のパラメーター(GnuTLS ドキュメントの 優先度文字列 および キーワード と呼ばれます)を --priority オプションに渡します。利用可能なすべての優先度文字列の一覧は、http://www.gnutls.org/manual/gnutls.html#Priority-Strings の GnuTLS ドキュメントを参照してください。たとえば、次のコマンドを実行すると、少なくとも 128 ビットのセキュリティーを提供する暗号スイートのリストを取得できます。

```
~]$ gnutls-cli --priority SECURE128 -l
```

「有効にするアルゴリズムの選択」 の推奨事項を満たす暗号化スイートを一覧表示するには、以下のようなコマンドを実行します。

```
~]$ gnutls-cli --priority SECURE256:+SECURE128:-VERS-TLS-ALL:+VERS-TLS1.2:-RSA:-DHE-
DSS:-CAMELLIA-128-CBC:-CAMELLIA-256-CBC -l
Cipher suites for SECURE256:+SECURE128:-VERS-TLS-ALL:+VERS-TLS1.2:-RSA:-DHE-DSS:-
CAMELLIA-128-CBC:-CAMELLIA-256-CBC
TLS_ECDHE_ECDSA_AES_256_GCM_SHA384                0xc0, 0x2c      TLS1.2
TLS_ECDHE_ECDSA_AES_256_CBC_SHA384                0xc0, 0x24      TLS1.2
TLS_ECDHE_ECDSA_AES_256_CBC_SHA1                  0xc0, 0x0a      SSL3.0
TLS_ECDHE_ECDSA_AES_128_GCM_SHA256                0xc0, 0x2b      TLS1.2
TLS_ECDHE_ECDSA_AES_128_CBC_SHA256                0xc0, 0x23      TLS1.2
TLS_ECDHE_ECDSA_AES_128_CBC_SHA1                  0xc0, 0x09      SSL3.0
TLS_ECDHE_RSA_AES_256_GCM_SHA384                  0xc0, 0x30      TLS1.2
TLS_ECDHE_RSA_AES_256_CBC_SHA1                    0xc0, 0x14      SSL3.0
TLS_ECDHE_RSA_AES_128_GCM_SHA256                  0xc0, 0x2f      TLS1.2
TLS_ECDHE_RSA_AES_128_CBC_SHA256                  0xc0, 0x27      TLS1.2
TLS_ECDHE_RSA_AES_128_CBC_SHA1                    0xc0, 0x13      SSL3.0
TLS_DHE_RSA_AES_256_CBC_SHA256                    0x00, 0x6b      TLS1.2
TLS_DHE_RSA_AES_256_CBC_SHA1                      0x00, 0x39      SSL3.0
TLS_DHE_RSA_AES_128_GCM_SHA256                    0x00, 0x9e      TLS1.2
TLS_DHE_RSA_AES_128_CBC_SHA256                    0x00, 0x67      TLS1.2
TLS_DHE_RSA_AES_128_CBC_SHA1                      0x00, 0x33      SSL3.0

Certificate types: CTYPE-X.509
Protocols: VERS-TLS1.2
```

```
Compression: COMP-NULL
Elliptic curves: CURVE-SECP384R1, CURVE-SECP521R1, CURVE-SECP256R1
PK-signatures: SIGN-RSA-SHA384, SIGN-ECDSA-SHA384, SIGN-RSA-SHA512, SIGN-ECDSA-
SHA512, SIGN-RSA-SHA256, SIGN-DSA-SHA256, SIGN-ECDSA-SHA256
```

上記のコマンドは、出力を少なくとも 128 ビットのセキュリティーを備えた暗号に制限し、より強力な暗号を優先します。また、RSA 鍵交換と DSS 認証も禁止されています。

これはかなり厳密な設定であり、より広範囲のクライアントとの互換性を確保するために、実際のシナリオの条件を緩和する必要がある場合があることに注意してください。

4.13.3. 特定アプリケーションの設定

さまざまなアプリケーションが TLS に独自の設定メカニズムを提供します。このセクションでは、最も一般的に使用されるサーバーアプリケーションで採用されている TLS関連の設定ファイルについて説明し、一般的な設定の例を提供します。

いずれの設定を選択しても、サーバーアプリケーションが強制的に サーバー側が指定した順序 で暗号を利用することを確認し、使用される暗号化スイートの選択がサーバーでの設定順に行われるように設定してください。

4.13.3.1. Apache HTTP Server の設定

Apache HTTP Server は、TLS のニーズに OpenSSL ライブラリーと NSS ライブラリーの両方を使用できます。TLS ライブラリーの選択に応じて、mod_ssl または mod_nss モジュールのいずれかをインストールする必要があります（ディープパッケージで提供される）。たとえば、OpenSSL mod_ssl モジュールを提供するパッケージをインストールするには、root で以下のコマンドを実行します。

```
~]# yum install mod_ssl
```

mod_ssl パッケージは、/etc/httpd/conf.d/ssl.conf 設定ファイルをインストールします。このファイルを使用して、Apache HTTP Server の TLS関連の設定を変更します。同様に、mod_nss パッケージは /etc/httpd/conf.d/nss.conf 設定ファイルをインストールします。

httpd-manual パッケージをインストールして、TLS 設定を含む Apache HTTP Server の完全なドキュメントを取得します。/etc/httpd/conf.d/ssl.conf 設定ファイルで利用可能なディレクティブの詳細は、/usr/share/httpd/manual/mod/mod_ssl.html を参照してください。さまざまな設定の例は /usr/share/httpd/manual/ssl/ssl_howto.html にあります。

/etc/httpd/conf.d/ssl.conf 設定ファイルの設定を変更する場合は、少なくとも以下の 3 つのディレクティブを考慮してください。

SSLProtocol

このディレクティブを使用して、許可する TLS （または SSL）のバージョンを指定します。

SSLCipherSuite

優先する暗号化スイートを指定する、もしくは許可しないスイートを無効にするディレクティブです。

SSLHonorCipherOrder

コメントを解除して、このディレクティブを on に設定し、接続しているクライアントが指定した暗号の順序に準拠するようにします。

以下に例を示します。

```
SSLProtocol all -SSLv2 -SSLv3
SSLCipherSuite HIGH:!aNULL:!MD5
SSLHonorCipherOrder on
```

上記の設定は最低限のものであり、「有効にするアルゴリズムの選択」 に概説されている推奨事項に従うことで、大幅に強化できることに注意してください。

mod_nss モジュールを設定して使用するには、/etc/httpd/conf.d/nss.conf 設定ファイルを変更します。mod_nss モジュールは mod_ssl から派生しているため、設定ファイルの構造や利用可能なディレクティブなど、多くの機能を共有します。mod_nss ディレクティブには、SSL の代わりに NSS の接頭辞があることに注意してください。mod_nss に該当しない mod_ssl 設定ディレクティブの一覧を含む mod_nss に関する情報の概要
は、https://git.fedorahosted.org/cgit/mod_nss.git/plain/docs/mod_nss.html を参照してください。

4.13.3.2. Dovecot メールサーバーの設定

TLS を使用するように Dovecot メールサーバーのインストールを設定するには、/etc/dovecot/conf.d/10-ssl.conf 設定ファイルを変更します。このファイルで利用可能な基本的な設定ディレクティブの一部は、/usr/share/doc/dovecot-2.2.10/wiki/SSL.DovecotConfiguration.txt にあります（このヘルプファイルは、Dovecotの標準インストールとともにインストールされます）。

/etc/dovecot/conf.d/10-ssl.conf 設定ファイルの設定を変更する場合は、少なくとも以下の 3 つのディレクティブを考慮してください。

ssl_protocols

このディレクティブを使用して、許可する TLS （または SSL）のバージョンを指定します。

ssl_cipher_list

優先する暗号化スイートを指定する、もしくは許可しないスイートを無効にするディレクティブです。

ssl_prefer_server_ciphers

コメントを解除して、このディレクティブを yes に設定して、接続しているクライアントが指定した暗号の順序に準拠するようにします。

以下に例を示します。

```
ssl_protocols = !SSLv2 !SSLv3
ssl_cipher_list = HIGH:!aNULL:!MD5
ssl_prefer_server_ciphers = yes
```

上記の設定は最低限のものであり、「有効にするアルゴリズムの選択」 に概説されている推奨事項に従うことで、大幅に強化できることに注意してください。

4.13.4. 追加情報

TLS 設定および関連トピックの詳細は、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

- config(1) : /etc/ssl/openssl.conf 設定ファイルの形式を説明します。
- ciphers(1) : 利用可能な OpenSSL キーワードおよび暗号文字列の一覧が含まれます。
- /usr/share/httpd/manual/mod/ mod_ssl.html: Apache HTTP Server の mod_ssl モジュールが使用する /etc/httpd/conf.d/ssl.conf 設定ファイルで利用可能なディレクティブの詳

細が含まれています。

- /usr/share/httpd/manual/ssl/ssl_howto.html: Apache HTTP Server の mod_ssl モジュールが使用する /etc/httpd/conf.d/ssl.conf 設定ファイルの実際の設定例が含まれています。

- /usr/share/doc/dovecot-2.2. 10/wiki/SSL.DovecotConfiguration.txt : Dovecot メールサーバーが使用する /etc/dovecot/conf.d/10-ssl.conf 設定ファイルで利用可能な基本的な設定ディレクティブの一部を説明します。

オンラインドキュメント

- Red Hat Enterprise Linux 7 SELinux ユーザーおよび管理者のガイド: Red Hat Enterprise Linux 7 『の SELinux ユーザーおよび管理者のガイド』 では、SELinux の原則と、SELinux を Apache HTTP Server などのさまざまなサービスで設定および使用する方法を説明します。

- http://tools.ietf.org/html/draft-ietf-uta-tls-bcp-00 - TLS および DTLS を安全に使用するための推奨事項です。

関連項目

- 「SSL/TLS」 は、SSL プロトコルおよび TLS プロトコルの簡潔な説明を提供します。

- 「OpenSSL の使用」 では、OpenSSL を使用して鍵を作成および管理し、証明書を生成し、ファイルを暗号化および復号化する方法について説明しています。

4.14. 共通システム証明書の使用

共有システム証明書ストレージは、NSS、GnuTLS、OpenSSL、および Java が、システムの証明書アンカーと、ブラックリスト情報を取得するデフォルトソースを共有することを許可します。トラストストアには、デフォルトで、Mozilla CA の一覧 (信頼される一覧および信頼されない一覧) を含みます。このシステムでは、コアとなる Mozilla CA リストの更新や、他の証明書リストの選択が可能です。

4.14.1. システム全体でトラストストアの使用

Red Hat Enterprise Linux 7 では、統合されたシステム全体のトラストストアが /etc/pki/ca-trust/ ディレクトリーおよび /usr/share/pki/ca-trust-source/ ディレクトリーに置かれま

す。/usr/share/pki/ca-trust-source/ の信頼設定は、/etc/pki/ca-trust/ の設定よりも低い優先順位で処理されます。

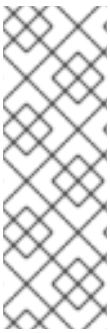
証明書ファイルは、インストールされているサブディレクトリーによって扱われ方が異なります。

- */usr/share/pki/ca-trust-source/anchors/ または /etc/pki/ca-trust/source/anchors/ - トラストアンカーの場合。「トラストアンカーについて」を参照してください。*
- */usr/share/pki/ca-trust-source/blacklist/ または /etc/pki/ca-trust/source/blacklist/ - 信頼できない証明書の場合。*
- */usr/share/pki/ca-trust-source/ または /etc/pki/ca-trust/source/ - 拡張 BEGIN TRUSTED ファイル形式の証明書の場合。*

4.14.2. 新しい証明書の追加

システムで信頼されている CA の一覧に、シンプルな PEM または DER ファイル形式で証明書を追加するには、証明書ファイルを /usr/share/pki/ca-trust-source/anchors/ ディレクトリーまたは /etc/pki/ca-trust/source/anchors/ ディレクトリーにコピーします。システム全体のトラストストア設定を更新するには、update-ca-trust コマンドを使用します。以下に例を示します。

```
# cp ~/certificate-trust-examples/Cert-trust-test-ca.pem /usr/share/pki/ca-trust-source/anchors/
# update-ca-trust
```



注記

Firefox ブラウザーは update-ca-trust を実行せずに追加した証明書を使用できますが、CA の変更後に update-ca-trust を実行することが推奨されます。Firefox、Epiphany、Chromium などのブラウザーはファイルをキャッシュするため、現在のシステム証明書の設定を読み込むために、ブラウザーのキャッシュを削除して、ブラウザーを再起動することが必要になるかもしれません。

4.14.3. 信頼されているシステム証明書の管理

トラストアンカーを一覧表示、抽出、追加、削除、または変更するには、trust コマンドを使用します。このコマンドの組み込みヘルプを表示するには、引数を付けずに、または --help ディレクティブを付けて実行します。

```
$ trust
```

```
usage: trust command <args>...

Common trust commands are:
  list            List trust or certificates
  extract         Extract certificates and trust
  extract-compat  Extract trust compatibility bundles
  anchor          Add, remove, change trust anchors
  dump            Dump trust objects in internal format

See 'trust <command> --help' for more information
```

システムのトラストアンカーおよび証明書の一覧を表示するには、trust list コマンドを使用します。

```
$ trust list
pkcs11:id=%d2%87%b4%e3%df%37%27%93%55%f6%56%ea%81%e5%36%cc%8c%1e%3f%bd;type=cert
    type: certificate
    label: ACCVRAIZ1
    trust: anchor
    category: authority

pkcs11:id=%a6%b3%e1%2b%2b%49%b6%d7%73%a1%aa%94%f5%01%e7%73%65%4c%ac%50;type=cert
    type: certificate
    label: ACEDICOM Root
    trust: anchor
    category: authority
...
[output has been truncated]
```

trust コマンドのすべてのサブコマンドは、以下のような詳細な組み込みヘルプを提供します。

```
$ trust list --help
usage: trust list --filter=<what>

  --filter=<what>     filter of what to export
                      ca-anchors        certificate anchors
                      blacklist         blacklisted certificates
                      trust-policy      anchors and blacklist (default)
                      certificates      all certificates
                      pkcs11:object=xx  a PKCS#11 URI
  --purpose=<usage>   limit to certificates usable for the purpose
                      server-auth       for authenticating servers
                      client-auth       for authenticating clients
                      email             for email protection
                      code-signing      for authenticating signed code
                      1.2.3.4.5...      an arbitrary object id
  -v, --verbose       show verbose debug output
  -q, --quiet         suppress command output
```

トラストアンカーをシステム全体のトラストストアに保存するには、trust anchor サブコマンドを使用して、証明書への path.to を指定します。以下に例を示します。

```
# trust anchor path.to/certificate.crt
```

証明書を削除するには、証明書の path.to、または証明書の ID を使用します。

```
# trust anchor --remove path.to/certificate.crt
# trust anchor --remove "pkcs11:id=%AA%BB%CC%DD%EE;type=cert"
```

4.14.4. 関連情報

詳細は、以下の man ページを参照してください。

- update-ca-trust(8)
- trust(1)

4.15. MACSEC の使用

Media Access Control Security (MACsec IEEE 802.1AE)は、LAN のすべてのトラフィックを GCM-AES-128 アルゴリズムで暗号化して認証します。MACsec は、IP だけでなく、ARP (Address Resolution Protocol)、ND (ND)、または DHCP も保護できます。IPsec は、ネットワーク層（レイヤー 3）および SSL または TLS をアプリケーション層（レイヤー 7）上で動作しますが、MACsec はデータリンク層（レイヤー 2）で動作します。MACsec と他のネットワーク層のセキュリティープロトコルを組み合わせて、これらの標準規格が提供するさまざまなセキュリティー機能を活用します。

MACsec ネットワークのアーキテクチャー、ユースケースシナリオ、設定例の詳細は、MACsec: a different solution to encrypt network traffic の記事を参照してください。

wpa_supplicant および NetworkManager を使用して MACsec を設定する方法は、Red Hat Enterprise Linux 7 ネットワークガイド を参照してください。

4.16. SCRUBを使用してデータを安全に削除

scrub ユーティリティーは、特別なファイルまたはディスクデバイスにパターンを設定して、データ

の取得をより困難にします。scrub の使用は、ディスクにランダムデータを書き込むよりも高速です。このプロセスは、高可用性、信頼性、およびデータ保護を提供します。

scrub コマンドの使用を開始するには、scrub パッケージをインストールします。

```
~]# yum install scrub
```

scrub ユーティリティーは、以下の基本モードのいずれかで動作します。

文字またはブロックデバイス

ディスク全体に対応する 特殊ファイル がスクラブされ、その上のすべてのデータが破棄されます。これが最も効果的な方法です。

```
scrub [OPTIONS] special file
```

ファイル

通常のファイルはスクラブされ、ファイル内のデータのみが破棄されます。

```
scrub [OPTIONS] file
```

ディレクトリー

-X オプションを使用すると、ディレクトリーが作成され、ファイルシステムがいっぱいになるまで、ファイルで埋め尽くされます。次に、ファイルモードの場合と同様にファイルがスクラブされます。

```
scrub -X [OPTIONS] directory
```

例4.7 Raw デバイスのスクラブ

デフォルトの National Nuclear Security Administration (NNSA)パターンを使用して raw デバイス /dev/sdf1 を スクラブ するには、次のコマンドを入力します。

```
~]# scrub /dev/sdf1
scrub: using NNSA NAP-14.1-C patterns
scrub: please verify that device size below is correct!
scrub: scrubbing /dev/sdf1 1995650048 bytes (~1GB)
scrub: random  |.....................................................|
```

```
scrub: random  |.....................................|
scrub: 0x00    |.....................................|
scrub: verify  |.....................................|
```

例4.8 ファイルのスクラブ

1. **1MB ファイルを作成します。**

```
~]$ base64 /dev/urandom | head -c $[ 1024*1024 ] > file.txt
```

2. **ファイルサイズを表示します。**

```
~]$ ls -lh
total 1.0M
-rw-rw-r--. 1 username username 1.0M Sep  8 15:23 file.txt
```

3. **ファイルの内容を表示します。**

```
~]$ head -1 file.txt
JnNpaTEveB/lYsbM9lhuJdw+0jKhwClBUsxLXLAyB8ultotUlNHKKUeS/7bCRKDogE
P+yJm8VQkL
```

4. **ファイルをスクラブします。**

```
~]$ scrub file.txt
scrub: using NNSA NAP-14.1-C patterns
scrub: scrubbing file.txt 1048576 bytes (~1024KB)
scrub: random  |.....................................|
scrub: random  |.....................................|
scrub: 0x00    |.....................................|
scrub: verify  |.....................................|
```

5. **ファイルの内容がスクラブされていることを確認します。**

```
~]$ cat file.txt
SCRUBBED!
```

6. ファイルサイズに変更がないことを確認します。

```
~]$ ls -lh
total 1.0M
-rw-rw-r--. 1 username username 1.0M Sep  8 15:24 file.txt
```

スクラブ モード、オプション、方法、および注意事項の詳細は、scrub(1) の man ページを参照してください。

第5章 ファイアウォールの使用

5.1. FIREWALLDの使用

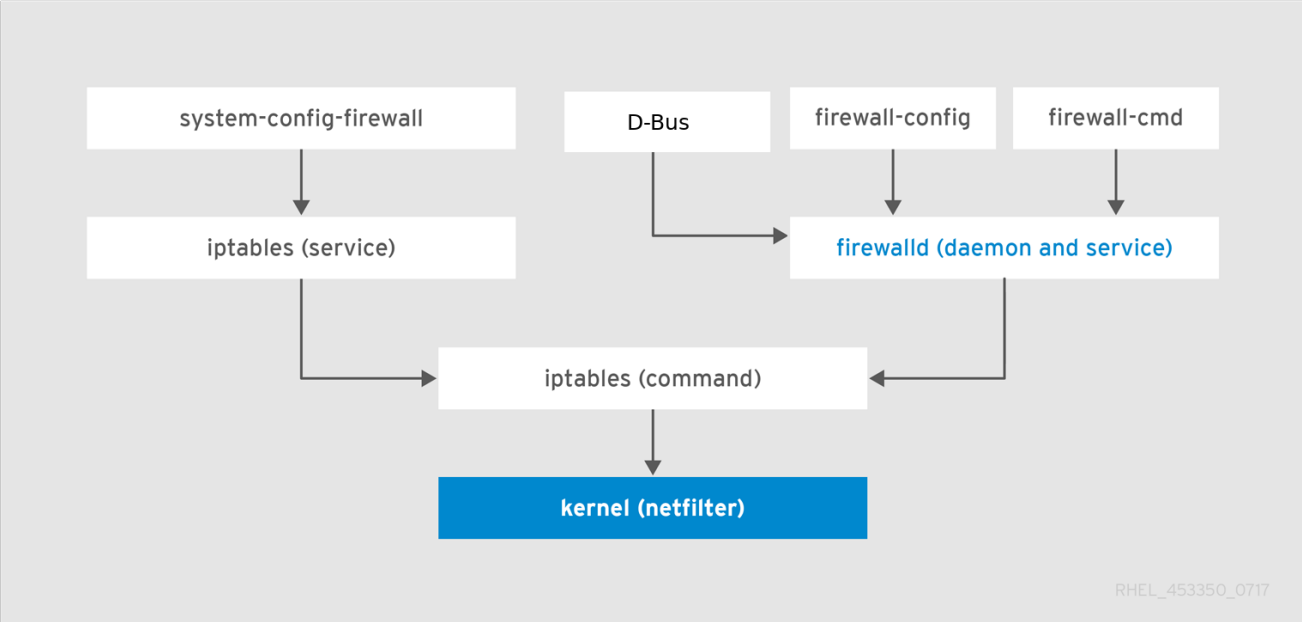
ファイアウォール は、外部からの不要なトラフィックからマシンを保護する方法です。ファイアウォールルール セットを定義することで、ホストマシンへの着信ネットワークトラフィックを制御できます。このようなルールは、着信トラフィックを分類して、拒否または許可するために使用されます。

firewalld は、**D-Bus** インターフェイスを使用して動的にカスタマイズ可能なホストベースのファイアウォールを提供するファイアウォールサービスデーモンです。ルールが変更するたびに、ファイアウォールデーモンを再起動しなくても、ルールの作成、変更、および削除を動的に可能にします。

firewalld は、ゾーン および サービス の概念を使用して、トラフィック管理を簡素化します。ゾーンは、事前定義したルールセットです。ネットワークインターフェイスおよびソースをゾーンに割り当てることができます。許可されているトラフィックは、コンピューターが接続するネットワークと、このネットワークが割り当てられているセキュリティーレベルに従います。ファイアウォールサービスは、特定のサービスに着信トラフィックを許可するのに必要なすべての設定を扱う事前定義のルールで、ゾーンに適用されます。

サービスは、ネットワーク接続に 1 つ以上の ポート または アドレス を使用します。ファイアウォールは、ポートに基づいて接続のフィルターを設定します。サービスに対してネットワークトラフィックを許可するには、そのポートを 開く 必要があります。**firewalld** は、明示的に開いていないポートのトラフィックをすべてブロックします。**trusted** などのゾーンでは、デフォルトですべてのトラフィックを許可します。

図5.1 ファイアウォールスタック



[D]

5.1.1. ゾーン

firewalld を使用して、ユーザーがインターフェイスに配置する信頼レベルと、そのネットワーク内のトラフィックに応じて、複数のゾーンにネットワークを分離できます。接続は、1 つのゾーンにしか指定できませんが、ゾーンは多くのネットワーク接続に使用できます。

NetworkManager は、firewalld にインターフェイスのゾーンを通知します。NetworkManager、firewall-config ツール、または firewall-cmd コマンドラインツールを使用して、インターフェイスにゾーンを割り当てることができます。後者の 2 つは、適切な NetworkManager 設定ファイルのみを編集します。firewall-cmd または firewall-config を使用してインターフェイスのゾーンを変更すると、リクエストは NetworkManager に転送され、firewalld では処理されません。

事前定義したゾーンは /usr/lib/firewalld/zones/ ディレクトリーに保存され、利用可能なネットワークインターフェイスに即座に適用できます。このファイルは、変更しないと /etc/firewalld/zones/ ディレクトリーにコピーされません。以下の表は、事前定義したゾーンのデフォルト設定について説明しています。

block

IPv4 の場合は icmp-host-prohibited メッセージ、IPv6 の場合は icmp6- adm-prohibited メッセージで、すべての着信ネットワーク接続が拒否されます。システムで開始したネットワーク接続のみが可能です。

dmz

公開アクセスは可能ですが、内部ネットワークへのアクセスに制限がある非武装地帯にあるコンピューター向けです。選択した着信接続のみが許可されます。

drop

着信ネットワークパケットは、通知なしで遮断されます。発信ネットワーク接続だけが可能です。

external

マスカレードをルーター用に特別に有効にした外部ネットワークでの使用向けです。自分のコンピューターを保護するため、ネットワーク上の他のコンピューターを信頼しません。選択した着

信接続のみが許可されます。

home

そのネットワークでその他のコンピューターをほぼ信頼できる自宅での使用向けです。選択した着信接続のみが許可されます。

internal

そのネットワークでその他のコンピューターをほぼ信頼できる内部ネットワーク向けです。選択した着信接続のみが許可されます。

public

そのネットワークでその他のコンピューターを信頼できないパブリックエリーア向けです。選択した着信接続のみが許可されます。

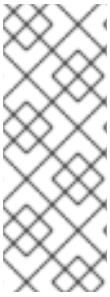
trusted

すべてのネットワーク接続が許可されます。

work

そのネットワークで、その他のコンピューターをほぼ信頼できる職場での使用向けです。選択した着信接続のみが許可されます。

このゾーンのいずれかを デフォルト ゾーンに設定できます。インターフェイス接続が NetworkManager に追加されると、デフォルトゾーンに割り当てられます。インストールでは、firewalld のデフォルトゾーンが public ゾーンに設定されます。デフォルトゾーンは変更できます。



注記

ネットワークゾーン名は、分かりやすく、ユーザーが妥当な決定をすばやく下せるような名前が付けられています。セキュリティー問題を回避するために、ニーズおよびリスク評価に合わせて、デフォルトゾーンの設定の見直しを行ったり、不要なサービスを無効にしてください。

5.1.2. 事前定義サービス

サービスが、ローカルポート、プロトコル、ソースポート、宛先、そしてサービスが有効になると自動的に読み込まれるファイアウォールのヘルパーモジュールの一覧を指す場合があります。サービスを使用すると、ポートのオープン、プロトコルの定義、パケット転送の有効化などを 1 つ 1 つ行うのではなく、1 回のステップで定義できます。

サービス設定オプションと一般的なファイル情報は、firewalld.service (5) の man ページで説明されています。サービスは、個別の XML 設定ファイルで指定します。このファイルは、service-name.xml の形式で指定されます。プロトコル名は、firewalld のサービス名またはアプリケーション名よりも優先されます。

5.1.3. ランタイムおよび永続化の設定

ランタイム モードでコミットされた変更は、firewalld が実行されている間にのみ適用されます。firewalld を再起動すると、設定は 永続的な 値に戻ります。

変更した内容を再起動後も持続させる場合は、--permanent オプションを使用して適用します。firewalld の実行中に変更を永続化させるには、--runtime-to-permanent firewall-cmd オプションを使用します。

--permanent オプションのみを使用して firewalld の実行中にルールを設定すると、firewalld が再起動する前にルールが有効になりません。ただし、firewalld を再起動すると、開いているポートがすべて閉じ、ネットワークトラフィックが停止します。

5.1.4. CLI を使用したランタイムおよび永続化の設定の変更

CLI では、2 つのモードのファイアウォール設定を同時に修正することができません。CLI では、ランタイムまたは永続モードを修正します。永続的なモードでファイアウォール設定を変更するには、firewall-cmd コマンドで --permanent オプションを使用します。

```
~]# firewall-cmd --permanent <other options>
```

このオプションを使用しないと、コマンドはランタイムモードを変更します。

両方のモードで設定を変更する場合は、2 つの方法を使用できます。

1. *以下のように、ランタイム設定を変更して、永続化します。*

```
~]# firewall-cmd <other options>
~]# firewall-cmd --runtime-to-permanent
```

2. *永続的な設定を行い、ランタイムモードで設定を再ロードします。*

```
~]# firewall-cmd --permanent <other options>
~]# firewall-cmd --reload
```

最初の方法では、永続モードで設定を適用する前に、設定をテストできます。

注記

*特にリモートシステムでは、設定を間違えると、ユーザーが自身をロックする結果となります。そのような状況を回避するには、**--timeout** オプションを使用します。指定した時間が経つと、変更は元に戻ります。このオプションを使用した場合は、**--permanent** オプションが無効になります。*

*たとえば、**15 分間 SSH** サービスを追加するには、次のコマンドを実行します。*

```
~]# firewall-cmd --add-service=ssh --timeout 15m
```

5.2. FIREWALL-CONFIG GUI 設定ツールのインストール

*firewall-config GUI 設定ツールを使用するには、**firewall-config** パッケージを **root** としてインストールします。*

```
~]# yum install firewall-config
```

*または、**GNOME** で **Super** キーを使用して **Software** と入力して、**Software Sources** アプリケーションを起動します。右上隅の検索ボタンを選択すると表示される検索ボックスに **firewall** と入力しま*

す。検索結果から Firewall 項目を選択し、Install ボタンをクリックします。

firewall-config を実行するには、firewall-config コマンドを使用するか、Super キーを押してアクティビティーの 概要に入り、firewall と 入力 して Enter を押します。

5.3. FIREWALLDの現在の状況および設定の表示

5.3.1. firewalldの現在の状況の表示

ファイアウォールサービスの firewalld は、デフォルトでシステムにインストールされます。firewalld CLI インターフェイスを使用して、サービスが実行されていることを確認します。

サービスの状況を表示するには、次のコマンドを実行します。

```
~]# firewall-cmd --state
```

サービスステータスの詳細は、systemctl status サブコマンドを使用します。

```
~]# systemctl status firewalld
firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor pr
   Active: active (running) since Mon 2017-12-18 16:05:15 CET; 50min ago
     Docs: man:firewalld(1)
 Main PID: 705 (firewalld)
    Tasks: 2 (limit: 4915)
   CGroup: /system.slice/firewalld.service
           └─705 /usr/bin/python3 -Es /usr/sbin/firewalld --nofork --nopid
```

さらに、設定を編集する前に、firewalld の設定方法と、強制しているルールを確認することが重要です。ファイアウォール設定を表示する場合は、「現在の firewalld 設定の表示」を参照してください。

5.3.2. 現在の firewalld 設定の表示

5.3.2.1. GUI を使用して許可されるサービスの表示

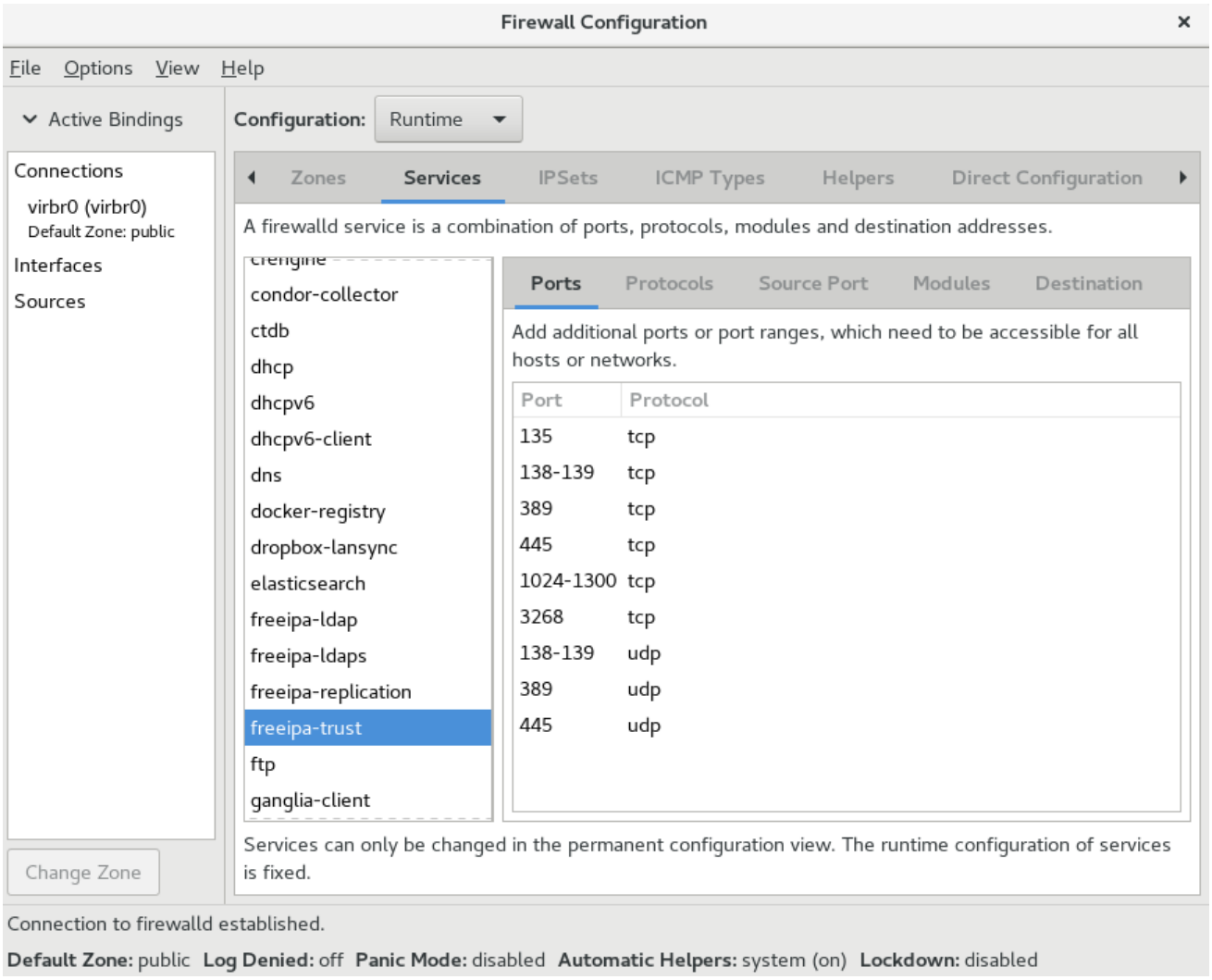
グラフィカルの firewall-config ツールを使用してサービスの一覧を表示するには、Super キーを押してアクティビティーの概要に入り、firewall と 入力 して Enter を押します。firewall-config ツールが表示されます。Services タブの下にサービスの一覧が表示されます。

もしくは、コマンドラインを使用してグラフィカルなファイアウォール設定ツールを起動する場合は、次のコマンドを入力します。

```
~]$ firewall-config
```

Firewall Configuration ウィンドウが開きます。このコマンドは通常のユーザーとして実行できますが、監理者パスワードが求められる場合もあります。

図5.2 firewall-config のサービスタブ



[D]

5.3.2.2. CLI を使用した firewalld 設定の表示

CLI クライアントで、現在のファイアウォール設定を、複数の方法で表示できます。**--list-all** オプションは、**firewalld** 設定の完全な概要を表示します。

firewalld はゾーンを使用してトラフィックを管理します。**--zone** オプションでゾーンを指定しない

と、コマンドは、アクティブネットワークインターフェイスおよび接続に割り当てたデフォルトゾーンに対して有効になります。

デフォルトゾーンに関連する情報をすべて表示するには、次のコマンドを実行します。

```
~]# firewall-cmd --list-all
public
  target: default
  icmp-block-inversion: no
  interfaces:
  sources:
  services: ssh dhcpv6-client
  ports:
  protocols:
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

注記

設定を表示するゾーンを指定するには、**--zone=zone-name**引数を **firewall- cmd --list-all** コマンドに追加します。以下に例を示します。

```
~]# firewall-cmd --list-all --zone=home
home
  target: default
  icmp-block-inversion: no
  interfaces:
  sources:
  services: ssh mdns samba-client dhcpv6-client
... [output truncated]
```

サービス、ポートなど、特定情報の設定を確認するには、特定のオプションを使用します。**firewalld** の **man** ページを参照するか、コマンド **help** を使用してオプションの一覧を表示します。

```
~]# firewall-cmd --help

Usage: firewall-cmd [OPTIONS...]

General Options
  -h, --help          Prints a short help text and exists
  -V, --version       Print the version string of firewalld
  -q, --quiet         Do not print status messages
```

```
Status Options
  --state             Return and print firewalld state
  --reload            Reload firewall and keep state information
... [output truncated]
```

たとえば、現在のゾーンで許可されているサービスを表示します。

```
~]# firewall-cmd --list-services
ssh dhcpv6-client
```

CLI ツールを使用して一覧表示した特定のサブパートの設定は、解釈が難しいことがしばしばあります。たとえば、SSH サービスを許可し、firewalld がサービスに必要なポート(22)を開くことができます。後に、許可されたサービスを一覧表示すると、リストには SSH サービスが表示されますが、開いているポートを一覧表示すると、何も表示されません。したがって、--list-all オプションを使用して、完全な情報を取得することが推奨されます。

5.4. FIREWALLDの起動

firewalld を起動するには、root で以下のコマンドを入力します。

```
~]# systemctl unmask firewalld
~]# systemctl start firewalld
```

システムの起動時に firewalld が自動的に起動するようにするには、root で次のコマンドを実行します。

```
~]# systemctl enable firewalld
```

5.5. FIREWALLDの停止

firewalld を停止するには、root で以下のコマンドを入力します。

```
~]# systemctl stop firewalld
```

システムの起動時に firewalld が自動的に起動しないようにするには、root で次のコマンドを実行します。

```
~]# systemctl disable firewalld
```

firewalld D-Bus インターフェイスにアクセスして firewalld を開始しておらず、他のサービスで firewalld が必要な場合は、root で次のコマンドを実行します。

```
~]# systemctl mask firewalld
```

5.6. トラフィックの制御

5.6.1. 事前定義サービス

サービスは、グラフィカルな firewall-config ツール、firewall-cmd、および firewall-offline-cmd を使用して追加および削除できます。

または、/etc/firewalld/services/ ディレクトリーの XML ファイルを編集することもできます。ユーザーがサービスを追加または変更しないと、対応する XML ファイルは /etc/firewalld/services/ にはありません。サービスを追加または変更する場合は、/usr/lib/firewalld/services/ ディレクトリーのファイルをテンプレートとして使用できます。

5.6.2. 緊急時に CLI を使用してすべてのトラフィックの無効化

システムへの攻撃などの緊急な状態にあるとき、すべてのネットワークトラフィックを無効にし、攻撃を遮断できます。

ネットワークトラフィックを直ちに無効にするには、パニックモードをオンにします。

```
~]# firewall-cmd --panic-on
```

パニックモードをオフにし、ファイアウォールを永続設定に戻します。パニックモードを無効にするには、次のコマンドを実行します。

```
~]# firewall-cmd --panic-off
```

パニックモードを有効または無効にするには、次のコマンドを実行します。

```
~]# firewall-cmd --query-panic
```

5.6.3. CLI を使用して事前定義されたサービスでトラフィックの制御

トラフィックを制御する最も簡単な方法は、事前定義サービスを firewalld に追加することです。こ

れにより、必要なすべてのポートが開き、service definition file に従ってその他の設定が変更されます。

1. サービスが許可されていないことを確認します。

```
~]# firewall-cmd --list-services
ssh dhcpv6-client
```

2. 事前定義したサービスの一覧を表示します。

```
~]# firewall-cmd --get-services
RH-Satellite-6 amanda-client amanda-k5-client bacula bacula-client bitcoin bitcoin-rpc
bitcoin-testnet bitcoin-testnet-rpc ceph ceph-mon cfengine condor-collector ctdb dhcp dhcpv6
dhcpv6-client dns docker-registry ...
[output truncated]
```

3. サービスを、許可されたサービスに追加します。

```
~]# firewall-cmd --add-service=<service-name>
```

4. 新しい設定を永続化します。

```
~]# firewall-cmd --runtime-to-permanent
```

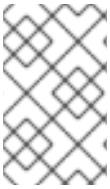
5.6.4. GUI を使用して事前定義サービスでトラフィックを制御

事前定義したサービスまたはカスタムサービスを有効または無効にするには、firewall-config ツールを起動し、サービスを設定するネットワークゾーンを選択します。Services タブを選択し、信頼するサービスタイプのチェックボックスを選択します。サービスをブロックする場合は、チェックボックスをオフにします。

サービスを編集するには、firewall-config ツールを起動し、Configuration というラベルの付いたメニューから Permanent を選択します。Services ウィンドウの下部に、追加のアイコンとメニューボタンが表示されます。設定するサービスを選択します。

ポート 、 プロトコル、および ソースポートタブ で は、選択したサービスのポート、プロトコル、

およびソースポートの追加、変更、および削除が可能です。modules タブは、Netfilter ヘルパーモジュールを設定するためのものです。Destination タブでは、特定の宛先アドレスおよびインターネットプロトコル(IPv4 または IPv6)へのトラフィックを制限できます。



注記

Runtime モードでは、サービス設定を変更できません。

5.6.5. 新しいサービスの追加

サービスは、グラフィカルな firewall-config ツール、firewall-cmd、および firewall-offline-cmd を使用して追加および削除できます。または、/etc/firewalld/services/ にある XML ファイルを編集することもできます。ユーザーがサービスを追加または変更しないと、対応する XML ファイルは /etc/firewalld/services/ にはありません。サービスを追加または変更する場合は、/usr/lib/firewalld/services/ ファイルをテンプレートとして使用できます。

ターミナルで新しいサービスを追加するには、firewall-cmd を使用するか、firewalld がアクティブな場合は firewall-offline-cmd を使用します。以下のコマンドを実行して、新規で空のサービスを追加します。

```
~]$ firewall-cmd --new-service=service-name
```

ローカルファイルを使用して新規サービスを追加するには、次のコマンドを使用します。

```
~]$ firewall-cmd --new-service-from-file=service-name.xml
```

追加オプション **--name=service-name** を指定して、サービス名を変更できます。

サービス設定を変更したらすぐに、サービスの更新されたコピーが /etc/firewalld/services/ に配置されます。

root で以下のコマンドを実行して、サービスを手動でコピーできます。

```
~]# cp /usr/lib/firewalld/services/service-name.xml /etc/firewalld/services/service-name.xml
```

firewalld は、最初に /usr/lib/firewalld/services からファイルを読み込みます。ファイルが /etc/firewalld/services に配置されており、ファイルが有効な場合は、/usr/lib/firewalld/services から一致するファイルを上書きします。/usr/lib/firewalld/services の上書きファイル

は、/etc/firewalld/services 内の一致するファイルが削除されるとすぐに、またはサービスのデフォルトを読み込むように firewalld が求められた場合に使用されます。これに該当するのは永続環境のみです。ランタイム環境でフォールバックさせるには、再読み込みが必要です。

5.6.6. CLI を使用したポートの制御

ポートは、オペレーティングシステムが、ネットワークトラフィックを受信し、区別し、システムサービスに従って転送する論理デバイスです。これは、通常、ポートをリッスンするデーモンにより示されますが、このポートに入るトラフィックを待ちます。

通常、システムサービスは、サービスに予約されている標準ポートでリッスンします。httpd デーモンは、たとえばポート 80 をリッスンします。ただし、デフォルトでは、システム管理者は、セキュリティーを強化するため、またはその他の理由により、別のポートをリッスンするようにデーモンを設定します。

ポートを開く

開かれたポートを介して、システムが外部からアクセスできます。これはセキュリティーリスクでもあります。一般的に、ポートを閉じたままにし、特定サービスに要求される場合に限り開きます。

現在のゾーンで開かれたポートの一覧を表示するには、以下を行います。

1. *許可されているポートの一覧を表示します。*

```
~]# firewall-cmd --list-ports
```

2. *許可されているポートにポートを追加して、着信トラフィックに対してそのポートを開きます。*

```
~]# firewall-cmd --add-port=port-number/port-type
```

3. *新しい設定を永続化します。*

```
~]# firewall-cmd --runtime-to-permanent
```

ポートタイプは、tcp、udp、sctp、または dccp のいずれかです。このタイプは、ネットワーク接続の種類と一致させる必要があります。

ポートを閉じる

開いているポートが必要なくなったら、**firewalld** でそのポートを閉じます。ポートをそのままにするとセキュリティーリスクとなるため、使用されなくなったらすぐに不要なポートを閉じることが強く推奨されます。

ポートを閉じるには、許可されているポートの一覧からそれを削除します。

1. 許可されているポートの一覧を表示します。

```
~]# firewall-cmd --list-ports
[WARNING]
====
This command will only give you a list of ports that have been opened as ports. You will not be able to see any open ports that have been opened as a service. Therefore, you should consider using the --list-all option instead of --list-ports.
====
```

2. 許可されているポートからポートを削除し、着信トラフィックに対して閉じます。

```
~]# firewall-cmd --remove-port=port-number/port-type
```

3. 新しい設定を永続化します。

```
~]# firewall-cmd --runtime-to-permanent
```

5.6.7. GUI を使用してポートを開く

特定のポートへのファイアウォールを通過するトラフィックを許可するには、**firewall-config** ツールを起動して、設定を変更するネットワークゾーンを選択します。右側の **Ports** タブを選択し、**Add** ボタンをクリックします。**Port and Protocol** ウィンドウが開きます。

許可するポート番号またはポートの範囲を入力します。一覧から **tcp** または **udp** を選択します。

5.6.8. GUI を使用してプロトコルを使用したトラフィックの制御

特定のプロトコルを使用してファイアウォールを通過するトラフィックを許可するには、**firewall-config** ツールを起動し、設定を変更するネットワークゾーンを選択します。右側の **Protocols** タブを

選択し、Add ボタンをクリックします。Protocol ウィンドウが開きます。

一覧からプロトコルを選択するか、Other Protocol チェックボックスを選択し、フィールドにプロトコルを入力します。

5.6.9. GUI を使用してソースポートを開く

特定のポートからのトラフィックがファイアウォールを通過できるようにするには、firewall-config ツールを起動して、設定を変更するネットワークゾーンを選択します。右側の Source Port タブを選択し、Add ボタンをクリックします。Source Port ウィンドウが開きます。

許可するポート番号またはポートの範囲を入力します。一覧から tcp または udp を選択します。

5.7. ゾーンの使用

ゾーンは、着信トラフィックをより透過的に管理する概念を表しています。ゾーンはネットワークインターフェイスに接続されているか、ソースアドレスの範囲に割り当てられます。各ゾーンは個別にファイアウォールルールを管理しますが、これにより、複雑なファイアウォール設定を定義してトラフィックに割り当てることができます。

5.7.1. ゾーンの一覧

システムで利用可能なゾーンを確認するには、次のコマンドを実行します。

```
~]# firewall-cmd --get-zones
```

firewall-cmd --get-zones コマンドは、システムで利用可能なすべてのゾーンを表示しますが、特定のゾーンの詳細は表示されません。

すべてのゾーンで詳細情報を表示する場合は、次のコマンドを実行します。

```
~]# firewall-cmd --list-all-zones
```

特定ゾーンに関する詳細情報を表示する場合は、次のコマンドを実行します。

```
~]# firewall-cmd --zone=zone-name --list-all
```

5.7.2. 特定ゾーンの firewalld 設定の変更

「CLI を使用して事前定義されたサービスでトラフィックの制御」および「CLI を使用したポートの制御」は、現在作業中のゾーンの範囲にサービスを追加するか、またはゾーンの範囲にあるポートを修正する方法を説明します。別のゾーンへのルールの設定が必要になる場合もあります。

別のゾーンに指定するには、**--zone=zone-name** オプションを使用します。たとえば、**public** ゾーンで **SSH** サービスを許可するには、次のコマンドを実行します。

```
~]# firewall-cmd --add-service=ssh --zone=public
```

5.7.3. デフォルトゾーンの変更

システム管理者は、設定ファイルのネットワークインターフェイスにゾーンを割り当てます。特定のゾーンに割り当てられないインターフェイスは、デフォルトゾーンに割り当てられます。**firewalld** サービスを再起動するたびに、**firewalld** はデフォルトゾーンの設定を読み込み、アクティブにします。

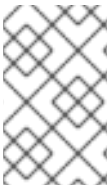
デフォルトゾーンを設定するには、以下を行います。

1. 現在のデフォルトゾーンを表示します。

```
~]# firewall-cmd --get-default-zone
```

2. 新しいデフォルトゾーンを設定します。

```
~]# firewall-cmd --set-default-zone zone-name
```



注記

この手順では、**--permanent** オプションを使用しなくても、設定は永続化します。

5.7.4. ゾーンへのネットワークインターフェイスの割り当て

複数のゾーンに複数のルールセットを定義して、使用されているインターフェイスのゾーンを変更することで、迅速に設定を変更できます。各インターフェイスに特定のゾーンを設定して、そのゾーンを通過するトラフィックを設定できます。

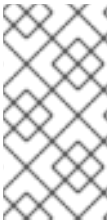
特定インターフェイスにゾーンを割り当てるには、以下を行います。

1. アクティブゾーン、およびそのゾーンに割り当てられているインターフェイスを一覧表示します。

```
~]# firewall-cmd --get-active-zones
```

2. 別のゾーンにインターフェイスを割り当てます。

```
~]# firewall-cmd --zone=zone-name --change-interface=<interface-name>
```



注記

再起動後も設定を持続させる **--permanent** オプションを使用する必要はありません。新しいデフォルトゾーンを設定すると、設定は永続化されます。

5.7.5. ネットワーク接続にデフォルトゾーンの割り当て

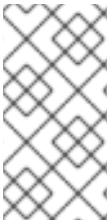
NetworkManager で接続を管理する場合は、その接続が使用するゾーンを認識する必要があります。すべてのネットワーク接続にゾーンを指定できます。これにより、ポータブルデバイスを使用したコンピューターの場所に従って、様々なファイアウォールを柔軟に設定できるようになります。したがって、ゾーンおよび設定には、会社または自宅など、様々な場所を指定できます。

インターネット接続にデフォルトゾーンを設定するには、**NetworkManager GUI** を使用するか、**/etc/sysconfig/network-scripts/ifcfg-connection-name** ファイルを編集し、この接続にゾーンを割り当てる行を追加します。

```
ZONE=zone-name
```

5.7.6. 新しいゾーンの作成

カスタムゾーンを使用するには、新しいゾーンを作成したり、事前定義したゾーンなどを使用したりします。



注記

新しいゾーンには --permanent オプションが必要となり、このオプションがなければコマンドは動作しません。

1. *新しいゾーンを作成します。*

```
~]# firewall-cmd --permanent --new-zone=zone-name
```

2. *新しいゾーンをリロードします。*

```
~]# firewall-cmd --reload
```

3. *作成したゾーンが永続設定に追加されたかどうかを確認します。*

```
~]# firewall-cmd --get-zones
```

4. *新しい設定を永続化します。*

```
~]# firewall-cmd --runtime-to-permanent
```

5.7.7. 設定ファイルを使用した新しいゾーンの作成

また、ゾーンの設定ファイル を使用してゾーンを作成できます。このアプローチは、新しいゾーンを作成する必要がある場合に、別のゾーンの設定を変更して利用する場合に便利です。

firewalld ゾーン設定ファイルには、ゾーンの情報が含まれます。これは、XML ファイル形式で、ゾーンの説明、サービス、ポート、プロトコル、icmp-block、マスカレード、転送ポート、およびリッチ言語ルールです。ファイル名は zone-name.xml である必要があります。ここでは、zone-name の長さは 17 文字に制限されます。ゾーンの設定ファイルは、/usr/lib/firewalld/zones/ ディレクトリーおよび /etc/firewalld/zones/ ディレクトリーにあります。

以下の例は、TCP プロトコルと UDP プロトコルの両方に、1 つのサービス(SSH)と 1 つのポート範囲を許可する設定を示しています。

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
```

```
  <short>My zone</short>
  <description>Here you can describe the characteristic features of the zone.</description>
  <service name="ssh"/>
  <port port="1025-65535" protocol="tcp"/>
  <port port="1025-65535" protocol="udp"/>
</zone>
```

*そのゾーンの設定を変更するには、セクションを追加または削除して、ポート、転送ポート、サービスなどを追加します。詳細は、**firewalld.zone** の **man** ページを参照してください。*

5.7.8. 着信トラフィックにデフォルトの動作を設定するゾーンターゲットの使用

*すべてのゾーンに対して、特に指定されていない着信トラフィックを処理するデフォルト動作を設定できます。そのような動作は、ゾーンのターゲットを設定することで定義されます。デフォルトの、**ACCEPT**、**REJECT**、および **DROP** の **3** つのオプションがあります。ターゲットを **ACCEPT** に設定すると、特定のルールで無効にされたパケット以外の着信パケットをすべて受け入れます。ターゲットを **REJECT** または **DROP** に設定すると、特定のルールで許可したパケット以外の着信パケットをすべて無効にします。パケットが拒否されるとソースマシンに通知されますが、パケットが破棄される時は情報が送信されません。*

ゾーンにターゲットを設定するには、以下を行います。

1. *特定ゾーンに対する情報を一覧表示して、デフォルトゾーンを確認します。*

```
~]$ firewall-cmd --zone=zone-name --list-all
```

2. *ゾーンに新しいターゲットを設定します。*

```
~]# firewall-cmd --zone=zone-name --set-target=<default|ACCEPT|REJECT|DROP>
```

5.8. ゾーンを使用し、ソースに応じた着信トラフィックの管理

ゾーンを使用して、そのソースに基づいて着信トラフィックを管理するゾーンを使用できます。これにより、着信トラフィックを分類し、複数のゾーンに向け、トラフィックにより到達できるサービスを許可または拒否できます。

*ソースをゾーンに追加する場合は、ゾーンがアクティブになり、そのソースからの着信トラフィックは、それを介して行われます。各ゾーンに異なる設定を指定できますが、それは指定したソースから順次トラフィックに適用されます。ネットワークインターフェイスが **1** つしかない場合でも、複数のゾーンを使用できます。*

5.8.1. ソースの追加

着信トラフィックを特定のソースに転送する場合は、そのゾーンにソースを追加します。ソースは、CIDR (Classless Inter-domain Routing) 表記法の IP アドレスまたは IP マスクになります。

1. 現在のゾーンにソースを設定するには、次のコマンドを実行します。

```
~]# firewall-cmd --add-source=<source>
```

2. 特定ゾーンのソース IP アドレスを設定するには、次のコマンドを実行します。

```
~]# firewall-cmd --zone=zone-name --add-source=<source>
```

以下の手順では、trusted ゾーンで 192.168.2.15 からのすべての着信トラフィックを許可します。

1. 利用可能なゾーンの一覧を表示します。

```
~]# firewall-cmd --get-zones
```

2. 永続化モードで、信頼ゾーンにソース IP を追加します。

```
~]# firewall-cmd --zone=trusted --add-source=192.168.2.15
```

3. 新しい設定を永続化します。

```
~]# firewall-cmd --runtime-to-permanent
```

5.8.2. ソースの削除

ゾーンからソースを削除すると、そのゾーンからのトラフィックを遮断します。

1. 必要なゾーンに対して許可されているソースの一覧を表示します。

```
~]# firewall-cmd --zone=zone-name --list-sources
```

2. ゾーンからソースを永続的に削除します。

```
~]# firewall-cmd --zone=zone-name --remove-source=<source>
```

3. 新しい設定を永続化します。

```
~]# firewall-cmd --runtime-to-permanent
```

5.8.3. ソースポートの追加

発信源となるポートに基づいたトラフィックの分類を有効にするには、**--add-source-port** オプションを使用してソースポートを指定します。**--add-source** オプションと組み合わせて、トラフィックを特定の IP アドレスまたは IP 範囲に制限できます。

ソースポートを追加するには、次のコマンドを実行します。

```
~]# firewall-cmd --zone=zone-name --add-source-port=<port-name>/<tcp|udp|sctp|dccp>
```

5.8.4. ソースポートの削除

ソースポートを削除して、送信元ポートに基づいてトラフィックの分類を無効にします。

ソースポートを削除するには、次のコマンドを実行します。

```
~]# firewall-cmd --zone=zone-name --remove-source-port=<port-name>/<tcp|udp|sctp|dccp>
```

5.8.5. ゾーンおよびソースを使用して特定ドメインのみに対してサービスの許可

特定のネットワークからのトラフィックを許可して、マシンのサービスを使用するには、ゾーンおよびソースを使用します。以下の手順では、他のトラフィックがブロックされている間、**192.0.2.0/24** ネットワークからの **HTTP** トラフィックのみを許可します。

警告

このシナリオを設定する場合は、デフォルト のターゲットを持つゾーンを使用します。ターゲットが ACCEPT に設定されているゾーンを使用することは、192.0.2.0/24 からのトラフィックではすべてのネットワーク接続が許可されるため、セキュリティー上のリスクがあります。

1. **利用可能なゾーンの一覧を表示します。**

```
~]# firewall-cmd --get-zones
block dmz drop external home internal public trusted work
```

2. **IP 範囲を internal ゾーンに追加し、ソースから発信されるトラフィックを ゾーン経由でルーティングします。**

```
~]# firewall-cmd --zone=internal --add-source=192.0.2.0/24
```

3. **http サービスを internal ゾーンに追加します。**

```
~]# firewall-cmd --zone=internal --add-service=http
```

4. **新しい設定を永続化します。**

```
~]# firewall-cmd --runtime-to-permanent
```

5. **internal ゾーンがアクティブで、サービスが許可されていることを確認します。**

```
~]# firewall-cmd --zone=internal --list-all
internal (active)
  target: default
  icmp-block-inversion: no
  interfaces:
  sources: 192.0.2.0/24
  services: dhcpv6-client mdns samba-client ssh http
  ...
```

5.8.6. プロトコルに基づいてゾーンが許可したトラフィックの設定

プロトコルに基づいて、ゾーンが着信トラフィックを許可できます。指定したプロトコルを使用したすべてのトラフィックがゾーンにより許可されていますが、そこにさらにルールおよびフィルターリングを適用できます。

ゾーンへのプロトコルの追加

特定ゾーンへプロトコルを追加すると、このゾーンが許可するこのプロトコルを使用するすべてのトラフィックを許可します。

プロトコルをゾーンに追加するには、次のコマンドを実行します。

```
~]# firewall-cmd --zone=zone-name --add-protocol=port-name/tcp|udp|sctp|dccp|igmp
```

注記

マルチキャストトラフィックを受信するには、**--add-protocol** オプションを指定して **igmp** 値を使用します。

ゾーンからプロトコルの削除

特定ゾーンからプロトコルを削除するには、ゾーンにより、このプロトコルに基づいたすべてのトラフィックの許可を停止します。

ゾーンからプロトコルを削除するには、次のコマンドを削除します。

```
~]# firewall-cmd --zone=zone-name --remove-protocol=port-name/tcp|udp|sctp|dccp|igmp
```

5.9. ポート転送

firewalld を使用すると、システムで特定のポートに到達する着信トラフィックが、選択した別の内部ポートまたは別のマシンの外部ポートに配信されるようにポートのリダイレクトを設定できます。

5.9.1. リダイレクトするポートの追加

あるポートから別のポートまたは別のアドレスにトラフィックをリダイレクトする前に、パケットが到達するポート、使用されるプロトコル、リダイレクト先の **3** つを確認しておく必要があります。

ポートを別のポートにリダイレクトする場合は、次のコマンドを実行します。

```
~]# firewall-cmd --add-forward-port=port=port-number:proto=tcp|udp|sctp|dccp:toport=port-number
```

別の IP アドレスで、別のポートにポートをリダイレクトする場合は、次のコマンドを実行します。

1. 転送するポートを追加します。

```
~]# firewall-cmd --add-forward-port=port=port-number:proto=tcp|udp:toport=port-number:toaddr=IP
```

2. マスカレードを有効にします。

```
~]# firewall-cmd --add-masquerade
```

例5.1 同一マシンで TCP ポート 80 からポート 88 へのリダイレクト

ポートをリダイレクトするには、以下を行います。

1. TCP トラフィックに対して、ポート 80 からポート 88 へリダイレクトします。

```
~]# firewall-cmd --add-forward-port=port=80:proto=tcp:toport=88
```

2. 新しい設定を永続化します。

```
~]# firewall-cmd --runtime-to-permanent
```

3. そのポートがリダイレクトされていることを確認します。

```
~]# firewall-cmd --list-all
```

5.9.2. リダイレクトしているポートの削除

リダイレクトしているポートを削除するには、次のコマンドを実行します。

```
~]# firewall-cmd --remove-forward-port=port=port-number:proto=<tcp|udp>:toport=port-number:toaddr=<IP>
```

別のアドレスにリダイレクトした転送ポートを削除するには、以下を実行します。

1. 転送したポートを削除するには、以下を行います。

```
~]# firewall-cmd --remove-forward-port=port=port-number:proto=<tcp|udp>:toport=port-number:toaddr=<IP>
```

2. マスカレードを無効にするには、次のコマンドを実行します。

```
~]# firewall-cmd --remove-masquerade
```

注記

この方法を使用するポートのリダイレクトは、IPv4 ベースのトラフィックでのみ機能します。IPv6 リダイレクト設定には、リッチルールを使用する必要があります。詳細は、「リッチランゲージ構文を使用した複雑なファイアウォールルールの設定」 を参照してください。

外部システムにリダイレクトするには、マスカレードを有効にする必要があります。詳細は、「IP アドレスのマスカレードの設定」 を参照してください。

例5.2 同じマシンで TCP ポート 88 に転送されるポート 80 の削除

ポートのリダイレクトを削除するには、以下を行います。

1. リダイレクトしたポートの一覧を表示します。

```
~]# firewall-cmd --list-forward-ports
port=80:proto=tcp:toport=88:toaddr=
```

2. ファイアウォールからリダイレクトしたポートを削除します。

```
~]# firewall-cmd --remove-forward-port=port=80:proto=tcp:toport=88:toaddr=
```

3. 新しい設定を永続化します。

```
~]# firewall-cmd --runtime-to-permanent
```

5.10. IP アドレスのマスカレードの設定

IP マスカレードとは、あるコンピューターがネットワークの IP ゲートウェイとして機能するプロセスです。マスカレードの場合、ゲートウェイは常に発信インターフェイスの IP を動的に検索し、パケット内のソースアドレスをこのアドレスに置き換えます。

発信インターフェイスの IP が変更される可能性がある場合は、マスカレードを使用します。マスカレードの典型的な使用例は、ルーターがインターネット上でルーティングされないプライベート IP アドレスを、ルーター上の発信インターフェイスのパブリック動的 IP アドレスに置き換える場合です。

IP マスカレードが有効になっているかどうかを確認するには（例： external ゾーンの場合）、root で以下のコマンドを入力します。

```
~]# firewall-cmd --zone=external --query-masquerade
```

このコマンドは、有効な場合は yes と出力され、終了ステータスは 0 になります。そうでなければ no と出力され、終了ステータスは 1 になります。zone を省略すると、デフォルトのゾーンが使用されます。

IP マスカレードを有効にするには、root で以下のコマンドを入力します。

```
~]# firewall-cmd --zone=external --add-masquerade
```

この設定を永続的にするには、--permanent オプションを追加してコマンドを繰り返します。

IP マスカレードを無効にするには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --zone=external --remove-masquerade
```

この設定を永続的にするには、**--permanent** オプションを追加してコマンドを繰り返します。

詳細は以下を参照してください。

- 「異なる NAT タイプ: マスカレード、ソース NAT、宛先 NAT、リダイレクト」
- 「nftables を使用したマスカレードの設定」

5.11. ICMP リクエストの管理

Internet Control Message Protocol (ICMP)は、接続の問題（要求されたサービスが利用できないなど）を示すエラーメッセージと操作情報を送信するために、さまざまなネットワークデバイスによって使用されるサポートプロトコルです。ICMP は、システム間でデータを交換するために使用されないため、TCP や UDP などのトランスポートプロトコルとは異なります。

残念ながら、ICMP メッセージ（特に echo-request および echo-reply ）を使用して、ネットワークに関する情報を示し、さまざまな不正行為についてこのような情報を誤用できます。したがって、firewalld は、ネットワーク情報を保護するための ICMP リクエストをブロックすることを可能にします。

5.11.1. ICMP リクエストの一覧表示

ICMP リクエストは、/usr/lib/firewalld/icmptypes/ ディレクトリーにある個別の XML ファイルで説明されています。リクエストの説明は、このファイルを参照してください。firewall-cmd コマンドは、ICMP リクエストの操作を制御します。

利用可能な ICMP タイプの一覧を表示するには、以下のコマンドを実行します。

```
~]# firewall-cmd --get-icmptypes
```

ICMP リクエストは、IPv4、IPv6、またはその両方のプロトコルで使用できます。ICMP リクエストが使用されているプロトコルを表示するには、次のコマンドを実行します。

```
~]# firewall-cmd --info-icmptype=<icmptype>
```

ICMP リクエストのステータスは、リクエストが現在ブロックされている場合は yes、ブロックされていない場合は no と表示されます。ICMP リクエストが現在ブロックされているかどうかを確認するには、次のコマンドを実行します。

```
~]# firewall-cmd --query-icmp-block=<icmptype>
```

5.11.2. ICMP リクエストのブロックまたはブロック解除

サーバーが ICMP リクエストをブロックすると、通常の情報は提供されません。ただし、情報が全く提供されないというわけではありません。クライアントは、特定の ICMP リクエストがブロックされている（拒否される）情報を受け取ります。ICMP リクエストをブロックすると、特に IPv6 トラフィックで通信の問題が発生する可能性があるため、慎重に検討する必要があります。

ICMP リクエストが現在ブロックされているかどうかを確認するには、次のコマンドを実行します。

```
~]# firewall-cmd --query-icmp-block=<icmptype>
```

ICMP リクエストをブロックするには、次のコマンドを実行します。

```
~]# firewall-cmd --add-icmp-block=<icmptype>
```

ICMP リクエストのブロックを削除するには、次のコマンドを実行します。

```
~]# firewall-cmd --remove-icmp-block=<icmptype>
```

5.11.3. 情報を提供せずに ICMP リクエストのブロック

通常、ICMP リクエストをブロックすると、クライアントはブロックしていることを認識します。したがって、ライブの IP アドレスを傍受している潜在的な攻撃者は、IP アドレスがオンラインであることを確認できます。この情報を完全に非表示にするには、ICMP リクエストをすべて破棄する必要があります。

すべての ICMP リクエストをブロックして破棄するには、次のコマンドを実行します。

1. ゾーンのターゲットを DROP に設定します。

```
~]# firewall-cmd --set-target=DROP
```

2. 新しい設定を永続化します。

```
~]# firewall-cmd --runtime-to-permanent
```

ICMP リクエストを含むすべてのトラフィックがドロップされるようになりました（明示的に許可したトラフィックを除く）。

特定の ICMP リクエストをブロックして破棄し、その他を許可するには、次のコマンドを実行します。

1. ゾーンのターゲットを DROP に設定します。

```
~]# firewall-cmd --set-target=DROP
```

2. ICMP ブロックの反転を追加して、すべての ICMP リクエストを一度にブロックします。

```
~]# firewall-cmd --add-icmp-block-inversion
```

3. 許可する ICMP リクエストに ICMP ブロックを追加します。

```
~]# firewall-cmd --add-icmp-block=<icmptype>
```

4. 新しい設定を永続化します。

```
~]# firewall-cmd --runtime-to-permanent
```

ブロックの反転 は ICMP リクエストブロックの設定を反転するため、ブロックされていないすべてのリクエストはブロックされます。ブロックされているものはブロックされません。したがって、リクエストのブロックを解除する必要がある場合は、ブロックコマンドを使用してください。

これを完全に許容できる設定に戻すには、以下を行います。

1. ゾーンのターゲットを default または ACCEPT に設定します。

-

```
~]# firewall-cmd --set-target=default
```

2. ICMP リクエストに追加したすべてのブロックを削除します。

```
~]# firewall-cmd --remove-icmp-block=<icmptype>
```

3. ICMP ブロックの反転を削除します。

```
~]# firewall-cmd --remove-icmp-block-inversion
```

4. 新しい設定を永続化します。

```
~]# firewall-cmd --runtime-to-permanent
```

5.11.4. GUI を使用した ICMP フィルターの設定

ICMP フィルターを有効または無効にするには、firewall-config ツールを起動し、メッセージがフィルターされるネットワークゾーンを選択します。ICMP フィルター タブを 選択し、フィルターリングする ICMP メッセージの各タイプのチェックボックスを選択します。フィルターを無効にするには、チェックボックスの選択を外します。これは方向ごとに設定され、デフォルトではすべてが許可されます。

ICMP フィルター の反転を有効にするには、右側の Invert Filter チェックボックスをクリックします。マークされた ICMP タイプのみが許可され、その他はすべて拒否されます。DROP ターゲットを使用するゾーンでは破棄されます。

5.12. FIREWALLDを使用した IP セットの設定および制御

firewalld で対応している IP セットタイプの一覧を表示するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --get-ipset-types
hash:ip hash:ip,mark hash:ip,port hash:ip,port,ip hash:ip,port,net hash:mac hash:net hash:net,iface
hash:net,net hash:net,port hash:net,port,net
```

5.12.1. コマンドラインクライアントを使用した IP セットオプションの設定

IP セットは、firewalld ゾーンでソースとして使用でき、リッチルールのソースとして使用できます。Red Hat Enterprise Linux 7 では、直接ルールで firewalld で作成された IP セットを使用すること

が推奨されます。

永続的な環境で firewalld が認識している IP セットを一覧表示するには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --permanent --get-ipsets
```

新しい IP セットを追加するには、root で永続環境を使用し、以下のコマンドを使用します。

```
~]# firewall-cmd --permanent --new-ipset=test --type=hash:net
success
```

上記のコマンドは、名前 test と IPv4 の hash:net タイプで新しい IP セットを作成します。IPv6 で使用する IP セットを作成するには、--option=family=inet6 オプションを追加します。ランタイム環境で新しい設定を有効にするには、firewalld をリロードします。root で以下のコマンドを実行して、新しい IP セットを一覧表示します。

```
~]# firewall-cmd --permanent --get-ipsets
test
```

IP セットに関する詳細情報を取得するには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --permanent --info-ipset=test
test
type: hash:net
options:
entries:
```

この時点では IP セットにエントリーがありません。IP セットにエントリーを追加するには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --permanent --ipset=test --add-entry=192.168.0.1
success
```

上記のコマンドは、IP アドレス 192.168.0.1 を IP セットに追加します。IP セットの現在のエントリー一覧を取得するには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
```

IP アドレスの一覧を含むファイルを生成します。以下に例を示します。

```
~]# cat > iplist.txt <<EOL
192.168.0.2
192.168.0.3
192.168.1.0/24
192.168.2.254
EOL
```

IP セットの IP アドレスの一覧が含まれるファイルには、行ごとにエントリーが含まれている必要があります。ハッシュ、セミコロン、また空の行から始まる行は無視されます。

iplist.txt ファイルからアドレスを追加するには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --permanent --ipset=test --add-entries-from-file=iplist.txt
success
```

IP セットの拡張エントリー一覧を表示するには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
192.168.0.2
192.168.0.3
192.168.1.0/24
192.168.2.254
```

IP セットからアドレスを削除し、更新されたエントリー一覧を確認するには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --permanent --ipset=test --remove-entries-from-file=iplist.txt
success
~]# firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
```

IP セットをゾーンへのソースとして追加し、ゾーンを使用して、IP セットに記載されるアドレスから受信するすべてのトラフィックを処理します。たとえば、テスト IP セットをソースとして drop ゾーンに追加し、IP セットの テスト に一覧表示されているすべてのエントリーから送られるすべてのパケットをドロップするには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --permanent --zone=drop --add-source=ipset:test
success
```

ソースの ipset: 接頭辞は、ソースが IP セットであり、IP アドレスまたはアドレス範囲ではない firewalld を示します。

IP セットの作成および削除は、永続環境に限定されますが、その他の IP セットオプションは、--permanent オプションを使用しないランタイム環境で使用できます。

5.12.2. IP セットのカスタムサービスの設定

firewalld が起動する前に IP セット構造を作成してロードするようにカスタムサービスを設定するには、以下を実行します。

1. root で実行されているエディターを使用して、以下のようにファイルを作成します。

```
~]# vi /etc/systemd/system/ipset_name.service
[Unit]
Description=ipset_name
Before=firewalld.service

[Service]
Type=oneshot
RemainAfterExit=yes
ExecStart=/usr/local/bin/ipset_name.sh start
ExecStop=/usr/local/bin/ipset_name.sh stop

[Install]
WantedBy=basic.target
```

2. firewalld で IP セットを永続的に使用します。

```
~]# vi /etc/firewalld/direct.xml
<?xml version="1.0" encoding="utf-8"?>
<direct>
 <rule ipv="ipv4" table="filter" chain="INPUT" priority="0">-m set --match-set
<replaceable>ipset_name</replaceable> src -j DROP</rule>
</direct>
```

3. 変更を有効にするには、firewalld のリロードが必要です。

```
~]# firewall-cmd --reload
```

これにより、状態情報を失うことなくファイアウォールがリロードされます (TCP セッションは終了しません) が、リロード中にサービスが中断する可能性があります。

警告

Red Hat は、firewalld で管理されない IP セットを使用することは推奨していません。このような IP セットを使用すると、そのセットを参照する永続的なダイレクトルールが必要で、IP セットを作成するカスタムサービスを追加する必要があります。このサービスは、firewalld を起動する前に起動する必要があります。起動しないと、firewalld はこのセットを使用してダイレクトルールを追加できません。/etc/firewalld/direct.xml ファイルを使用して、永続的なダイレクトルールを追加できます。

5.13. IPTABLESを使用した IP セットの設定および制御

firewalld サービスと iptables （および ip6tables）サービスの基本的な相違点は次のとおりです。

- iptables サービス は設定を /etc/sysconfig/iptables および /etc/sysconfig/ip6tables に保存しますが、firewalld は設定を /usr/lib/firewalld/ および /etc/firewalld/ のさまざまな XML ファイルに保存します。デフォルトでは Red Hat Enterprise Linux に firewalld がインストールされているため、/etc/sysconfig/iptables ファイルが存在しないことに注意してください。

- iptables サービス では、すべての変更がすべての古いルールをフラッシュし、/etc/sysconfig/iptables からすべての新しいルールを読み取ることを意味しますが、firewalld ではすべてのルールを再作成することはありません。違いのみが適用されます。その結果、firewalld は、既存の接続を失うことなく、ランタイム時に設定を変更できます。

いずれも iptables ツール を使用してカーネルパケットフィルターと通信します。

firewalld の代わりに iptables サービスおよび ip6tables サービスを使用するには、最初に root で以下のコマンドを実行して firewalld を無効にします。

```
~]# systemctl disable firewalld
~]# systemctl stop firewalld
```

次に、root で以下のコマンドを入力して、iptables-services パッケージをインストールします。

```
~]# yum install iptables-services
```

iptables-services パッケージには、iptables サービスと ip6tables サービスが含まれます。

次に、iptables サービスおよび ip6tables サービスを起動するには、root で以下のコマンドを入力します。

```
~]# systemctl start iptables
~]# systemctl start ip6tables
```

システムが起動するたびにサービスを開始できるようにするには、次のコマンドを入力します。

```
~]# systemctl enable iptables
~]# systemctl enable ip6tables
```

ipset ユーティリティーは、Linux カーネルで IP セット を管理するために使用されます。IP セットは、IP アドレス、ポート番号、IP と MAC アドレスのペア、または IP アドレスとポート番号のペアを格納するためのフレームワークです。セットは、セットが非常に大きい場合でも、セットに対して非常に高速なマッチングを行うことができるようにインデックスが付けられます。IP セットは、よりシンプルで管理しやすい設定を可能にし、iptables を使用する際のパフォーマンス上の利点を提供します。iptables が一致し、セットを参照するターゲットは、カーネル内の指定されたセットを保護する参照を作成します。セットを指す参照が 1 つある間は、セットを破棄することはできません。

ipset を使用すると、以下のような iptables コマンドをセットに置き換えることができます。

```
~]# iptables -A INPUT -s 10.0.0.0/8 -j DROP
~]# iptables -A INPUT -s 172.16.0.0/12 -j DROP
~]# iptables -A INPUT -s 192.168.0.0/16 -j DROP
```

セットは、以下のように作成されます。

```
~]# ipset create my-block-set hash:net
~]# ipset add my-block-set 10.0.0.0/8
~]# ipset add my-block-set 172.16.0.0/12
~]# ipset add my-block-set 192.168.0.0/16
```

次に、セットは以下のように **iptables** コマンドで参照されます。

```
~]# iptables -A INPUT -m set --set my-block-set src -j DROP
```

セットを複数回使用すると、設定時間が節約されます。セットに多くのエントリーが含まれている場合、処理時間を短縮することができます。

5.14. ダイレクトインターフェイスの使用

firewall-cmd ツールで **--direct** オプションを使用すると、ランタイム時にチェーンを追加および削除できます。ここではいくつかの例を紹介します。詳細は、**firewall-cmd (1) man** ページを参照してください。

iptables に精通していない場合は、ファイアウォールで誤って違反している可能性があるため、直接インターフェイスを使用することは危険です。

ダイレクトインターフェイスモードは、サービスやアプリケーションが実行時に特定のファイアウォールルールを追加することを目的としています。ルールは、**firewall-cmd --permanent --direct** コマンドを使用して **--permanent** オプションを追加するか、**/etc/firewalld/direct.xml** を変更することで、永続的にすることができます。**/etc/ firewalld/direct.xml** ファイルの詳細は、**firewalld.direct (5)** の **man** を参照してください。

5.14.1. ダイレクトインターフェイスを使用するルールの追加

「**IN_public_allow**」 チェーンにルールを追加するには、**root** で以下のコマンドを入力します。

```
~]# firewall-cmd --direct --add-rule ipv4 filter IN_public_allow \
      0 -m tcp -p tcp --dport 666 -j ACCEPT
```

設定を永続的にするために **--permanent** オプションを追加します。

5.14.2. ダイレクトインターフェイスを使用したルールの削除

「**IN_public_allow**」 チェーンからルールを削除するには、**root** で以下のコマンドを実行します。

```
~]# firewall-cmd --direct --remove-rule ipv4 filter IN_public_allow \
      0 -m tcp -p tcp --dport 666 -j ACCEPT
```

設定を永続的にするために --permanent オプションを追加します。

5.14.3. ダイレクトインターフェイスを使用したルールの一覧表示

「IN_public_allow」 チェーンのルールを一覧表示するには、root で以下のコマンドを入力します。

```
~]# firewall-cmd --direct --get-rules ipv4 filter IN_public_allow
```

このコマンド (--get-rules オプション) は、--add-rule オプションを使用して以前に追加されたルールのみを一覧表示することに注意してください。他の手段で追加された既存の iptables ルールは一覧表示されません。

5.15. リッチランゲージ構文を使用した複雑なファイアウォールルールの設定

「リッチ言語」 構文を使用すると、直接インターフェイス方式よりも理解しやすい方法で、複雑なファイアウォールルールを作成できます。さらに、設定を永続的にすることができます。言語は値を持つキーワードを使用し、iptables ルールの抽象表現です。ゾーンはこの言語を使って設定することができますが、現在の設定方法は引き続きサポートされます。

5.15.1. リッチ言語コマンドの形式

このセクションのすべてのコマンドは、root として実行する必要があります。ルールを追加するコマンドの書式は以下の通りです。

```
firewall-cmd [--zone=zone] --add-rich-rule='rule' [--timeout=timeval]
```

これにより、ゾーン ゾーン のリッチ言語ルール ルール が追加されます。このオプションは複数回指定できます。ゾーンを省略すると、デフォルトのゾーンが使用されます。タイムアウトを指定すると、指定された時間だけルールが有効になり、その後は自動的に削除されます。時間値の後に、s (秒)、m (分)、h (時) を続けて、時間の単位を指定することができる。デフォルトは秒です。

ルールの削除

```
firewall-cmd [--zone=zone] --remove-rich-rule='rule'
```

これにより、ゾーン ゾーン のリッチ言語ルール ルール が削除されます。このオプションは複数回指定できます。ゾーンを省略すると、デフォルトのゾーンが使用されます。

ルールが存在するかを確認するには、以下を実行します。

```
firewall-cmd [--zone=zone] --query-rich-rule='rule'
```

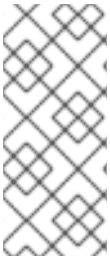
これにより、ゾーン ゾーン にリッチ言語ルール ルール が追加されているかどうかが返されます。このコマンドは、有効な場合は **yes** と出力され、終了ステータスは **0** になります。そうでなければ **no** と出力され、終了ステータスは **1** になります。ゾーンを省略すると、デフォルトのゾーンが使用されます。

ゾーン設定ファイルで使用されるリッチ言語表現の詳細については、**firewalld.zone(5)** の **man** ページを参照してください。

5.15.2. リッチルールの構造について

リッチルールコマンドの形式または構造は次のとおりです。

```
rule [family="rule family"]
  [ source [NOT] [address="address"] [mac="mac-address"] [ipset="ipset"] ]
  [ destination [NOT] address="address" ]
  [ element ]
  [ log [prefix="prefix text"] [level="log level"] [limit value="rate/duration"] ]
  [ audit ]
  [ action ]
```



注記

ファイル内のリッチルールの構造では、送信元アドレスと宛先アドレスのコマンドの意味を反転させるために **NOT** キーワードを使用していますが、コマンドラインでは **invert="true"** オプションが使用されています。

ルールは、特定のゾーンに関連付けられます。ゾーンには複数のルールを関連付けることができます。いくつかのルールが相互作用したり矛盾したりする場合、パケットにマッチする最初のルールが適用されます。

5.15.3. リッチルールのコマンドオプションについて

family

ルールファミリー(ipv4 または ipv6 のいずれか)を指定すると、ルールがそれぞれ IPv4 または IPv6 に制限されます。ルールファミリーが指定されていない場合は、IPv4 と IPv6 の両方にルールが追加されます。ルールで送信元アドレスまたは宛先アドレスが使用されている場合は、ルールファミリーを提供する必要があります。これは、ポートフォワーディングにも当てはまります。

ソースアドレスおよび宛先のアドレス

source

送信元アドレスを指定することにより、接続試行の発信元を送信元アドレスに制限できます。ソースアドレスまたはアドレス範囲は、IPv4 または IPv6 のマスクを持つ IP アドレスまたはネットワーク IP アドレスのいずれかです。IPv4 の場合、マスクはネットワークマスクまたは単純な番号になります。IPv6 の場合、マスクは単純な番号です。ホスト名の使用はサポートされていません。NOT キーワードを追加することで、source address コマンドの意味を反転させることができます。指定されたアドレス以外はすべて一致します。

ルールに ファミリー が指定されていない場合は、IPv4 および IPv6 に MAC アドレスと hash:mac タイプの IP セットを追加できます。他の IP セットは、ルールの family 設定と一致する必要があります。

destination

宛先アドレスを指定することにより、ターゲットを宛先アドレスに限定することができます。宛先アドレスは、IP アドレスまたはアドレス範囲の送信元アドレスと同じ構文を使用します。送信元アドレスと宛先アドレスの使用はオプションであり、すべての要素で宛先アドレスを使用できるわけではありません。これは、たとえば、サービスエントリーでの宛先アドレスの使用によって異なります。destination と action を組み合わせることができます。

要素

要素には、service、port、プロトコル、masquerade、icmp-block、forward-port 、および source-port の いずれかのタイプのみ を使用できます。

service

service 要素は、firewalld が提供するサービスの 1 つです。定義済みサービスの一覧を取得するには、次のコマンドを入力します。

```
~]$ firewall-cmd --get-services
```

サービスが宛先アドレスを提供する場合、ルール内の宛先アドレスと競合し、エラーが発生します。内部で宛先アドレスを使用するサービスは、ほとんどがマルチキャストを使用するサービスである。コマンドは以下の形式になります。

```
service name=service_name
```

-

port

port 要素には、単一のポート番号またはポート範囲（例：5060- 50 62）と、その後に tcp または udp などのプロトコルを使用できます。コマンドは以下の形式になります。

```
port port=number_or_range protocol=protocol
```

protocol

protocol 値は、プロトコル ID 番号またはプロトコル名のいずれかになります。許可される プロトコル エントリーについては、/etc/protocols を参照してください。コマンドは以下の形式になります。

```
protocol value=protocol_name_or_ID
```

icmp-block

このコマンドを使用して、1 つ以上の ICMP タイプをブロックします。ICMP タイプは、firewalld がサポートする ICMP タイプの 1 つです。サポートされている ICMP タイプの一覧を取得するには、次のコマンドを入力します。

```
~]$ firewall-cmd --get-icmptypes
```

ここではアクションの特定はできません。icmp-block はアクション reject を内部で使用します。コマンドは以下の形式になります。

```
icmp-block name=icmptype_name
```

masquerade

ルール内の IP マスカレードを有効にします。マスカレードをこのエリアに限定するために送信元アドレスを指定できますが、宛先アドレスは指定できません。ここではアクションの特定はできません。

forward-port

プロトコルが tcp または udp として指定されたローカルポートから、ローカルにある別のポー

ト、別のマシン、または別のマシンの別のポートにパケットを転送します。port と to-port は、単一のポート番号またはポート範囲のいずれかになります。宛先アドレスは、単純な IP アドレスです。ここではアクションの特定はできません。forward-port コマンドは、内部で accept のアクションを使用します。コマンドは以下の形式になります。

```
forward-port port=number_or_range protocol=protocol /
        to-port=number_or_range to-addr=address
```

source-port

パケットのソースポート、つまり接続を試みる際の発信元で使用されるポートに一致します。現在のマシンのポートに一致させるには、port 要素を使用します。source-port 要素は、単一のポート番号またはポート範囲（例：5060-5062）のいずれかで、その後に tcp または udp としてプロトコルを指定できます。コマンドは以下の形式になります。

```
source-port port=number_or_range protocol=protocol
```

ロギング

log

カーネルロギングを使用して、ルールへの新しい接続試行を syslog などに記録します。ログメッセージに接頭辞として追加される接頭辞テキストを定義できます。ログレベルは、emerg、alert、crit、error、warning、notice、info、または debug のいずれかです。ログの使用はオプションです。

```
log [prefix=prefix text] [level=log level] limit value=rate/duration
```

でロギングを制限することができます。レートは自然な正の数 [1, ..] で、期間は s、m、h、d です。s は秒、m は分、h は時間、d 日を意味します。最大制限値は 1/d です。これは、1 日あたり最大 1 つのログエントリーを意味します。

audit

Audit は、サービス auditd に送信される監査レコードを使用してロギングを行う別の方法を提供します。監査タイプは ACCEPT、REJECT、または DROP のいずれかにすることができますが、ルールアクションから自動的に 監査 タイプを収集するため、コマンド audit の後に指定されません。監査には独自のパラメーターはありませんが、オプションで制限を追加できます。監査の使用はオプションになります。

アクション

accept|reject|drop|mark

アクションは、accept、reject、drop、または mark のいずれかになります。ルールは、要素またはソースのみを含むことができます。ルールに要素が含まれる場合、その要素に一致する新しい接続は、アクションで処理されます。ルールにソースが含まれている場合、送信元アドレスからのすべてが、指定されたアクションで処理されます。

```
accept | reject [type=reject type] | drop | mark set="mark[/mask]"
```

accept を指定すると、すべての新しい接続試行が許可されます。reject を指定すると、それらは拒否され、そのソースには拒否メッセージが表示されます。拒否タイプは、他の値を使用するように設定することができます。drop を指定すると、すべてのパケットが直ちにドロップされ、ソースには何も情報が送られません。mark を指定すると、すべてのパケットは、指定された mark とオプションの mask でマークされます。

5.15.4. リッチルールログコマンドの使用

ロギングは、Netfilter ログターゲットと audit ターゲットを使用して実行できます。新しいチェーンは、「zone_log」 (zone はゾーン名) という形式の名前を持つすべてのゾーンに追加されます。これは、deny チェーンが適切に順序付けされる前に処理されます。これらのルールまたはその一部は、以下のように、ルールのアクションに従って個別のチェーンに配置されます。

```
zone_log
  zone_deny
  zone_allow
```

すべてのログ記録ルールは 「zone_log」 チェーンに置かれ、最初に解析されます。すべての reject ルールおよび drop ルールは 「ゾーン_deny」 チェーンに置かれ、ログチェーンの後に解析されます。すべての accept ルールは 「ゾーン_allow」 チェーンに置かれ、deny チェーンの後に解析されます。ルールに log が含まれ、deny または allow アクションも含まれる場合、これらのアクションを指定するルールの一部は一致するチェーンに配置されます。

5.15.4.1. リッチルールログコマンドの使用例 1

認証ヘッダープロトコル AH に対して、新しい IPv4 接続および IPv6 接続を有効にします。

```
rule protocol value="ah" accept
```

5.15.4.2. リッチルールログコマンドの使用例 2

プロトコル FTP の新しい IPv4 および IPv6 接続を許可し、監査を使用して 1 分あたり 1 回ログに記録します。

```
rule service name="ftp" log limit value="1/m" audit accept
```

5.15.4.3. リッチルールログコマンドの使用例 3

プロトコル TFTP のアドレス 192.168.0.0/24 からの新しい IPv4 接続を許可し、syslog を使用して 1 分あたり 1 回ログに記録します。

```
rule family="ipv4" source address="192.168.0.0/24" service name="tftp" log prefix="tftp"
level="info" limit value="1/m" accept
```

5.15.4.4. リッチルールログコマンドの使用例 4

プロトコル RADIUS の 1:2:3:4:6:: からの新しい IPv6 接続はすべて拒否され、1 分あたり 3 の速度でログに記録されます。他のソースからの新しい IPv6 接続が許可されます。

```
rule family="ipv6" source address="1:2:3:4:6::" service name="radius" log prefix="dns"
level="info" limit value="3/m" reject
rule family="ipv6" service name="radius" accept
```

5.15.4.5. リッチルールログコマンドの使用例 5

プロトコル TCP を使用するポート 4011 の 1: 2:3:4:6:: から受信した IPv6 パケットを、ポート 4012 の 1::2:3:4:7 に転送します。

```
rule family="ipv6" source address="1:2:3:4:6::" forward-port to-addr="1::2:3:4:7" to-
port="4012" protocol="tcp" port="4011"
```

5.15.4.6. リッチルールログコマンドの使用例 6

ソースアドレスをホワイトリストに登録し、このソースからのすべての接続を許可します。

```
rule family="ipv4" source address="192.168.2.2" accept
```

その他の例は、firewalld.richlanguage (5) の man ページを参照してください。

5.16. ファイアウォールロックダウンの設定

ローカルアプリケーションやサービスは、root として実行している場合は、ファイアウォール設定を変更できます（例： libvirt）。管理者は、この機能を使用してファイアウォール設定をロックし、すべてのアプリケーションでファイアウォール変更を要求できなくするか、ロックダウンのホワイトリストに追加されたアプリケーションのみがファイアウォール変更を要求できるようにすることが可能になります。ロックダウン設定はデフォルトで無効になっています。これを有効にすると、ローカルのアプリケーションやサービスによるファイアウォールへの望ましくない設定変更を確実に防ぐことができます。

5.16.1. コマンドラインクライアントを使用したロックダウンの設定

ロックダウンが有効になっているかどうかを確認するには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --query-lockdown
```

ロックダウンが有効な場合は、コマンドにより yes と出力され、終了ステータスは 0 になります。そうでなければ no と出力され、終了ステータスは 1 になります。

ロックダウンを有効にするには、root で以下のコマンドを入力します。

```
~]# firewall-cmd --lockdown-on
```

ロックダウンを無効にするには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --lockdown-off
```

5.16.2. コマンドラインクライアントを使用したロックダウンのホワイトリストオプションの設定

ロックダウンのホワイトリストには、コマンド、セキュリティーのコンテキスト、ユーザー、およ

びユーザー ID を追加できます。ホワイトリストのコマンドエントリーがアスタリスク「*」で終了している場合は、そのコマンドで始まるすべてのコマンドラインが一致することになります。「*」がなければ、コマンドと引数が完全に一致する必要があります。

ここでのコンテキストは、実行中のアプリケーションやサービスのセキュリティー (SELinux) コンテキストです。実行中のアプリケーションのコンテキストを確認するには、次のコマンドを実行します。

```
~]$ ps -e --context
```

このコマンドは、実行中のアプリケーションをすべて返します。grep ツールを介して出力をパイプ処理して、対象のアプリケーションを取得します。以下に例を示します。

```
~]$ ps -e --context | grep example_program
```

ホワイトリストにあるコマンドラインの一覧を表示するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --list-lockdown-whitelist-commands
```

ホワイトリストに command コマンドを追加するには、root で以下のコマンドを入力します。

```
~]# firewall-cmd --add-lockdown-whitelist-command='/usr/bin/python -Es /usr/bin/command'
```

ホワイトリストから command コマンドを削除するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --remove-lockdown-whitelist-command='/usr/bin/python -Es /usr/bin/command'
```

command コマンドがホワイトリストにあるかどうかを確認するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --query-lockdown-whitelist-command='/usr/bin/python -Es /usr/bin/command'
```

このコマンドは、true の場合は yes と出力され、終了ステータスは 0 になります。そうでなければ no と出力され、終了ステータスは 1 になります。

ホワイトリストにあるセキュリティーコンテキストの一覧を表示するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --list-lockdown-whitelist-contexts
```

ホワイトリストに context コンテキスト を追加するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --add-lockdown-whitelist-context=context
```

ホワイトリストから context コンテキスト を削除するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --remove-lockdown-whitelist-context=context
```

context コンテキスト がホワイトリストにあるかどうかを確認するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --query-lockdown-whitelist-context=context
```

yes を終了ステータス 0 で出力し、true の場合は no と出力し、それ以外は終了ステータス 1 を出力します。

ホワイトリストにあるユーザー ID の一覧を表示するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --list-lockdown-whitelist-uids
```

ホワイトリストにユーザー ID (uid)を追加するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --add-lockdown-whitelist-uid=uid
```

ホワイトリストからユーザー ID (uid)を削除するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --remove-lockdown-whitelist-uid=uid
```

ホワイトリストにユーザー ID (uid) があるかどうかを確認するには、次のコマンドを実行します。

```
~]$ firewall-cmd --query-lockdown-whitelist-uid=uid
```

yes を終了ステータス 0 で出力し、true の場合は no と出力し、それ以外は終了ステータス 1 を出力します。

ホワイトリストにあるユーザー名の一覧を表示するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --list-lockdown-whitelist-users
```

ユーザー名(user)をホワイトリストに追加するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --add-lockdown-whitelist-user=user
```

ホワイトリストからユーザー名(user)を削除するには、root で次のコマンドを実行します。

```
~]# firewall-cmd --remove-lockdown-whitelist-user=user
```

ホワイトリストにユーザー名 (user) があるかどうかを確認するには、次のコマンドを実行します。

```
~]$ firewall-cmd --query-lockdown-whitelist-user=user
```

yes を終了ステータス 0 で出力し、true の場合は no と出力し、それ以外は終了ステータス 1 を出力します。

5.16.3. 設定ファイルを使用したロックダウンのホワイトリストオプションの設定

デフォルトのホワイトリスト設定ファイルには、NetworkManager コンテキストと、libvirt のデフォルトのコンテキストが含まれます。リストには、ユーザー ID (0) もあります。

```
<?xml version="1.0" encoding="utf-8"?>
 <whitelist>
   <selinux context="system_u:system_r:NetworkManager_t:s0"/>
   <selinux context="system_u:system_r:virtd_t:s0-s0:c0.c1023"/>
   <user id="0"/>
 </whitelist>
```

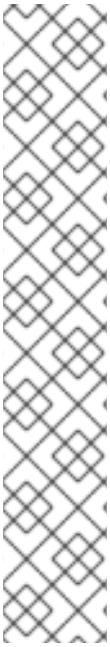
以下は、firewall-cmd ユーティリティーで、ユーザー ID が 815 である user ユーザーのすべてのコマンドを有効にするホワイトリスト設定ファイルの例です。

```
<?xml version="1.0" encoding="utf-8"?>
 <whitelist>
   <command name="/usr/bin/python -Es /bin/firewall-cmd*"/>
   <selinux context="system_u:system_r:NetworkManager_t:s0"/>
   <user id="815"/>
   <user name="user"/>
 </whitelist>
```

この例では、user id と user name の両方が使用されていますが、実際にはどちらか一方のオプションだけが必要です。Python はインタープリターとしてコマンドラインに追加されています。特定のコマンドを使用することもできます（例：

```
/usr/bin/python /bin/firewall-cmd --lockdown-on
```

-lockdown-on コマンドのみが許可されます）。



注記

Red Hat Enterprise Linux 7 では、すべてのユーティリティーが /usr/bin/ ディレクトリーに格納され、/bin/ ディレクトリーは /usr/bin/ ディレクトリーへのシンボリックリンクになります。つまり、root で実行する際の firewall-cmd のパスは /bin/firewall-cmd に解決しますが、/usr/bin/firewall-cmd を使用できるようになりました。新たなスクリプトは、すべて新しい格納場所を使用する必要があります。ただし、root で実行するスクリプトが /bin/firewall-cmd パスを使用するように書き込まれている場合は、これまではroot 以外のユーザーにのみ使用されていた /usr/bin/firewall-cmd パスに加えて、このコマンドのパスもホワイトリスト化する必要があります。

コマンドの名前属性の最後にある 「」 は、この文字列で始まるすべてのコマンドが一致することを意味します。「*」 がない場合は、引数を含む絶対コマンドが一致する必要があります。*

5.17. 拒否されたパケットに対するロギングの設定

firewalld に LogDenied オプションを使用すると、拒否されたパケットに単純なロギングメカニズムを追加できます。対象となるのは、拒否または破棄されるパケットになります。ログの設定を変更するには、/etc/firewalld/firewalld.conf ファイルを編集するか、コマンドラインまたは GUI 設定ツールを使用します。

LogDenied を有効にすると、デフォルトルールの INPUT チェイン、FORWARD チェイン、および OUTPUT チェインの reject ルールおよび drop ルールと、ゾーンの最後の reject ルールおよび drop ルールの直前に、ロギングルールが追加されます。この設定に使用できる値は、all、unicast、broadcast、マルチキャスト、および off です。デフォルト設定は off です。ユニキャスト、ブロードキャスト、および マルチキャスト の設定では、pkttype 一致を使用してリンク層パケットタイプを照合します。すべての で、すべて のパケットがログに記録されます。

firewall-cmd を使用して実際の LogDenied 設定を一覧表示するには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --get-log-denied
off
```

LogDenied 設定を変更するには、root で以下のコマンドを実行します。

```
~]# firewall-cmd --set-log-denied=all
success
```

-

firewalld GUI 設定ツールで LogDenied 設定を変更するには、firewall-config を起動し、Options メニュー をクリックし、Change Log Denied を選択します。LogDenied ウィンドウが表示されます。メニューから新しい LogDenied 設定を選択し、OK をクリックします。

5.18. 関連情報

以下の資料は、firewalld に関するその他の情報を提供します。

5.18.1. インストールされているドキュメント

- man ページの firewalld (1) - firewalld のコマンドオプションが説明されています。
- firewalld.conf (5) man ページ - firewalld を設定する情報が含まれます。
- man ページの firewall-cmd (1) - firewalld コマンドラインクライアントのコマンドオプションが説明されています。
- man ページの firewall-config (1) - firewall-config ツールの設定について説明しています。
- man ページの firewall-offline-cmd (1) - firewalld オフラインコマンドラインクライアントのコマンドオプションが説明されています。
- firewalld.icmptype (5) man ページ - ICMP フィルターリングの XML 設定ファイルが説明されています。
- firewalld.ipset (5) man ページ - firewalld IP セットの XML 設定ファイルが説明されています。
- man ページの firewalld.service (5) - firewalld サービス 用の XML 設定ファイルが説明されています。
- firewalld.zone (5) man ページ - firewalld ゾーン設定の XML 設定ファイルが説明されています。

- *firewalld.direct (5) man ページ - firewalld ダイレクトインターフェイスの設定ファイルが説明されています。*

- *man ページの firewalld.lockdown-whitelist (5) - firewalld ロックダウンホワイトリストの設定ファイルが説明されています。*

- *man ページの firewalld.richlanguage (5) - firewalld リッチ言語のルール構文が説明されています。*

- *man ページの firewalld.zones (5) - ゾーンの全般的な説明と設定方法が説明されています。*

- *firewalld.dbus (5) man ページ - firewalld の D-Bus インターフェイスが説明されています。*

5.18.2. オンラインドキュメント

- *http://www.firewalld.org/ — firewalld home page.*

第6章 NFTABLES の使用

nftables フレームワークは、パケットの分類機能を提供し、iptables ツール、ip6tables ツール、arptables ツール、ebtables ツール、および ipset ツールの後継となります。利便性、機能、パフォーマンスにおいて、以前のパケットフィルターリングツールに多くの改良が追加されました。以下に例を示します。

- 線形処理の代わりに組み込みルックアップテーブルを使用
- IPv4 プロトコルと IPv6 プロトコルの両方に対する単一のフレームワーク
- 完全ルールセットのフェッチ、更新、および保存を行わず、すべてアトミックに適用されるルール
- ルールセットでのデバッグとトレースのサポート(nftrace)およびトレースイベントの監視(nft ツール)
- より統一されたコンパクトな構文、プロトコル固有の拡張なし
- サードパーティーのアプリケーション用 Netlink API

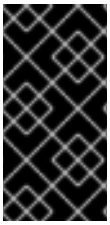
iptables と同様に、nftables はチェーンを保存するためにテーブルを使用します。このチェーンには、アクションを実行する個々のルールが含まれます。nft ツールは、以前のパケットフィルターリングフレームワークのツールをすべて置き換えます。libnftnl ライブラリーは、libmnl ライブラリーを介した Netlink API の nftables との低レベルの対話に使用できます。

ルールセットの変更効果を表示するには、nft list ruleset コマンドを使用します。これらのツールは、テーブル、チェーン、ルール、セット、およびその他のオブジェクトを nftables ルールセットに追加するため、nft flush ruleset コマンドなどの nftables ルールセット操作は、先に別のレガシーコマンドを使用してインストールしたルールセットに影響を及ぼす可能性があることに注意してください。

FIREWALLD または NFTABLES を使用するタイミング

- firewalld: 簡単な ファイアウォール のユースケースには、firewalld ユーティリティーを使用します。このユーティリティーは、使いやすく、このようなシナリオの一般的な使用例に対応しています。

- *nftables: nftables ユーティリティーを使用して、ネットワーク全体など、複雑なパフォーマンスに関する重要なファイアウォールを設定します。*



重要

異なるファイアウォールサービスが相互に影響することを回避するには、RHEL ホストでそのうちの 1 つだけを実行し、他のサービスを無効にします。

6.1. NFTABLES スクリプトの作成および実行

nftables フレームワークは、シェルスクリプトを使用して ファイアウォール ルールを維持するための主な利点を提供するネイティブのスクリプト環境を提供します。スクリプトの実行はアトミックです。つまり、システムがスクリプト全体を適用するか、エラーが発生した場合には実行を阻止することを意味します。これにより、ファイアウォールは常に一貫した状態になります。

さらに、管理者は nftables スクリプト環境を使用すると、以下が可能になります。

- *コメントの追加*
- *変数の定義*
- *他のルールセットファイルの組み込み*

本セクションでは、この機能を使用する方法と、nftables スクリプトの作成および実行方法を説明します。

*nftables パッケージをインストールすると、Red Hat Enterprise Linux は、/etc/nftables/ ディレクトリーに *.nft スクリプトを自動的に作成します。このスクリプトには、さまざまな目的でテーブルと空のチェーンを作成するコマンドが含まれます。*

6.1.1. 対応している nftables スクリプトの形式

nftables スクリプト環境は、以下の形式でスクリプトに対応します。

-

nft list ruleset コマンドがルールセットを表示するのと同じ形式でスクリプトを作成できます。

```
#!/usr/sbin/nft -f

# Flush the rule set
flush ruleset

table inet example_table {
  chain example_chain {
    # Chain for incoming packets that drops all packets that
    # are not explicitly allowed by any rule in this chain
    type filter hook input priority 0; policy drop;

    # Accept connections to port 22 (ssh)
    tcp dport ssh accept
  }
}
```

- nft コマンドと同じ構文を使用できます。

```
#!/usr/sbin/nft -f

# Flush the rule set
flush ruleset

# Create a table
add table inet example_table

# Create a chain for incoming packets that drops all packets
# that are not explicitly allowed by any rule in this chain
add chain inet example_table example_chain { type filter hook input priority 0 ; policy drop ; }

# Add a rule that accepts connections to port 22 (ssh)
add rule inet example_table example_chain tcp dport ssh accept
```

6.1.2. nftables スクリプトの実行

nftables スクリプトは、nft ユーティリティーに渡すか、スクリプトを直接実行できます。

前提条件

- 本セクションの手順では、nftables スクリプトを /etc/nftables/example_firewall.nft ファイルに保存していることを前提としています。

手順6.1 nft ユーティリティーを使用した nftables スクリプトの実行

- *nftables スクリプトを nft ユーティリティーに渡して実行するには、次のコマンドを実行します。*

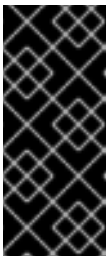
```
# nft -f /etc/nftables/example_firewall.nft
```

手順6.2 nftables スクリプトを直接実行します。

1. *以下の手順は、一度だけ必要です。*

   1. *スクリプトが以下のシバンシーケンスで始まることを確認します。*

      ```
      #!/usr/sbin/nft -f
      ```

      重要

      -f パラメーターを省略すると、nft ユーティリティーはスクリプトを読み取らず、Error: syntax error, unexpected newline, expecting string を表示します。

   2. *必要に応じて、スクリプトの所有者を root に設定します。*

      ```
      # chown root /etc/nftables/example_firewall.nft
      ```

   3. *所有者のスクリプトを実行ファイルに変更します。*

      ```
      # chmod u+x /etc/nftables/example_firewall.nft
      ```

2. *スクリプトを実行します。*

   ```
   # /etc/nftables/example_firewall.nft
   ```

   出力が表示されない場合は、システムがスクリプトを正常に実行します。

重要

nft がスクリプトを正常に実行したり、ルールを誤って配置したり、パラメーターが欠落したり、スクリプト内のその他の問題したりすると、ファイアウォールが期待どおりに動作しない可能性があります。

関連情報

- ファイルの所有者の設定に関する詳細は、chown (1) の man ページを参照してください。
- ファイルのパーミッション設定の詳細は、chmod (1) の man ページを参照してください。
- システム起動時に nftables ルールを読み込む方法は、を参照してください。 「システムの起動時に nftables ルールの自動読み込み」

6.1.3. nftables スクリプトでコメントの使用

nftables スクリプト環境は、# 文字の右側にあるすべてをコメントとして解釈します。

例6.1 nftables スクリプトのコメント

コメントは、コマンドの横だけでなく、行の先頭からも開始できます。

```
...
# Flush the rule set
flush ruleset

add table inet example_table  # Create a table
...
```

6.1.4. nftables スクリプトで変数の使用

nftables スクリプトで変数を定義するには、define キーワードを使用します。シングル値および匿名セットを変数に保存できます。より複雑なシナリオの場合は、名前付きセットまたは決定マップを使用します。

値を 1 つ持つ変数

以下の例は、値が **enp1s0** の **INET_DEV** という名前の変数を定義します。

```
define INET_DEV = enp1s0
```

スクリプトで変数を使用するには、**$** 記号とそれに続く変数名を指定します。

```
...
add rule inet example_table example_chain iifname $INET_DEV tcp dport ssh accept
...
```

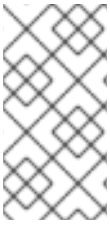
匿名セットを含む変数

以下の例では、匿名セットを含む変数を定義します。

```
define DNS_SERVERS = { 192.0.2.1, 192.0.2.2 }
```

スクリプトで変数を使用するには、**$** 記号とそれに続く変数名を指定します。

```
add rule inet example_table example_chain ip daddr $DNS_SERVERS accept
```



注記

中括弧は、変数がセットを表していることを示すため、ルールで使用する場合は、特別なセマンティクスを持つことに注意してください。

関連情報

- セットの詳細は、「**nftables** コマンドを使用したセットの使用」 を参照してください。
- 決定マップの詳細は、「**nftables** コマンドにおける決定マップの使用」 を参照してください。

6.1.5. nftables スクリプトへのファイルの追加

nftables スクリプト環境を使用すると、管理者は **include** ステートメントを使用して他のスクリプトを 含める ことができます。

絶対パスまたは相対パスなしでファイル名のみを指定すると、nftables には、Red Hat Enterprise Linux では /etc に設定されているデフォルトの検索パスからのファイルが含まれます。

> **例6.2 デフォルト検索ディレクトリーからのファイルを含む**
>
> デフォルトの検索ディレクトリーからファイルを指定するには、次のコマンドを実行します。
>
> ```
> include "example.nft"
> ```

> **例6.3 ディレクトリーからの *.nft ファイルをすべて含む**
>
> /etc/nftables/rulesets/ ディレクトリーに保存されている *.nft で終わるすべてのファイルを含めるには、次のコマンドを実行します。
>
> ```
> include "/etc/nftables/rulesets/*.nft"
> ```
>
> include ステートメントは、ドットで始まるファイルと一致しないことに注意してください。

関連情報

- 詳細は、nft (8) man ページの Include files セクションを参照してください。

6.1.6. システムの起動時に nftables ルールの自動読み込み

nftables systemd サービスは、/etc/sysconfig/nftables.conf ファイルに含まれるファイアウォールスクリプトを読み込みます。本セクションでは、システムの起動時にファイアウォールルールを読み込む方法を説明します。

前提条件

- nftables スクリプトは、/etc/nftables/ ディレクトリーに保存されます。

手順6.3 システムの起動時に nftables ルールの自動読み込み

1. /etc/sysconfig/nftables.conf ファイルを編集します。

nftables パッケージをインストールしたときに /etc/nftables/ で作成された *.nft スクリプトを強化する場合は、これらのスクリプトの include ステートメントのコメントを解除します。

- スクリプトを新規に作成する場合は、そのスクリプトを含む include ステートメントを追加します。たとえば、nftables サービスの起動時に /etc/nftables/example.nft スクリプトを読み込むには、以下を追加します。

```
include "/etc/nftables/example.nft"
```

2. 必要に応じて、nftables サービスを起動し、システムを再起動せずにファイアウォールルールを読み込みます。

```
# systemctl start nftables
```

3. nftables サービスを有効にします。

```
# systemctl enable nftables
```

関連情報

- 詳細は、「対応している nftables スクリプトの形式」 を参照してください。

6.2. NFTABLES テーブル、チェーン、およびルールの作成および管理

本セクションでは、nftables ルールセットを表示する方法と、その管理方法を説明します。

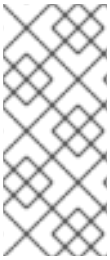
6.2.1. nftables ルールセットの表示

nftables のルールセットには、テーブル、チェーン、およびルールが含まれます。本セクションでは、このルールセットを表示する方法を説明します。

ルールセットを表示するには、以下のコマンドを実行します。

```
# nft list ruleset
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
```

```
    tcp dport http accept
    tcp dport ssh accept
  }
}
```



注記

デフォルトでは、nftables は事前にテーブルを作成しません。これにより、テーブルのないホストに設定されたルールセットを表示すると、nft list ruleset コマンドは出力を表示しません。

6.2.2. nftables テーブルの作成

nftables のテーブルは、チェーン、ルール、セット、およびその他のオブジェクトのコレクションを含む名前空間です。本セクションでは、テーブルの作成方法を説明します。

各テーブルには、アドレスファミリーが定義されている必要があります。テーブルのアドレスファミリーは、テーブルプロセスのアドレスタイプを定義します。テーブルを作成する際に、以下のいずれかのアドレスファミリーを設定できます。

- ip: IPv4 パケットのみに一致します。アドレスファミリーを指定しないと、これがデフォルトになります。
- ip6: IPv6 パケットのみに一致します。
- inet: IPv4 パケットと IPv6 パケットの両方に一致します。
- ARP: IPv4 アドレス解決プロトコル(ARP)パケットに一致します。
- bridge: ブリッジデバイスを通過するパケットと一致します。
- netdev: ingress からのパケットに一致します。

手順6.4 nftables テーブルの作成

1. nft add table コマンドを使用して、新しいテーブルを作成します。たとえば、IPv4 パケッ

トおよび IPv6 パケットを処理する example_table という名前のテーブルを作成するには、次のコマンドを実行します。

```
# nft add table inet example_table
```

2. 必要に応じて、ルールセットのテーブルを一覧表示します。

```
# nft list tables
table inet example_table
```

関連情報

- アドレスファミリーの詳細は、nft (8) man ページの Address family セクションを参照してください。
- テーブルで実行できるその他のアクションの詳細は、nft (8) man ページの Tables セクションを参照してください。

6.2.3. nftables チェーンの作成

チェーンは、ルールのコンテナーです。次の 2 つのルールタイプが存在します。

- ベースチェーン - ネットワークスタックからのパケットのエントリーポイントとしてベースチェーンを使用できます。
- 通常のチェーン： jump ターゲットとして通常のチェーンを使用し、ルールをより適切に整理できます。

この手順では、既存のテーブルにベースチェーンを追加する方法を説明します。

前提条件

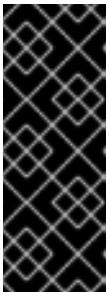
- 新しいチェーンを追加するテーブルが存在する。

手順6.5 nftables チェーンの作成

1. nft add chain コマンドを使用して、新しいチェーンを作成します。たとえ

ば、example_table に、example_chain という名前のチェーンを作成するには、次のコマンドを実行します。

```
# nft add chain inet example_table example_chain '{ type filter hook input priority 0 ; policy accept ; }'
```



重要

シェルで、セミコロンがコマンドの最後として解釈されないようにするには、セミコロンをバックスラッシュでエスケープする必要があります。さらに、一部のシェルは中括弧も解釈するので、中括弧とその中のものはティック(')で引用します。

このチェーンは、着信パケットをフィルターリングします。priority パラメーターは、nftables が同じフック値を持つチェーンを処理する順序を指定します。優先度の値が低いほど優先されます。policy パラメーターは、このチェーンのルールのデフォルトアクションを設定します。サーバーにリモートでログインし、デフォルトのポリシーを を ドロップ するように設定すると、他のルールでリモートアクセスが許可されていない場合は、すぐに切断されることに注意してください。

2. *必要に応じて、すべてのチェーンを表示します。*

```
# nft list chains
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
  }
}
```

関連情報

- *アドレスファミリーの詳細は、nft (8) man ページの Address family セクションを参照してください。*
- *チェーンで実行できるその他のアクションの詳細は、nft (8) man ページの Chains セクションを参照してください。*

6.2.4. nftables チェーンの最後に対するルールの追加

本セクションでは、既存の nftables チェーンの最後にルールを追加する方法を説明します。

前提条件

- *ルールを追加するチェーンが存在する。*

手順6.6 nftables チェーンの最後に対するルールの追加

1. *新しいルールを追加するには、**nft add rule** コマンドを使用します。たとえば、**example_table** の **example_chain** に、ポート **22** の **TCP** トラフィックを許可するルールを追加するには、次のコマンドを実行します。*

```
# nft add rule inet example_table example_chain tcp dport 22 accept
```

*代わりに、ポート番号の代わりにサービスの名前を指定することもできます。この例では、ポート番号 **22** の代わりに **ssh** を使用できます。サービス名は、**/etc/services** ファイルのエントリーに基づいてポート番号に解決されることに注意してください。*

2. *必要に応じて、**example_table** ですべてのチェーンとそのルールを表示します。*

```
# nft list table inet example_table
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    ...
    tcp dport ssh accept
  }
}
```

関連情報

- *アドレスファミリーの詳細は、**nft (8) man** ページの **Address family** セクションを参照してください。*
- *チェーンで実行できるその他のアクションの詳細は、**nft (8) man** ページの **Rules** セクションを参照してください。*

6.2.5. nftables チェーンの先頭へのルールの挿入

*本セクションでは、既存の **nftables** チェーンの先頭にルールを追加する方法を説明します。*

前提条件

-

- ルールを追加するチェーンが存在する。

手順6.7 nftables チェーンの先頭へのルールの挿入

1. 新しいルールを挿入するには、**nft insert rule** コマンドを使用します。たとえば、ポート **22** で **TCP** トラフィックを許可するルールを **example_table** の **example_chain** に挿入するには、次のコマンドを実行します。

```
# nft insert rule inet example_table example_chain tcp dport 22 accept
```

代わりに、ポート番号の代わりにサービスの名前を指定することもできます。この例では、ポート番号 **22** の代わりに **ssh** を使用できます。サービス名は、**/etc/services** ファイルのエントリーに基づいてポート番号に解決されることに注意してください。

2. 必要に応じて、**example_table** ですべてのチェーンとそのルールを表示します。

```
# nft list table inet example_table
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    tcp dport ssh accept
    ...
  }
}
```

関連情報

- アドレスファミリーの詳細は、**nft (8) man** ページの **Address family** セクションを参照してください。
- チェーンで実行できるその他のアクションの詳細は、**nft (8) man** ページの **Rules** セクションを参照してください。

6.2.6. nftables チェーンの特定の位置へのルールの挿入

本セクションでは、**nftables** チェーンで、既存のルールの前後にルールを追加する方法を説明します。これにより、正しい場所に新しいルールを配置することができます。

前提条件

- ルールを追加するチェーンが存在する。

手順6.8 nftables チェーンの特定の位置へのルールの挿入

1. **nft -a list ruleset** コマンドを使用して、ハンドルを含む **example_table** のすべてのチェーンとそのルールを表示します。

```
# nft -a list table inet example_table
table inet example_table { # handle 1
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport 22 accept # handle 2
    tcp dport 443 accept # handle 3
    tcp dport 389 accept # handle 4
  }
}
```

-a を使用すると、ハンドルが表示されます。次の手順で新しいルールを配置するときに、この情報が必要です。

2. **example_table** の **example_chain** チェーンに新しいルールを挿入します。

- ハンドル **3** の前に、ポート **636** で **TCP** トラフィックを許可するルールを挿入するには、次のコマンドを実行します。

```
# nft insert rule inet example_table example_chain position 3 tcp dport 636 accept
```

- ハンドル **3** の後ろに、ポート **80** で **TCP** トラフィックを許可するルールを追加するには、次のコマンドを実行します。

```
# nft add rule inet example_table example_chain position 3 tcp dport 80 accept
```

3. 必要に応じて、**example_table** ですべてのチェーンとそのルールを表示します。

```
# nft -a list table inet example_table
table inet example_table { # handle 1
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport 22 accept # handle 2
    tcp dport 636 accept # handle 5
```

```
    tcp dport 443 accept # handle 3
    tcp dport 80 accept # handle 6
    tcp dport 389 accept # handle 4
  }
}
```

関連情報

- アドレスファミリーの詳細は、nft (8) man ページの Address family セクションを参照してください。
- チェーンで実行できるその他のアクションの詳細は、nft (8) man ページの Rules セクションを参照してください。

6.3. NFTABLES を使用した NAT の設定

nftables を使用すると、以下のネットワークアドレス変換(NAT)タイプを設定できます。

- マスカレーディング
- ソース NAT (SNAT)
- 宛先 NAT (DNAT)
- リダイレクト

6.3.1. 異なる NAT タイプ: マスカレード、ソース NAT、宛先 NAT、リダイレクト

以下は、ネットワークアドレス変換(NAT)タイプになります。

マスカレードおよびソースの NAT (SNAT)

これらの NAT タイプのいずれかを使用して、パケットのソース IP アドレスを変更します。たとえば、インターネットサービスプロバイダーは、10.0.0.0/8 などのプライベート IP 範囲はルーティングしません。ネットワークでプライベート IP 範囲を使用し、ユーザーがインターネット上のサーバーにアクセスできるようにする必要がある場合は、この範囲のパケットのソース IP アドレスをパブリック IP アドレスにマップします。

マスカレードと SNAT の両方は非常に似ています。相違点は次のとおりです。

- マスカレードは、出力インターフェイスの IP アドレスを自動的に使用します。したがって、出力インターフェイスが動的 IP アドレスを使用する場合は、マスカレードを使用します。

- SNAT は、パケットのソース IP アドレスを指定された IP に設定し、送信インターフェイスの IP を動的に検索しません。したがって、SNAT はマスカレードよりも高速です。送信インターフェイスが固定 IP アドレスを使用する場合は SNAT を使用します。

宛先 NAT (DNAT)

この NAT タイプを使用して、着信トラフィックを別のホストにルーティングします。たとえば、Web サーバーが予約済み IP 範囲の IP アドレスを使用しているため、インターネットから直接アクセスできない場合は、ルーターに DNAT ルールを設定し、着信トラフィックをこのサーバーにリダイレクトできます。

リダイレクト

このタイプは、チェーンフックに応じてパケットをローカルマシンにリダイレクトする DNAT の特殊なケースです。たとえば、サービスが標準ポートとは異なるポートで実行する場合は、標準ポートからこの特定のポートに着信トラフィックをリダイレクトすることができます。

6.3.2. nftables を使用したマスカレードの設定

マスカレードを使用すると、ルーターは、インターフェイスを介して送信されるパケットのソース IP を、インターフェイスの IP アドレスに動的に変更できます。つまり、インターフェイスに新しい IP が割り当てられると、nftables はソース IP を置き換えるときに新しい IP を自動的に使用します。

以下の手順では、ens3 インターフェイスを介してホストから ens3 に設定された IP にホストから出るパケットのソース IP を置き換える方法を説明します。

手順6.9 nftables を使用したマスカレードの設定

1. テーブルを作成します。

```
# nft add table nat
```

2. テーブルに prerouting および postrouting チェーンを追加します。

```
# nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
# nft add chain nat postrouting { type nat hook postrouting priority 100 \; }
```

重要

prerouting チェーンにルールを追加しなくても、nftables フレームワークでは、受信パケット返信に一致するようにこのチェーンが必要になります。

シェルが優先度の負の値を nft コマンドのオプションとして解釈しないように、-- オプションを nft コマンドに渡す必要があることに注意してください。

3. **ens3 インターフェイスの送信パケットに一致するルールを postrouting チェーンに追加します。**

```
# nft add rule nat postrouting oifname "ens3" masquerade
```

6.3.3. nftables を使用したソース NAT の設定

ルーターでは、ソース NAT (SNAT)を使用して、インターフェイスを介して特定の IP アドレスに送信するパケットの IP を変更できます。

以下の手順では、ens3 インターフェイスを介してルーターから 192.0.2.1 に、パケットのソース IP を置き換える方法を説明します。

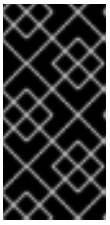
手順6.10 nftables を使用したソース NAT の設定

1. **テーブルを作成します。**

```
# nft add table nat
```

2. **テーブルに prerouting および postrouting チェーンを追加します。**

```
# nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
# nft add chain nat postrouting { type nat hook postrouting priority 100 \; }
```



重要

prerouting チェーンにルールを追加しなくても、nftables フレームワークでは、このチェーンが発信パケット返信に一致するようにする必要があります。

シェルが優先度の負の値を nft コマンドのオプションとして解釈しないように、-- オプションを nft コマンドに渡す必要があることに注意してください。

3. ens3 を介した送信パケットのソース IP を 192.0.2.1 に置き換えるルールを postrouting チェーンに追加します。

```
# nft add rule nat postrouting oifname "ens3" snat to 192.0.2.1
```

関連情報

- 詳細は、「特定のローカルポートで着信パケットを別のホストに転送」 を参照してください。

6.3.4. nftables を使用した宛先 NAT の設定

宛先 NAT により、ルーター上のトラフィックをインターネットから直接アクセスできないホストにリダイレクトできます。

以下の手順では、ルーターのポート 80 および 443 に送信された受信トラフィックを、IP アドレス 192.0.2.1 を持つホストにリダイレクトする方法を説明します。

手順6.11 nftables を使用した宛先 NAT の設定

1. テーブルを作成します。

```
# nft add table nat
```

2. テーブルに prerouting および postrouting チェーンを追加します。

```
# nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
# nft add chain nat postrouting { type nat hook postrouting priority 100 \; }
```

重要

postrouting チェーンにルールを追加しなくても、nftables フレームワークでは、このチェーンが発信パケット返信に一致するようにする必要があります。

シェルが優先度の負の値を nft コマンドのオプションとして解釈しないように、-- オプションを nft コマンドに渡す必要があることに注意してください。

3. IP が 192.0.2.1 のホストにポート 80 および 443 に送信された ens3 インターフェイスの着信トラフィックをリダイレクトするルールを prerouting チェーンに追加します。

```
# nft add rule nat prerouting iifname ens3 tcp dport { 80, 443 } dnat to 192.0.2.1
```

4. 環境に応じて、SNAT ルールまたはマスカレードルールを追加して、ソースアドレスを変更します。

   1. ens3 インターフェイスが動的 IP アドレスを使用している場合は、マスカレードルールを追加します。

   ```
   # nft add rule nat postrouting oifname "ens3" masquerade
   ```

   2. ens3 インターフェイスが静的 IP アドレスを使用する場合は、SNAT ルールを追加します。たとえば、ens3 が 198.51.100.1 IP アドレスを使用する場合：

   ```
   # nft add rule nat postrouting oifname "ens3" snat to 198.51.100.1
   ```

関連情報

- 詳細は、「異なる NAT タイプ: マスカレード、ソース NAT、宛先 NAT、リダイレクト」を参照してください。

6.3.5. nftables を使用したリダイレクトの設定

リダイレクト 機能は、チェーンフックに応じてパケットをローカルマシンにリダイレクトする宛先ネットワークアドレス変換(DNAT)の特別なケースです。

以下の手順では、ローカルホストの 22 ポートに送信される着信トラフィックおよび転送されたトラ

フィックを 2222 ポートにリダイレクトする方法を説明します。

手順6.12 nftables を使用したリダイレクトの設定

1. テーブルを作成します。

```
# nft add table nat
```

2. テーブルに **prerouting** チェーンを追加します。

```
# nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
```

シェルが優先度の負の値を **nft** コマンドのオプションとして解釈しないように、**--** オプションを **nft** コマンドに渡す必要があることに注意してください。

3. 22 ポートの着信トラフィックを 2222 ポートにリダイレクトするルールを **prerouting** チェーンに追加します。

```
# nft add rule nat prerouting tcp dport 22 redirect to 2222
```

関連情報

- 詳細は、「異なる NAT タイプ: マスカレード、ソース NAT、宛先 NAT、リダイレクト」を参照してください。

6.4. NFTABLES コマンドを使用したセットの使用

nftables フレームワークは、セットをネイティブにサポートします。たとえば、ルールが複数の IP アドレス、ポート番号、インターフェイス、またはその他の一致基準に一致する必要がある場合など、セットを使用できます。

6.4.1. nftables での匿名セットの使用

匿名セットには、ルールで直接使用する { 22, 80, 443 } などの中括弧で囲まれたコンマ区切りの値が含まれます。IP アドレスやその他の一致基準に匿名セットを使用することもできます。

匿名セットの欠点は、セットを変更する場合はルールを置き換える必要があることです。動的なソ

リューションの場合は、「nftables で名前付きセットの使用」に従って名前付きセットを使用します。

前提条件

- inet ファミリーに example_chain チェーンと example_table テーブルが存在する。

手順6.13 nftables での匿名セットの使用

1. たとえば、ポート 22、80、および 443 への着信トラフィックを許可するルールを example_table の example_chain に追加するには、次のコマンドを実行します。

```
# nft add rule inet example_table example_chain tcp dport { 22, 80, 443 } accept
```

2. 必要に応じて、example_table ですべてのチェーンとそのルールを表示します。

```
# nft list table inet example_table
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    tcp dport { ssh, http, https } accept
  }
}
```

6.4.2. nftables で名前付きセットの使用

nftables フレームワークは、変更可能な名前付きセットに対応します。名前付きセットは、テーブル内の複数のルールで使用できる要素の一覧または範囲です。匿名セットに対する別の利点として、セットを使用するルールを置き換えることなく、名前付きセットを更新できます。

名前付きセットを作成する場合は、セットに含まれる要素のタイプを指定する必要があります。以下のタイプを設定できます。

- 192. 0.2.1 や 192. 0.2.0/24 など、IPv4 アドレスまたは範囲を含むセットの場合は ipv4_addr。
- 2001:db8: 1::1 や 2001:db8: 1::1 / 64 など、IPv6 アドレスまたは範囲を含むセットの場合は ipv6_addr。
- 52:54:00:6b:66: 42 など、メディアアクセス制御(MAC)アドレスの一覧を含むセットの場

合は ether_addr。

- tcp など、インターネットプロトコルタイプの一覧を含むセットの場合は inet_proto。
- ssh などのインターネットサービスの一覧を含むセットの場合は inet_service。
- パケット マーク の一覧を含むセットの場合は mark。パケットマークは、任意の正の 32 ビットの整数値にすることができます(0 から 2147483647)。

前提条件

- example_chain チェーンと example_table テーブルが存在する。

手順6.14 nftables で名前付きセットの使用

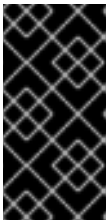
1. 空のファイルを作成します。以下の例では、IPv4 アドレスのセットを作成します。

   a. 複数の IPv4 アドレスを保存できるセットを作成するには、次のコマンドを実行します。

   ```
   # nft add set inet example_table example_set { type ipv4_addr \; }
   ```

   b. IPv4 アドレス範囲を保存できるセットを作成するには、次のコマンドを実行します。

   ```
   # nft add set inet example_table example_set { type ipv4_addr \; flags interval \; }
   ```

   重要

   シェルで、セミコロンがコマンドの最後として解釈されないようにするには、セミコロンをバックスラッシュでエスケープする必要があります。

2. 必要に応じて、セットを使用するルールを作成します。たとえば、次のコマンドは、example_set の IPv4 アドレスからのパケットをすべて破棄するルールを example_table の example_chain に追加します。

   ```
   # nft add rule inet example_table example_chain ip saddr @example_set drop
   ```

example_set が空のままなので、ルールには現在影響がありません。

3. IPv4 アドレスを example_set に追加します。

a. 個々の IPv4 アドレスを保存するセットを作成する場合は、次のコマンドを実行します。

```
# nft add element inet example_table example_set { 192.0.2.1, 192.0.2.2 }
```

b. IPv4 範囲を保存するセットを作成する場合は、次のコマンドを実行します。

```
# nft add element inet example_table example_set { 192.0.2.0-192.0.2.255 }
```

IP アドレス範囲を指定する場合は、上記の例の 192.0.2.0/24 など、CIDR(Classless Inter-Domain Routing)表記を使用することもできます。

6.4.3. 関連情報

セットの詳細は、man ページの nft (8) の Sets セクションを参照してください。

6.5. NFTABLES コマンドにおける決定マップの使用

ディクショナリーとしても知られる決定マップにより、nft は一致基準をアクションにマッピングすることで、パケット情報に基づいてアクションを実行できます。

6.5.1. nftables での匿名マップの使用

匿名マップは、ルール内で直接使用する { match_criteria : action } ステートメントです。ステートメントには、複数のコンマ区切りマッピングを含めることができます。

匿名マップの欠点は、マップを変更する場合には、ルールを置き換える必要があることです。動的なソリューションの場合は、「nftables での名前付きマップの使用」 の説明に従って名前付きマップを使用します。

この例では、匿名マップを使用して、IPv4 プロトコルおよび IPv6 プロトコルの TCP パケットと UDP パケットの両方を別のチェーンにルーティングし、着信 TCP パケットと UDP パケットを個別に

カウントする方法を説明します。

手順6.15 nftables での匿名マップの使用

1. **example_table を作成します。**

```
# nft add table inet example_table
```

2. **example_table に tcp_packets チェーンを作成します。**

```
# nft add chain inet example_table tcp_packets
```

3. **このチェーンのトラフィックをカウントする tcp_packets にルールを追加します。**

```
# nft add rule inet example_table tcp_packets counter
```

4. **example_table で udp_packets チェーンを作成します。**

```
# nft add chain inet example_table udp_packets
```

5. **このチェーンのトラフィックをカウントする udp_packets にルールを追加します。**

```
# nft add rule inet example_table udp_packets counter
```

6. **着信トラフィックのチェーンを作成します。たとえば、example_table で着信トラフィックをフィルターする incoming_traffic という名前のチェーンを作成するには、次のコマンドを実行します。**

```
# nft add chain inet example_table incoming_traffic { type filter hook input priority 0 \; }
```

7. **匿名マップを持つルールを incoming_traffic に追加します。**

```
# nft add rule inet example_table incoming_traffic ip protocol vmap { tcp : jump
tcp_packets, udp : jump udp_packets }
```

匿名マップはパケットを区別し、プロトコルに基づいて別のカウンターチェーンに送信し

ます。

8. トラフィックカウンターの一覧を表示する場合は、example_table を表示します。

```
# nft list table inet example_table
table inet example_table {
  chain tcp_packets {
    counter packets 36379 bytes 2103816
  }

  chain udp_packets {
    counter packets 10 bytes 1559
  }

  chain incoming_traffic {
    type filter hook input priority filter; policy accept;
    ip protocol vmap { tcp : jump tcp_packets, udp : jump udp_packets }
  }
}
```

tcp_packets および udp_packets チェーンのカウンターは、受信したパケットとバイトの数の両方を表示します。

6.5.2. nftables での名前付きマップの使用

nftables フレームワークは、名前付きマップに対応します。テーブルの複数のルールでこのマップを使用できます。匿名マップに対する別の利点は、名前付きマップを使用するルールを置き換えることなく、名前付きマップを更新できることです。

名前付きマップを作成する場合は、要素のタイプを指定する必要があります。

- 一致する部分に 192.0.2.1 などの IPv4 アドレスが含まれるマップの場合は ipv4_addr。
- 一致する部分に 2001:db8:1::1 などの IPv6 アドレスが含まれるマップの場合は ipv6_addr。
- 52:54:00:6b:66: 42 など、一致する部分にメディアアクセス制御(MAC)アドレスが含まれるマップの場合は ether_addr。
- 一致する部分に tcp などのインターネットプロトコルタイプが含まれるマップの場合は

inet_proto。

- *一致する部分に ssh や 22 などのインターネットサービス名のポート番号が含まれるマップの場合は inet_service。*

- *一致する部分にパケット マーク が含まれるマップの場合は mark。パケットマークは、任意の正の 32 ビットの整数値(0 ~ 2147483647)にできます。*

- *一致する部分に カウンター 値が含まれるマップの場合は counter。カウンター値は、正の値の 64 ビットであれば任意の値にすることができます。*

- *一致する部分に クォータ 値が含まれるマップの場合は quota。クォータの値は、64 ビットの整数値にできます。*

この例では、送信元の IP アドレスに基づいて着信パケットを許可または破棄する方法を説明します。名前付きマップを使用すると、このシナリオを設定するのに必要なルールは 1 つだけで、IP アドレスとアクションがマップに動的に保存されます。この手順では、マップからエントリーを追加および削除する方法を説明します。

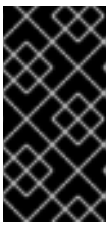
手順6.16 nftables での名前付きマップの使用

1. *テーブルを作成します。たとえば、IPv4 パケットを処理する example_table という名前のテーブルを作成するには、次のコマンドを実行します。*

```
# nft add table ip example_table
```

2. *チェーンを作成します。たとえば、example_table に、example_chain という名前のチェーンを作成するには、次のコマンドを実行します。*

```
# nft add chain ip example_table example_chain { type filter hook input priority 0 \; }
```



重要

シェルで、セミコロンがコマンドの最後として解釈されないようにするには、セミコロンをバックスラッシュでエスケープする必要があります。

3.

空のマップを作成します。たとえば、IPv4 アドレスのマップを作成するには、次のコマンドを実行します。

```
# nft add map ip example_table example_map { type ipv4_addr : verdict \; }
```

4. マップを使用するルールを作成します。たとえば、次のコマンドは、両方とも example_map で定義されている IPv4 アドレスにアクションを適用するルールを example_table の example_chain に追加します。

```
# nft add rule example_table example_chain ip saddr vmap @example_map
```

5. IPv4 アドレスと対応するアクションを example_map に追加します。

```
# nft add element ip example_table example_map { 192.0.2.1 : accept, 192.0.2.2 : drop }
```

この例では、IPv4 アドレスのアクションへのマッピングを定義します。上記で作成したルールと組み合わせて、ファイアウォールは 192.0. 2 .1 からのパケットを受け入れ、192.0. 2. からのパケットをドロップします。

6. 必要に応じて、別の IP アドレスおよび action ステートメントを追加してマップを拡張します。

```
# nft add element ip example_table example_map { 192.0.2.3 : accept }
```

7. 必要に応じて、マップからエントリーを削除します。

```
# nft delete element ip example_table example_map { 192.0.2.1 }
```

8. 必要に応じて、ルールセットを表示します。

```
# nft list ruleset
table ip example_table {
  map example_map {
    type ipv4_addr : verdict
    elements = { 192.0.2.2 : drop, 192.0.2.3 : accept }
  }

  chain example_chain {
    type filter hook input priority filter; policy accept;
```

```
    ip saddr vmap @example_map
  }
}
```

6.5.3. 関連情報

決定マップの詳細は、nft (8) man ページの Maps セクションを参照してください。

6.6. NFTABLES を使用したポート転送の設定

ポート転送を使用すると、管理者は特定の宛先ポートに送信されたパケットを、別のローカルまたはリモートポートに転送できます。

たとえば、Web サーバーにパブリック IP アドレスがない場合は、ファイアウォールのポート 80 および 443 で着信パケットを Web サーバーに転送するファイアウォールにポート転送ルールを設定できます。このファイアウォールルールを使用すると、インターネットのユーザーは、ファイアウォールの IP またはホスト名を使用して Web サーバーにアクセスできます。

6.6.1. 着信パケットの別のローカルポートへの転送

本セクションでは、ポート 8022 の着信 IPv4 パケットをローカルシステムのポート 22 に転送する方法を説明します。

手順6.17 着信パケットの別のローカルポートへの転送

1. IP アドレスファミリーを使用して、nat という名前のテーブルを作成します。

```
# nft add table ip nat
```

2. テーブルに prerouting および postrouting チェーンを追加します。

```
# nft -- add chain ip nat prerouting { type nat hook prerouting priority -100 \; }
```

注記

nft コマンドに -- オプションを渡して、シェルが優先度の負の値を nft コマンドのオプションとして解釈しないようにします。

3. 8022 ポートの着信パケットをローカルポート 22 にリダイレクトするルールを prerouting チェーンに追加します。

```
# nft add rule ip nat prerouting tcp dport 8022 redirect to :22
```

6.6.2. 特定のローカルポートで着信パケットを別のホストに転送

宛先ネットワークアドレス変換 (DNAT) ルールを使用して、ローカルポートの着信パケットをリモートホストに転送できます。これにより、インターネット上のユーザーは、プライベート IP アドレスを持つホストで実行しているサービスにアクセスできるようになります。

この手順では、ローカルポート 443 の着信 IPv4 パケットを、IP アドレス 192.0.2.1 を持つリモートシステムの同じポート番号に転送する方法を説明します。

前提条件

- パケットを転送するシステムに root ユーザーとしてログインしている。

手順6.18 特定のローカルポートで着信パケットを別のホストに転送

1. IP アドレスファミリーを使用して、nat という名前のテーブルを作成します。

```
# nft add table ip nat
```

2. テーブルに prerouting および postrouting チェーンを追加します。

```
# nft -- add chain ip nat prerouting { type nat hook prerouting priority -100 \; }
# nft add chain ip nat postrouting { type nat hook postrouting priority 100 \; }
```

注記

nft コマンドに -- オプションを渡して、シェルが優先度の負の値を nft コマンドのオプションとして解釈しないようにします。

3. ポート 443 の着信パケットを 192.0.2.1 の同じポートにリダイレクトするルールを prerouting チェーンに追加します。

```
# nft add rule ip nat prerouting tcp dport 443 dnat to 192.0.2.1
```

4. ***postrouting* チェーンにルールを追加して、送信トラフィックをマスカレードします。**

```
# nft add rule ip nat postrouting ip daddr 192.0.2.1 masquerade
```

5. **パケット転送を有効にします。**

```
# echo "net.ipv4.ip_forward=1" > /etc/sysctl.d/95-IPv4-forwarding.conf
# sysctl -p /etc/sysctl.d/95-IPv4-forwarding.conf
```

6.7. NFTABLES を使用した接続の量の制限

nftables を使用して、接続の数を制限したり、一定量の接続の確立を試みる IP アドレスをブロックして、大量のシステムリソースを使用しないようにすることができます。

6.7.1. nftables を使用した接続数の制限

nft ユーティリティーの ct count パラメーターを使用すると、管理者は接続数を制限できます。この手順では、着信接続を制限する方法の基本的な例を説明します。

前提条件

- **example_table にベースの example_chain が存在する。**

手順6.19 nftables を使用した接続数の制限

1. **IPv4 アドレスから SSH ポート(22)への 2 つの同時接続のみを許可し、同じ IP からの接続をすべて拒否するルールを追加します。**

```
# nft add rule ip example_table example_chain tcp dport ssh meter
example_meter { ip saddr ct count over 2 } counter reject
```

2. **必要に応じて、前の手順で作成した meter を表示します。**

```
# nft list meter ip example_table example_meter
table ip example_table {
 meter example_meter {
```

```
    type ipv4_addr
    size 65535
    elements = { 192.0.2.1 : ct count over 2 , 192.0.2.2 : ct count over 2  }
  }
}
```

elements エントリーは、現在ルールに一致するアドレスを表示します。この例では、elements は、SSH ポートへのアクティブな接続がある IP アドレスを一覧表示します。出力には、アクティブな接続の数を表示しないため、接続が拒否された場合は表示されないことに注意してください。

6.7.2. 1 分以内に新しい着信 TCP 接続を 11 個以上試行する IP アドレスのブロック

nftables フレームワークを使用すると、管理者はセットを動的に更新できます。このセクションでは、この機能を使用して、1 分以内に 11 個以上の IPv4 TCP 接続を確立しているホストを一時的にブロックする方法を説明します。nftables は、5 分後に拒否リストから IP アドレスを自動的に削除します。

手順6.20 1 分以内に新しい着信 TCP 接続を 11 個以上試行する IP アドレスのブロック

1. ip アドレスファミリーを使用して filter テーブルを作成します。

```
# nft add table ip filter
```

2. input チェーンを filter テーブルに追加します。

```
# nft add chain ip filter input { type filter hook input priority 0 \; }
```

3. denylist という名前のセットを filter テーブルに追加します。

```
# nft add set ip filter denylist { type ipv4_addr \; flags dynamic, timeout \; timeout 5m \; }
```

このコマンドは、IPv4 アドレスの動的セットを作成します。timeout 5m パラメーターは、nftables がセットから 5 分後にエントリーを自動的に削除することを定義します。

4. 1 分以内に新規 TCP 接続を 10 個以上確立しようとするホストのソース IP アドレスを denylist セットに自動的に追加するルールを追加します。

```
# nft add rule ip filter input ip protocol tcp ct state new, untracked limit rate over 10/minute
add @denylist { ip saddr }
```

-

5. *denylist セットの IP アドレスからの接続をすべて破棄するルールを追加します。*

```
# nft add rule ip filter input ip saddr @denylist drop
```

6.7.3. 関連情報

- *詳細は、「nftables で名前付きセットの使用」 を参照してください。*

6.8. NFTABLES ルールのデバッグ

nftables フレームワークは、管理者がルールをデバッグし、パケットが一致するかどうかを確認するためのさまざまなオプションを提供します。本セクションでは、3 つのオプションを説明します。

6.8.1. カウンターによるルールの作成

ルールが一致しているかどうかを確認するには、カウンターを使用できます。本セクションでは、カウンターを使用して新しいルールを作成する方法を説明します。

既存のルールにカウンターを追加する手順は、「既存のルールへのカウンターの追加」を参照してください。

前提条件

- *ルールを追加するチェーンが存在する。*

手順6.21 カウンターによるルールの作成

1. *counter パラメーターを使用して新しいルールをチェーンに追加します。以下の例では、ポート 22 で TCP トラフィックを許可し、このルールに一致するパケットとトラフィックをカウントするカウンターを使用するルールを追加します。*

```
# nft add rule inet example_table example_chain tcp dport 22 counter accept
```

2. *カウンター値を表示するには、次のコマンドを実行します。*

```
# nft list ruleset
```

```
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
    tcp dport ssh counter packets 6872 bytes 105448565 accept
  }
}
```

6.8.2. 既存のルールへのカウンターの追加

ルールが一致しているかどうかを確認するには、カウンターを使用できます。本セクションでは、既存のルールにカウンターを追加する方法を説明します。

カウンターで新しいルールを追加する手順は、「カウンターによるルールの作成」を参照してください。

前提条件

- カウンターを追加するルールがある。

手順6.22 既存のルールへのカウンターの追加

1. チェーンのルール (ハンドルを含む) を表示します。

```
# nft --handle list chain inet example_table example_chain
table inet example_table {
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport ssh accept # handle 4
  }
}
```

2. ルールの代わりに **counter** パラメーターを使用して カウンター を追加します。以下の例は、前の手順で表示したルールの代わりに、カウンターを追加します。

```
# nft replace rule inet example_table example_chain handle 4 tcp dport 22 counter accept
```

3. カウンター値を表示するには、次のコマンドを実行します。

```
# nft list ruleset
table inet example_table {
  chain example_chain {
    type filter hook input priority filter; policy accept;
```

```
    tcp dport ssh counter packets 6872 bytes 105448565 accept
  }
}
```

6.8.3. 既存のルールに一致するパケットの監視

nftables のトレース機能と nft monitor コマンドを組み合わせると、管理者はルールに一致するパケットを表示できます。この手順では、ルールのトレースと、このルールに一致するパケットの監視を有効にする方法を説明します。

前提条件

- カウンターを追加するルールがある。

手順6.23 既存のルールに一致するパケットの監視

1. チェーンのルール (ハンドルを含む) を表示します。

```
# nft --handle list chain inet example_table example_chain
table inet example_table {
  chain example_chain { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport ssh accept # handle 4
  }
}
```

2. ルールを置き換えてトレース機能を追加しますが、**meta nftrace set 1** パラメーターを使用します。以下の例は、前の手順で表示したルールの代わりに、トレースを有効にします。

```
# nft replace rule inet example_table example_chain handle 4 tcp dport 22 meta nftrace set 1 accept
```

3. **nft monitor** コマンドを使用してトレースを表示します。以下の例は、コマンドの出力をフィルターリングして、**inet example_table example_chain** を含むエントリーのみを表示します。

```
# nft monitor | grep "inet example_table example_chain"
trace id 3c5eb15e inet example_table example_chain packet: iif "enp1s0" ether saddr 52:54:00:17:ff:e4 ether daddr 52:54:00:72:2f:6e ip saddr 192.0.2.1 ip daddr 192.0.2.2 ip dscp cs0 ip ecn not-ect ip ttl 64 ip id 49710 ip protocol tcp ip length 60 tcp sport 56728 tcp dport ssh tcp flags == syn tcp window 64240
trace id 3c5eb15e inet example_table example_chain rule tcp dport ssh nftrace set 1 accept (verdict accept)
    ...
```

警告

nft monitor コマンドは、トレースが有効なルールの数と、一致するトラフィックの量に応じて、多くの出力を表示できます。grep などのユーティリティーを使用して出力をフィルターリングします。

第7章 システム監査

Linux の Audit システムは、システムのセキュリティー関連情報を追跡する方法を提供します。事前設定されたルールに基づき、Audit は、ログエントリーを生成し、システムで発生しているイベントに関する情報をできるだけ多く記録します。この情報は、ミッションクリティカルな環境でセキュリティーポリシーの違反者と、違反者によるアクションを判断する上で必須のものです。Audit は、追加のセキュリティー機能をシステムに提供するのではありません。システムで使用されるセキュリティーポリシーの違反を発見するために使用できます。このような違反は、SELinux などの別のセキュリティー対策で防ぐことができます。

以下は、Audit がログファイルに記録できる情報の概要です。

- イベントの日時、タイプ、結果
- サブジェクトとオブジェクトの機密性のラベル
- イベントを開始したユーザーの ID とイベントの関連性
- Audit 設定の全修正および Audit ログファイルへのアクセス試行
- SSH、Kerberos、およびその他の認証メカニズムの全使用
- /etc/passwd などの信頼できるデータベースへの変更。
- システムからの情報のインポート、およびシステムへの情報のエクスポートの試行
- ユーザー ID、サブジェクトおよびオブジェクトラベルなどの属性に基づく include または exclude イベント

Audit システムの使用は、多くのセキュリティー関連の認定における要件でもあります。Audit は、以下の認定またはコンプライアンスガイドの要件に合致するか、それを超えるように設計されています。

- *Controlled Access Protection Profile (CAPP)*
- *Labeled Security Protection Profile (LSPP)*
- *Rule Set Base Access Control (RSBAC)*
- *NISPOM (National Industrial Security Program Operating Manual)*
- *Federal Information Security Management Act (FISMA)*
- *PCI DSS (Payment Card Industry Data Security Standard)*
- *セキュリティー技術実装ガイド (Security Technical Implementation Guide (STIG))*

Audit は以下でも認定されています。

- *National Information Assurance Partnership (NIAP) および Best Security Industries (BSI) による評価*
- *Red Hat Enterprise Linux 5 における LSPP/CAPP/RSBAC/EAL4 以降の認定*
- *Red Hat Enterprise Linux 6 における OSPP/EAL4 以降 (Operating System Protection Profile / Evaluation Assurance Level 4 以降) の認定*

使用例

ファイルアクセスの監視

Audit は、ファイルやディレクトリーがアクセス、修正、または実行されたか、もしくはファイルの属性が変更されたかを追跡できます。これはたとえば、重要なファイルへのアクセスを検出し、これらのファイルが破損した場合に監査証跡を入手可能とする際に役に立ちます。

システムコールの監視

Audit は、一部のシステムコールが使用されるたびにログエントリーを生成するように設定できます。たとえば、settimeofday、clock_adjtime、およびその他の時間関連のシステムコールを監視することで、システム時間への変更を追跡するために使用できます。

ユーザーが実行したコマンドの記録

Audit はファイルが実行されたかどうかを追跡できるため、特定のコマンドの実行を毎回記録するようにルールを定義できます。たとえば、/bin ディレクトリー内のすべての実行可能ファイルにルールを定義できます。これにより作成されるログエントリーをユーザー ID で検索すると、ユーザーごとに実行されたコマンドの監査証跡を生成できます。

システムのパス名の実行の記録

ルールの呼び出し時にパスを inode に変換するファイルアクセスをウォッチする以外に、ルールの呼び出し時に存在しない場合や、ルールの呼び出し後にファイルが置き換えられた場合でも、Audit がパスの実行をウォッチできるようになりました。これにより、ルールは、プログラム実行ファイルをアップグレードした後、またはインストールされる前にも機能を継続できます。

セキュリティーイベントの記録

pam_faillock 認証モジュールは、失敗したログイン試行を記録できます。Audit で失敗したログイン試行も記録するように設定すると、ログインを試みたユーザーに関する追加情報が提供されます。

イベントの検索

Audit は ausearch ユーティリティーを提供します。これを使用すると、ログエントリーをフィルターリングし、さまざまな条件に基づいて完全な監査証跡を提供できます。

サマリーレポートの実行

aureport ユーティリティーを使用すると、記録されたイベントの日次レポートを生成できます。システム管理者は、このレポートを分析し、疑わしいアクティビティーをさらに調べることができます。

ネットワークアクセスの監視

iptables ユーティリティーおよび ebtables ユーティリティーは、Audit イベントをトリガーするように設定できます。これにより、システム管理者はネットワークアクセスを監視できます。

注記

システムのパフォーマンスは、Audit が収集する情報量によって影響される可能性があります。

7.1. AUDIT システムのアーキテクチャー

Audit システムは、ユーザー空間アプリケーションおよびユーティリティーと、カーネル側のシステムコール処理という 2 つの主要部分で設定されます。カーネルコンポーネントは、ユーザー空間アプリケーションからシステムコールを受け、これを **user**、**task**、**fstype**、*または* **exit** *のいずれかのフィルターで振り分けます。*

システムコールが **exclude** *フィルターを通過すると、前述のフィルターのいずれかに送られます。このフィルターにより、Audit ルール設定に基づいてシステムコールが Audit デーモンに送信され、さらに処理されます。*

ユーザー空間の Audit デーモンは、カーネルから情報を収集し、ログファイルのエントリーを作成します。他のユーザー空間ユーティリティーは、Audit デーモン、カーネルの Audit コンポーネント、または Audit ログファイルと相互作用します。

- *audisp - Audit dispatcher デーモンは Audit デーモンと対話し、さらに処理するためにイベントを他のアプリケーションに送信します。このデーモンの目的は、リアルタイム分析プログラムが Audit イベントと対話できるようにするためのプラグインメカニズムを提供することです。*

- *auditctl: Audit 制御ユーティリティーは、カーネル Audit コンポーネントと対話してルールを管理し、イベント生成プロセスの多くの設定およびパラメーターを制御します。*

- *残りの Audit ユーティリティーは、Audit ログファイルのコンテンツを入力として受け取り、ユーザーの要件に基づいて出力を生成します。たとえば、aureport ユーティリティーは、記録されたすべてのイベントのレポートを生成します。*

7.2. AUDIT パッケージのインストール

Audit システムを使用するには、システムに audit 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 7 では、audit パッケージ (audit および audit-libs) がデフォルトでインストールされます。これらのパッケージがインストールされていない場合は、root ユーザーで以下のコマンドを実行し、Audit とその依存パッケージをインストールしてください。

```
~]# yum install audit
```

7.3. AUDIT サービスの設定

Audit デーモンは、/etc/audit/auditd.conf ファイルで設定できます。このファイルは、Audit デーモンの動作を変更する設定パラメーターで設定されています。ハッシュ記号(#)に続く空の行とテキストは無視されます。詳細は、auditd.conf(5) の man ページをご覧ください。

7.3.1. セキュアな環境への auditd の設定

デフォルトの auditd 設定は、ほとんどの環境に適しています。ただし、環境が厳格なセキュリティーポリシーを満たす必要がある場合は、/etc/audit/auditd.conf ファイル内の Audit デーモン設定に次の設定が推奨されます。

log_file

Audit ログファイル（通常は /var/log/audit/）を保持するディレクトリーは、別のマウントポイントに存在する必要があります。これにより、その他のプロセスがこのディレクトリー内の領域を使用しないようにし、Audit デーモンの残りの領域を正確に検出します。

max_log_file

1 つの Audit ログファイルの最大サイズを指定します。これは、Audit ログファイルを保持するパーティションで利用可能な領域を完全に使用するように設定する必要があります。

max_log_file パラメーターは、最大ファイルサイズをメガバイトで指定します。指定する値は、数値にする必要があります。

max_log_file_action

max_log_file に設定された制限に達すると実行するアクションを決定します。Audit ログファイルが上書きされないように keep_logs に設定する必要があります。

space_left

space_left_action パラメーターに設定したアクションがトリガーされるディスクに残っている空き領域の量を指定します。管理者は、ディスクの領域を反映して解放するのに十分な時間を設定する必要があります。space_left の値は、Audit ログファイルが生成される速度によって異なります。

space_left の値が整数として指定される場合、絶対サイズ(MiB)として解釈されます。値が 1

から 99 までの数字とそれに続くパーセンテージ記号(5% など)として指定される場合、監査デーモンは log_file を含むファイルシステムのサイズに基づいて絶対サイズをメガバイト単位で計算します。

space_left_action

適切な通知方法を使用して、space_left_action パラメーターを email または exec に設定することが推奨されます。

admin_space_left

admin_space_left_action パラメーターに設定されたアクションがトリガーされる空き領域の絶対最小量を指定します。これは、管理者が実行するアクションをログに記録するのに十分な領域を残す値に設定する必要があります。

このパラメーターの数値は、space_left の数字よりも小さくする必要があります。また、数値にパーセント記号を追加 (1% など) して、Audit デーモンが、ディスクパーティションサイズに基づいて、数値を計算するようにすることもできます。

admin_space_left_action

single に設定してシステムをシングルユーザーモードにし、管理者がディスク領域を解放できるようにします。

disk_full_action

Audit ログファイルを保持するパーティションに空き領域がない場合に発生するアクションを指定します。halt または single に設定する必要があります。これにより、Audit がイベントをログに記録できなくなると、システムは、シングルユーザーモードでシャットダウンまたは動作します。

disk_error_action

Audit ログファイルを保持するパーティションでエラーが検出された場合に発生するアクションを指定します。ハードウェアの誤作動処理に関するローカルセキュリティーポリシーに応じて、syslog、single、または halt に設定する必要があります。

flush

incremental_async に設定する必要があります。これは freq パラメーターと組み合わせて機能し、ハードドライブとのハード同期を強制する前にディスクに送信できるレコードの数を決定します。freq パラメーターは 100 に設定する必要があります。このパラメーターにより、アクティビ

ティーが集中した際に高いパフォーマンスを保ちつつ、Audit イベントデータがディスクのログファイルと確実に同期されるようになります。

残りの設定オプションは、ローカルのセキュリティーポリシーに合わせて設定します。

7.4. AUDIT サービスの起動

auditd が設定されたら、サービスを起動して Audit 情報を収集し、ログファイルに保存します。root ユーザーで以下のコマンドを実行し、auditd を起動します。

```
~]# service auditd start
```

注記

service コマンドは、auditd デーモンと正しく対話する唯一の方法です。auid 値が適切に記録されるように、service コマンドを使用する必要があります。systemctl コマンドは、enable と status の 2 つのアクションにのみ使用できます。

システムの起動時に auditd が開始するように設定するには、次のコマンドを実行します。

```
~]# systemctl enable auditd
```

service auditd action コマンドを使用して auditd でさまざまなアクションを実行できます。ここで、action は以下のいずれかになります。

stop

auditd を停止します。

restart

auditd を再起動します。

reload または force-reload

/etc/audit/ auditd.conf ファイルから auditd の設定を再読み込みします。

rotate

/var/log/audit/ ディレクトリーのログファイルをローテーションします。

resume

Audit イベントのログが一旦停止した後、再開します。たとえば、Audit ログファイルが含まれるディスクパーティションの未使用領域が不足している場合などです。

condrestart または try-restart

auditd がすでに実行されている場合にのみ、これを再起動します。

status

auditd の実行ステータスを表示します。

7.5. AUDIT ルールの定義

Audit システムは、ログファイルに何を取得するかを定義する一連のルールで動作します。以下の種類の Audit ルールを指定することができます。

制御ルール

Audit システムの動作とその設定の一部を変更できるようにします。

ファイルシステムのルール

ファイルウォッチとも呼ばれ、特定のファイルやディレクトリーへのアクセスを監査することができます。

システムコールのルール

指定されたプログラムが行うシステムコールのログ記録を許可します。

Audit ルールは、以下のように設定できます。

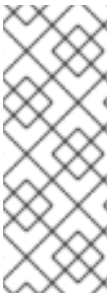
- *auditctl ユーティリティーを使用してコマンドラインで設定します。これらのルールは、再*

起動しても持続しないことに注意してください。詳細は、「auditctlを使用した Audit ルールの定義」 を参照してください。

- /etc/audit/audit.rules ファイル内。詳細は、を参照してください。 「/etc/audit/audit.rules ファイルでの永続的な監査ルールと制御の定義」

7.5.1. auditctlを使用した Audit ルールの定義

auditctl コマンドを使用すると、Audit システムの基本機能を制御し、どの Audit イベントをログに記録するかを決定するルールを定義できます。



注記

Audit サービスおよび Audit ログファイルと対話するすべてのコマンドには、root 権限が必要です。これらのコマンドは、必ず root ユーザーで実行してください。さらに、監査サービスの設定には CAP_AUDIT_CONTROL 機能が必要で、ユーザーメッセージをログに記録するために必要な CAP_AUDIT_WRITE 機能が必要です。

制御ルールの定義

以下は、Audit システムの動作を変更できるようにする制御ルールの一部です。

-b

カーネル内の既存の Audit バッファーの最大量を設定します。以下に例を示します。

```
~]# auditctl -b 8192
```

-f

重大なエラーが検出されたときに実行されるアクションを設定します。以下に例を示します。

```
~]# auditctl -f 2
```

上記の設定は、重大なエラーが発生した場合にカーネルパニックを引き起こします。

-e

Audit システムを有効または無効にするか、その設定をロックします。以下に例を示します。

```
~]# auditctl -e 2
```

上記のコマンドは、Audit 設定をロックします。

-r

1 秒あたりに生成されるメッセージのレートを設定します。以下に例を示します。

```
~]# auditctl -r 0
```

上記の設定では、生成されるメッセージにレート制限は設定されていません。

-s

Audit システムのステータスを報告します。以下に例を示します。

```
~]# auditctl -s
AUDIT_STATUS: enabled=1 flag=2 pid=0 rate_limit=0 backlog_limit=8192 lost=259 backlog=0
```

-l

現在ロードされているすべての Audit ルールを一覧表示します。以下に例を示します。

```
~]# auditctl -l
-w /etc/passwd -p wa -k passwd_changes
-w /etc/selinux -p wa -k selinux_changes
-w /sbin/insmod -p x -k module_insertion
⋮
```

-D

現在読み込まれているすべての Audit ルールを削除します。以下に例を示します。

```
~]# auditctl -D
No rules
```

ファイルシステムルールの定義

ファイルシステムルールを定義するには、以下の構文を使用します。

```
auditctl -w path_to_file -p permissions -k key_name
```

-

ここでは、以下のようになります。

- *path_to_file は、監査対象となるファイルまたはディレクトリーです。*
- *permissions は、ログに記録されるパーミッションです。*
  - *r — ファイルまたはディレクトリーへの読み取りアクセス。*
  - *w — ファイルまたはディレクトリーへの書き込みアクセス。*
  - *x — ファイルまたはディレクトリーへのアクセスを実行します。*
  - *a — ファイルまたはディレクトリーの属性の変更。*
- *key_name は、どのルールまたはルールセットが特定のログエントリーを生成したかを特定する際に役立つオプションの文字列です。*

例7.1 ファイルシステムルール

/etc/passwd ファイルへのすべての書き込みアクセスと、すべての属性変更をログに記録するルールを定義するには、以下のコマンドを実行します。

```
~]# auditctl -w /etc/passwd -p wa -k passwd_changes
```

-k オプションに続く文字列は任意であることに注意してください。

/etc/selinux/ ディレクトリー内のすべてのファイルに対するすべての書き込みアクセスと、すべての属性変更をログに記録するルールを定義するには、以下のコマンドを実行します。

```
~]# auditctl -w /etc/selinux/ -p wa -k selinux_changes
```

モジュールを Linux カーネルに挿入する /sbin/insmod コマンドの実行をログに記録するルール

を定義するには、以下のコマンドを実行します。

```
~]# auditctl -w /sbin/insmod -p x -k module_insertion
```

システムコールルールの定義

システムコールルールを定義するには、以下の構文を使用します。

```
auditctl -a action,filter -S system_call -F field=value -k key_name
```

ここでは、以下のようになります。

- action と filter は、特定のイベントがログに記録されるタイミングを指定します。action は、always または never のいずれかです。filter は、イベントに適用されるカーネルルールマッチングフィルターを指定します。rule-matching フィルターは、task、exit、ユーザー、および exclude のいずれかです。これらのフィルターの詳細については、「Audit システムのアーキテクチャー」の冒頭を参照してください。
- system_call は、名前でシステムコールを指定します。すべてのシステムコールのリストは、/usr/include/asm/unistd_64.h ファイルにあります。複数のシステムコールを 1 つのルールにまとめることができ、それぞれを独自の -S オプションの後に指定することができます。
- field=value は、指定されたアーキテクチャー、グループ ID、プロセス ID などに基づいてイベントに一致するようにルールをさらに変更する追加のオプションを指定します。利用可能なすべてのフィールドタイプとその値の完全なリストについては、auditctl(8) の man ページを参照してください。
- key_name は、どのルールまたはルールセットが特定のログエントリーを生成したかを特定する際に役立つオプションの文字列です。

例7.2 システムコールのルール

adjtimex または settimeofday システムコールがプログラムによって使用され、システムが 64 ビットアーキテクチャーを使用するたびにログエントリーを作成するルールを定義するには、以下のコマンドを使用します。

```
~]# auditctl -a always,exit -F arch=b64 -S adjtimex -S settimeofday -k time_change
```

ID が 1000 以上のシステムユーザーによってファイルが削除または名前変更されるたびにログエントリーを作成するルールを定義するには、以下のコマンドを使用します。

```
~]# auditctl -a always,exit -S unlink -S unlinkat -S rename -S renameat -F auid>=1000 -F auid!=4294967295 -k delete
```

-F auid!=4294967295 オプションが、ログイン UID が設定されていないユーザーを除外するために使用されています。

システムコールルール構文を使用して、ファイルシステムルールを定義することもできます。以下のコマンドは、**-w /etc/shadow -p wa** ファイルシステムルールに類似したシステムコールのルールを作成します。

```
~]# auditctl -a always,exit -F path=/etc/shadow -F perm=wa
```

7.5.2. 実行可能なファイルルールの定義

実行ファイルルールを定義するには、以下の構文を使用します。

```
auditctl  -a action,filter [ -F arch=cpu -S system_call] -F exe=path_to_executable_file -k key_name
```

ここでは、以下のようになります。

- **action** と **filter** は、特定のイベントがログに記録されるタイミングを指定します。**action** は、**always** または **never** のいずれかです。**filter** は、イベントに適用されるカーネルルールマッチングフィルターを指定します。**rule-matching** フィルターは、**task**、**exit**、ユーザー、および **exclude** のいずれかです。これらのフィルターの詳細については、「Audit システムのアーキテクチャー」の冒頭を参照してください。
- **system_call** は、名前でシステムコールを指定します。すべてのシステムコールのリストは、**/usr/include/asm/unistd_64.h** ファイルにあります。複数のシステムコールを 1 つのルールにまとめることができ、それぞれを独自の **-S** オプションの後に指定することができます。
- **path_to_executable_file** は、監査される実行可能ファイルへの絶対パスです。
- **key_name** は、どのルールまたはルールセットが特定のログエントリーを生成したかを特定する際に役立つオプションの文字列です。

例7.3 実行可能なファイルルール

/bin/id プログラムのすべての実行をログに記録するルールを定義するには、以下のコマンドを実行します。

```
~]# auditctl -a always,exit -F exe=/bin/id -F arch=b64 -S execve -k execution_bin_id
```

7.5.3. /etc/audit/audit.rules ファイルでの永続的な監査ルールと制御の定義

再起動後も持続する Audit ルールを定義するには、/etc/audit/audit.rules ファイルに直接追加するか、/etc/audit/rules.d/ ディレクトリーにあるルールを読み取る augenrules プログラムを使用する必要があります。/etc/audit/audit.rules ファイルは、同じ auditctl コマンドライン構文を使用してルールを指定します。ハッシュ記号(#)に続く空の行とテキストは無視されます。

auditctl コマンドを使用して、-R オプションを使用して指定されたファイルからルールを読み取ることもできます。以下に例を示します。

```
~]# auditctl -R /usr/share/doc/audit/rules/30-stig.rules
```

制御ルールの定義

ファイルには、Audit システムの動作を変更する制御ルール (-b、-D、-e、-f、-r、--loginuid-immutable、および --backlog_wait_time) のみを含めることができます。このオプションの詳細は「制御ルールの定義」 を参照してください。

例7.4 audit.rulesの制御ルール

```
# Delete all previous rules
-D

# Set buffer size
-b 8192

# Make the configuration immutable -- reboot is required to change audit rules
-e 2

# Panic when a failure occurs
-f 2

# Generate at most 100 audit messages per second
-r 100

# Make login UID immutable once it is set (may break containers)
--loginuid-immutable 1
```

ファイルシステムおよびシステムコールのルールの定義

ファイルシステムおよびシステムコールのルールは、**auditctl** 構文を使用して定義されます。「auditctlを使用した Audit ルールの定義」 の例は、以下のルールファイルで表すことができます。

例7.5 audit.rulesのファイルシステムおよびシステムコールのルール

```
-w /etc/passwd -p wa -k passwd_changes
-w /etc/selinux/ -p wa -k selinux_changes
-w /sbin/insmod -p x -k module_insertion

-a always,exit -F arch=b64 -S adjtimex -S settimeofday -k time_change
-a always,exit -S unlink -S unlinkat -S rename -S renameat -F auid>=1000 -F auid!=4294967295 -k delete
```

事前設定ルールのファイル

/usr/share/doc/audit/rules/ ディレクトリーには、**audit** パッケージがさまざまな認定標準に従って事前設定されたルールファイルのセットを提供します。

- **30-NISPOM.rules - National Industrial Security Program Operating Manual の Information System Security の章で指定されている要件を満たす Audit ルール設定。**
- **30-pci-dss-v31.rules - Payment Card Industry Data Security Standard (PCI DSS) v3.1 によって設定された要件を満たす Audit ルールの設定。**
- **30-STIG.rules - セキュリティー技術実装ガイド(STIG)によって設定された要件を満たす Audit ルールの設定。**

これらの設定ファイルを使用するには、元の **/etc/audit/audit.rules** ファイルのバックアップを作成し、選択した設定ファイルを **/etc/audit/audit.rules** ファイルにコピーします。

```
~]# cp /etc/audit/audit.rules /etc/audit/audit.rules_backup
~]# cp /usr/share/doc/audit/rules/30-stig.rules /etc/audit/audit.rules
```

注記

Audit ルールには、順序付けが可能な番号指定スキームがあります。命名スキームの詳細は、/usr/share/doc/audit/rules/README-rules ファイルを参照してください。

永続ルールを定義する augenrules の使用

augenrules スクリプトは、/etc/audit/rules.d/ ディレクトリーにあるルールを読み取り、audit.rules ファイルにコンパイルします。このスクリプトは、.rules で終わるすべてのファイルを、自然なソート順序に基づいて特定の順序で処理します。このディレクトリーのファイルは、以下の意味を持つグループに分類されます。

- *10 - カーネルおよび auditctl の設定*
- *20 - 一般的なルールに該当してしまう可能性もあるが、ユーザー側で独自ルールを作成することも可能*
- *30 - 主なルール*
- *40 - 任意のルール*
- *50 - サーバー固有のルール*
- *70 - システムのローカルルール*
- *90 - ファイナライズ (不変)*

ルールは、すべてを一度に使用することは意図されていません。これらは考慮する必要のあるポリシーの一部であり、個々のファイルは /etc/audit/rules.d/ にコピーされます。たとえば、STIG 設定でシステムを設定し、10-base-config、30-stig、31-privileged、99-finalize の各ルールをコピーします。

/etc/audit/rules.d/ ディレクトリーにルールを取得し、--load ディレクティブを指定して augenrules スクリプトを実行してそれらを読み込みます。

```
~]# augenrules --load
augenrules --load No rules
enabled 1
failure 1
pid 634
rate_limit 0
backlog_limit 8192
lost 0
backlog 0
enabled 1
failure 1
pid 634
rate_limit 0
backlog_limit 8192
lost 0
backlog 1
```

Audit ルールおよび augenrules スクリプトの詳細は、man ページの audit.rules (8)および augenrules (8) を参照し てください。

7.6. AUDIT ログファイルについて

デフォルトでは、Audit システムはログエントリーを /var/log/audit/audit.log ファイルに保存します。ログローテーションが有効な場合は、ローテーションされた audit.log ファイルは同じディレクトリーに保存されます。

以下の Audit ルールは、/etc/ssh/sshd_config ファイルの読み取りまたは変更の試行をすべてログに記録します。

```
-w /etc/ssh/sshd_config -p warx -k sshd_config
```

auditd デーモンが実行している場合は、たとえば以下のコマンドを使用して、Audit ログファイルに新しいイベントを作成します。

```
~]$ cat /etc/ssh/sshd_config
```

このイベントは、audit.log ファイルでは以下のようになります。

```
type=SYSCALL msg=audit(1364481363.243:24287): arch=c000003e syscall=2 success=no exit=-13
a0=7fffd19c5592 a1=0 a2=7fffd19c4b50 a3=a items=1 ppid=2686 pid=3538 auid=1000 uid=1000
```

```
gid=1000 euid=1000 suid=1000 fsuid=1000 egid=1000 sgid=1000 fsgid=1000 tty=pts0 ses=1
comm="cat" exe="/bin/cat" subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
key="sshd_config"
type=CWD msg=audit(1364481363.243:24287):  cwd="/home/shadowman"
type=PATH msg=audit(1364481363.243:24287): item=0 name="/etc/ssh/sshd_config" inode=409248
dev=fd:00 mode=0100600 ouid=0 ogid=0 rdev=00:00 obj=system_u:object_r:etc_t:s0
objtype=NORMAL cap_fp=none cap_fi=none cap_fe=0 cap_fver=0
type=PROCTITLE msg=audit(1364481363.243:24287) :
proctitle=636174002F6574632F7373682F737368645F636F6E666967
```

上記のイベントは 4 つのレコードで設定されており、タイムスタンプとシリアル番号を共有します。レコードは、常に **type=** キーワードで始まります。各レコードは、空白またはコンマで区切られた複数の **name=value** ペアで設定されます。上記のイベントの詳細な分析は以下のようになります。

1 つ目のレコード

type=SYSCALL

type フィールドには、レコードのタイプが含まれます。この例では、**SYSCALL** 値は、このレコードがカーネルへのシステムコールによってトリガーされることを指定しています。

利用可能なすべてのタイプ値のリストとその説明については、**Audit Record Types** を参照してください。

msg=audit(1364481363.243:24287):

msg フィールドは以下を記録します。

- **audit (time_stamp: ID)**形式のレコードのタイムスタンプと一意の**ID**。複数のレコードが同じ **Audit** イベントの一部として生成されている場合は、同じタイムスタンプおよび **ID** を共有できます。タイムスタンプは **Unix** の時間形式です **(1970** 年 **1** 月 **1** 日 **00:00:00 UTC** からの秒数**)**。
- カーネルまたはユーザー空間のアプリケーションが提供するさまざまなイベント固有の **name=value** ペア。

arch=c000003e

arch フィールドには、システムの **CPU** アーキテクチャーに関する情報が含まれます。**c000003e** の値は **16** 進数表記でエンコードされます。**ausearch** コマンドで **Audit** レコードを検索する場合は、**-i** オプションまたは **--interpret** オプションを使用して、**16** 進数の値を人間が判読できる値に自動的に変換します。**c000003e** 値は **x86_64** として解釈されます。

syscall=2

syscall フィールドは、カーネルに送信されたシステムコールのタイプを記録します。値 2 は、/usr/include/asm/unistd_64.h ファイルで人間が判読できる値と一致します。この場合、2 はオープン システムコールです。ausyscall ユーティリティーを使用すると、システムコール番号を、人間が判読できるものに変換できます。ausyscall --dump コマンドを使用して、すべてのシステムコールの一覧とその数字を表示します。詳細は、ausyscall(8) の man ページを参照してください。

success=no

success フィールドは、その特定のイベントで記録されたシステムコールが成功したか失敗したかを記録します。この例では、呼び出しが成功しませんでした。

exit=-13

exit フィールドには、システムコールによって返される終了コードを指定する値が含まれます。この値は、システムコールにより異なります。次のコマンドを実行すると、この値を人間が判読可能なものに変換できます。

```
~]# ausearch --interpret --exit -13
```

上記の例では、監査ログに終了コード -13 で失敗したイベントが含まれていることを前提としています。

a0=7fffd19c5592, a1=0, a2=7fffd19c5592, a3=a

a0 to a3 フィールドは、このイベントにおけるシステムコールの最初の 4 つの引数を 16 進数表記で記録します。これらの引数は、使用されるシステムコールによって異なります。ausearch ユーティリティーで解釈できます。

items=1

items フィールドには、syscall レコードに続く PATH 補助レコードの数が含まれます。

ppid=2686

ppid フィールドは、親プロセス ID (PPID)を記録します。この場合、2686 は bash などの親プロセスの PPID です。

pid=3538

pid フィールドは、プロセス ID (PID)を記録します。この場合、3538 は cat プロセスの PID で

す。

auid=1000

auid フィールドは、loginuid である Audit ユーザー ID を記録します。この ID は、ログイン時にユーザーに割り当てられ、ユーザーの ID が変更された場合でもすべてのプロセスに継承されます（例： su - john コマンドでユーザーアカウントの切り替え）。

uid=1000

uid フィールドは、解析しているプロセスを開始したユーザーのユーザー ID を記録します。ユーザー ID は、ausearch -i --uid UID のコマンドを使用してユーザー名に解釈できます。

gid=1000

gid フィールドは、解析しているプロセスを開始したユーザーのグループ ID を記録します。

euid=1000

euid フィールドは、解析しているプロセスを開始したユーザーの実効ユーザー ID を記録します。

suid=1000

suid フィールドは、解析しているプロセスを開始したユーザーの設定ユーザー ID を記録します。

fsuid=1000

fsuid フィールドは、解析しているプロセスを開始したユーザーのファイルシステムユーザー ID を記録します。

egid=1000

egid フィールドは、解析しているプロセスを開始したユーザーの実効グループ ID を記録します。

sgid=1000

sgid フィールドは、解析しているプロセスを開始したユーザーのセットグループ ID を記録します。

fsgid=1000

fsgid フィールドは、解析しているプロセスを開始したユーザーのファイルシステムグループ ID を記録します。

tty=pts0

tty フィールドは、解析しているプロセスが呼び出された端末を記録します。

ses=1

ses フィールドは、解析しているプロセスが呼び出されたセッションのセッション ID を記録します。

comm="cat"

comm フィールドは、解析しているプロセスを開始するために使用したコマンドのコマンドライン名を記録します。この場合、この Audit イベントをトリガーするために cat コマンドを使用します。

exe="/bin/cat"

exe フィールドは、解析しているプロセスを呼び出すために使用された実行可能ファイルへのパスを記録します。

subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023

subj フィールドは、解析しているプロセスが実行時にラベル付けされた SELinux コンテキストを記録します。

key="sshd_config"

key フィールドは、Audit ログでこのイベントを生成したルールに関連付けられた管理者定義の文字列を記録します。

2 つ目のレコード

type=CWD

2 つ目のレコードでは、type フィールドの値は CWD - 現在の作業ディレクトリーです。このタイプは、最初のレコードで指定されたシステムコールを開始したプロセスの作業ディレクトリーを記録するために使用されます。

この記録の目的は、相対パスが関連する PATH 記録に保存された場合に、現行プロセスの位置を記録することにあります。これにより、絶対パスを再構築できます。

msg=audit(1364481363.243:24287)

msg フィールドは、最初のレコードと同じタイムスタンプと ID 値を保持します。タイムスタンプは Unix の時間形式です (1970 年 1 月 1 日 00:00:00 UTC からの秒数)。

cwd="/home/user_name"

cwd フィールドには、システムコールが呼び出されたディレクトリーへのパスが含まれます。

3 つ目のレコード

type=PATH

3 つ目のレコードでは、type フィールドの値は PATH です。Audit イベントには、システムコールに引数として渡されるすべてのパスの PATH-type レコードが含まれます。この Audit イベントでは、1 つのパス(/etc/ssh/sshd_config)を引数として使用されました。

msg=audit(1364481363.243:24287):

msg フィールドは、1 番目と 2 番目のレコードと同じタイムスタンプと ID 値を保持します。

item=0

item フィールドは、SYSCALL タイプレコードで参照されているアイテムの合計数のうち、現在のレコードがどのアイテムであるかを示します。この数はゼロベースで、値 0 は最初の項目であることを示します。

name="/etc/ssh/sshd_config"

name フィールドは、システムコールに引数として渡されたファイルまたはディレクトリーのパスを記録します。ここでは、/etc/ssh/sshd_config ファイルです。

inode=409248

inode フィールドには、このイベントで記録されたファイルまたはディレクトリーに関連付けられた inode 番号が含まれます。以下のコマンドは、inode 番号 409248 に関連するファイルまたはディレクトリーを表示します。

```
~]# find / -inum 409248 -print
/etc/ssh/sshd_config
```

dev=fd:00

dev フィールドは、このイベントで記録されたファイルまたはディレクトリーを含むデバイスのマイナーおよびメジャー ID を指定します。この場合、値は /dev/fd/0 デバイスを表します。

mode=0100600

mode フィールドは、ファイルまたはディレクトリーのパーミッションを記録します。これは、st_mode フィールドの stat コマンドで返される数値表記でエンコードされます。詳細は、stat (2) の man ページを参照してください。この場合、0100600 は -rw------- として解釈できます。つまり、root ユーザーのみが /etc/ssh/sshd_config ファイルへの読み取りおよび書き込みパーミッションを持ちます。

ouid=0

ouid フィールドは、オブジェクトの所有者のユーザー ID を記録します。

ogid=0

ogid フィールドは、オブジェクトの所有者のグループ ID を記録します。

rdev=00:00

rdev フィールドには、特別なファイルに対してのみ記録されたデバイス識別子が含まれます。ここでは、記録されたファイルは通常のファイルであるため、このフィールドは使用されません。

obj=system_u:object_r:etc_t:s0

obj フィールドは、実行時に記録されたファイルまたはディレクトリーにラベル付けされた SELinux コンテキストを記録します。

objtype=NORMAL

objtype フィールドは、指定したシステムコールのコンテキストで各パスレコード操作の意図を記録します。

cap_fp=none

cap_fp フィールドは、ファイルまたはディレクトリーオブジェクトで許可されたファイルシステムベースの機能の設定に関連するデータを記録します。

cap_fi=none

cap_fi フィールドは、ファイルまたはディレクトリーオブジェクトの継承されたファイルシステムベースの機能の設定に関連するデータを記録します。

cap_fe=0

cap_fe フィールドは、ファイルまたはディレクトリーオブジェクトのファイルシステムベースの機能の有効ビットの設定を記録します。

cap_fver=0

cap_fver フィールドは、ファイルまたはディレクトリーオブジェクトのファイルシステムベースの機能のバージョンを記録します。

4 つ目のレコード

type=PROCTITLE

type フィールドには、レコードのタイプが含まれます。この例では、PROCTITLE 値は、このレコードがカーネルへのシステムコールによってトリガーされたこの Audit イベントをトリガーした完全なコマンドラインを提供することを指定しています。

proctitle=636174002F6574632F7373682F737368645F636F6E666967

proctitle フィールドは、解析しているプロセスを開始するために使用したコマンドのコマンドラインを記録します。このフィールドは 16 進数の表記で記録され、Audit ログパーサーに影響が及ばないようにします。このテキストは、この Audit イベントを開始したコマンドに復号します。ausearch コマンドで Audit レコードを検索する場合は、-i オプションまたは --interpret オプションを使用して、16 進数の値を人間が判読できる値に自動的に変換します。636174002F6574632F7373682F737368645F636F6E666967 値は cat /etc/ssh/sshd_config として解釈されます。

上記で分析した Audit イベントには、イベントに含めることができるすべての可能なフィールドのサブセットのみが含まれています。すべてのイベントフィールドのリストとその説明については、Audit Event Fields を参照してください。すべてのイベントタイプのリストとその説明については、Audit Record Types を参照してください。

> *例7.6 追加の audit.log イベント*
>
> *以下の Audit イベントは、auditd デーモンが正常に起動したことを記録します。ver フィールド*

は、起動した Audit デーモンのバージョンを示します。

```
type=DAEMON_START msg=audit(1363713609.192:5426): auditd start, ver=2.2 format=raw
kernel=2.6.32-358.2.1.el6.x86_64 auid=1000 pid=4979 subj=unconfined_u:system_r:auditd_t:s0
res=success
```

以下の Audit イベントは、UID が 1000 のユーザーが root ユーザーとしてログインしようとして失敗したことを記録します。

```
type=USER_AUTH msg=audit(1364475353.159:24270): user pid=3280 uid=1000 auid=1000
ses=1 subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
msg='op=PAM:authentication acct="root" exe="/bin/su" hostname=? addr=? terminal=pts/0
res=failed'
```

7.7. AUDIT ログファイルの検索

ausearch ユーティリティーを使用すると、Audit ログファイルで特定のイベントを検索できます。デフォルトでは、ausearch は /var/log/audit/audit.log ファイルを検索します。ausearch options -if file_nameコマンドを使用して、別のファイル を指定できます。1 つの ausearch コマンドで複数のオプションを指定することは、フィールドタイプ間で AND 演算子を使用し、同じフィールドタイプの複数のインスタンス間で OR 演算子を使用することと同じです。

例7.7 ausearch を使用した Audit ログファイルの検索

/var/log/audit/audit.log ファイルで失敗したログイン試行を検索するには、次のコマンドを使用します。

```
~]# ausearch --message USER_LOGIN --success no --interpret
```

すべてのアカウント、グループ、およびロールの変更を検索するには、以下のコマンドを使用します。

```
~]# ausearch -m ADD_USER -m DEL_USER -m ADD_GROUP -m USER_CHAUTHTOK -m
DEL_GROUP -m CHGRP_ID -m ROLE_ASSIGN -m ROLE_REMOVE -i
```

ユーザーのログイン ID (auid)を使用して、特定のユーザーが実行したすべてのログに記録されたアクションを検索するには、次のコマンドを使用します。

```
~]# ausearch -ua 1000 -i
```

昨日から現時点までの失敗したシステムコールをすべて検索するには、以下のコマンドを使用します。

```
~]# ausearch --start yesterday --end now -m SYSCALL -sv no -i
```

すべての **ausearch** オプションの完全なリストは、**ausearch(8)** の **man** ページを参照してください。

7.8. AUDIT レポートの作成

aureport ユーティリティーを使用すると、**Audit** ログファイルに記録されたイベントに関する要約レポートと列ャーレポートを生成できます。デフォルトでは、**/var/log/audit/** ディレクトリー内のすべての **audit.log** ファイルがレポートを作成するためにクエリーされます。**aureport options -if file_name**コマンドを使用して、レポートを実行する別のファイル を指定できます。

例7.8 aureport を使用した Audit レポートの生成

現在のサンプル日を除く過去 **3** 日間のログイベントのレポートを生成するには、以下のコマンドを使用します。

```
~]# aureport --start 04/08/2013 00:00:00 --end 04/11/2013 00:00:00
```

すべての実行ファイルイベントのレポートを生成するには、以下のコマンドを使用します。

```
~]# aureport -x
```

上記の実行ファイルイベントレポートのサマリーを生成するには、以下のコマンドを使用します。

```
~]# aureport -x --summary
```

すべてのユーザーの失敗したイベントの要約レポートを生成するには、以下のコマンドを使用します。

```
~]# aureport -u --failed --summary -i
```

各システムユーザーごとに、失敗したすべてのログイン試行の要約レポートを生成するには、以下のコマンドを使用します。

```
~]# aureport --login --summary -i
```

ユーザー ID 1000 のすべてのファイルアクセスイベントを検索する ausearch クエリーからレポートを生成するには、以下のコマンドを使用します。

```
~]# ausearch --start today --loginuid 1000 --raw | aureport -f --summary
```

クエリーされたすべての Audit ファイルとそれらに含まれるイベントの時間範囲のレポートを生成するには、以下のコマンドを使用します。

```
~]# aureport -t
```

すべての aureport オプションの完全なリストは、aureport(8) の man ページを参照してください。

7.9. 関連情報

Audit システムの詳細は、以下の資料を参照してください。

オンラインのリソース

- RHEL Audit System Refence: https://access.redhat.com/articles/4409591
- Linux Audit ドキュメントのプロジェクトページ (https://github.com/linux-audit/audit-documentation/wiki)

インストールされているドキュメント

audit パッケージが提供するドキュメントは、/usr/share/doc/audit/ ディレクトリーにあります。

man ページ

- ***audispd.conf(5)***
- ***auditd.conf(5)***
- ***ausearch-expression(5)***
- ***audit.rules(7)***
- ***audispd(8)***
- ***auditctl(8)***
- ***auditd(8)***
- ***aulast(8)***
- ***aulastlog(8)***
- ***aureport(8)***
- ***ausearch(8)***
- ***ausyscall(8)***
- ***autrace(8)***
- ***auvirt(8)***

第8章 設定コンプライアンスおよび脆弱性スキャンの開始

コンプライアンス監査は、指定したオブジェクトが、コンプライアンスポリシーに指定されているすべてのルールに従っているかどうかを判断するプロセスです。コンプライアンスポリシーは、コンピューティング環境で使用される必要な設定を指定するセキュリティー専門家が定義します。これは多くの場合は、チェックリストの形式を取ります。

コンプライアンスポリシーは組織により大幅に異なることがあり、同一組織内でもシステムが異なるとポリシーが異なる可能性があります。ポリシーは、各システムの目的や、組織におけるシステム重要性により異なります。カスタマイズしたソフトウェア設定や導入の特徴によっても、カスタマイズしたポリシーのチェックリストが必要になってきます。

8.1. RHEL における設定コンプライアンスツール

Red Hat Enterprise Linux は、完全に自動化されたコンプライアンス監査を可能にするツールを提供します。このツールは SCAP (Security Content Automation Protocol) 規格に基づいており、コンプライアンスポリシーの自動化に合わせるように設計されています。

- SCAP Workbench - scap-workbench グラフィカルユーティリティーは、単一のローカルシステムまたはリモートシステムで設定スキャンと脆弱性スキャンを実行するように設計されています。これらのスキャンと評価に基づくセキュリティーレポートの生成にも使用できます。

- OpenSCAP: OpenSCAP ライブラリーは、付随する oscap コマンドラインユーティリティーとともに、ローカルシステムで設定スキャンと脆弱性スキャンを実行し、設定コンプライアンスコンテンツを検証し、これらのスキャンおよび評価に基づいてレポートおよびガイドを生成するように設計されています。

- SCAP セキュリティーガイド(SSG) - scap-security-guide パッケージは、Linux システムのセキュリティーポリシーの最新コレクションを提供します。このガイダンスは、セキュリティー強化に関する実践的なアドバイスのカタログから設定されます (該当する場合は、法規制要件へのリンクが含まれます)。このプロジェクトは、一般的なポリシー要件と特定の実装ガイドラインとの間にあるギャップを埋めることを目的としています。

- Script Check Engine (SCE) - SCE は SCAP プロトコルの拡張であり、管理者は Bash、Python、Ruby などのスクリプト言語を使用してセキュリティーコンテンツを作成できます。SCE 拡張機能は、openscap-engine-sce パッケージで提供されます。SCE 自体は SCAP 環境の一部ではありません。

複数のリモートシステムで自動コンプライアンス監査を実行する必要がある場合は、Red Hat

Satellite 用の OpenSCAP ソリューションを利用できます。

関連情報

- oscap (8): oscap コマンドラインユーティリティーの man ページは、利用可能なオプションとその使用方法に関する完全な一覧を提供します。
- Red Hat Security Demos: Creating Customized Security Policy Content to Automate Security Compliance - Red Hat Enterprise Linux に含まれるツールを使用してセキュリティーコンプライアンスを自動化し、業界標準のセキュリティーポリシーとカスタムセキュリティーポリシーの両方に準拠するためのハンズオンラボ。トレーニングや、このラボ演習へのアクセスを希望する場合は、Red Hat アカウントチームにお問い合わせください。
- Red Hat Security Demos: Defend Yourself with RHEL Security Technologies - OpenSCAP を含む Red Hat Enterprise Linux で利用可能な主要なセキュリティー技術を使用して、RHEL システムの全レベルでセキュリティーを実装する方法を学ぶためのハンズオンラボ。トレーニングや、このラボ演習へのアクセスを希望する場合は、Red Hat アカウントチームにお問い合わせください。
- scap-workbench (8) - SCAP Workbench アプリケーションの man ページでは、アプリケーションの基本情報と、SCAP コンテンツの潜在的なソースへのリンクが提供されます。
- scap-security-guide (8)- scap-security-guide プロジェクトの man ページでは、利用可能な SCAP セキュリティープロファイルに関する詳細が記載されています。また、OpenSCAP ユーティリティーを使用して提供されたベンチマークを使用する例も含まれています。
- Security Compliance Management in the Administering Red Hat Satellite Guide には、Red Hat Satellite で OpenSCAP を使用する方法の詳細が記載されています。

8.2. 脆弱性スキャン

8.2.1. Red Hat Security Advisories OVAL フィード

Red Hat Enterprise Linux のセキュリティー監査機能は、標準規格セキュリティー設定共通化手順 (Security Content Automation Protocol (SCAP)) を基にしています。SCAP は、自動化された設定、脆弱性およびパッチの確認、技術的な制御コンプライアンスアクティビティー、およびセキュリティーの測定に対応している多目的な仕様のフレームワークです。

SCAP 仕様は、スキャナーまたはポリシーエディターの実装が義務付けられていなくても、セキュ

リティーコンテンツの形式がよく知られて標準化されているエコシステムを作成します。これにより、組織は、採用しているセキュリティーベンダーの数に関係なく、セキュリティーポリシー (SCAP コンテンツ) を構築するのは一度で済みます。

セキュリティー検査言語 OVAL (Open Vulnerability Assessment Language) は、SCAP に不可欠で最も古いコンポーネントです。その他のツールやカスタマイズされたスクリプトとは異なり、OVAL は、宣言型でリソースが必要な状態を記述します。OVAL コードは、スキャナーと呼ばれる OVAL インタープリターツールを使用して直接実行されることは決してありません。OVAL が宣言型であるため、評価されるシステムの状態が偶然修正されることはありません。

他のすべての SCAP コンポーネントと同様に、OVAL は XML に基づいています。SCAP 標準規格は、いくつかのドキュメント形式を定義します。この形式にはそれぞれ異なる種類の情報が記載され、異なる目的に使用されます。

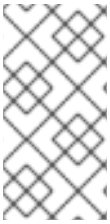
Red Hat 製品セキュリティー を使用すると、Red Hat 製品をお使いのお客様に影響を及ぼすセキュリティー問題をすべて追跡して調査します。Red Hat カスタマーポータルで簡潔なパッチやセキュリティーアドバイザリーを適時提供します。Red Hat は OVAL パッチ定義を作成してサポートし、マシンが判読可能なセキュリティーアドバイザリーを提供します。

プラットフォーム、バージョン、およびその他の要因が異なるため、Red Hat 製品セキュリティー 質的な脆弱性の重大度評価 は、サードパーティーが提供する Common Vulnerability Scoring System (CHC) のベースライン評価と完全に一致しているわけではありません。したがって、サードパーティーが提供する定義ではなく、RHSA OVAL 定義を使用することが推奨されます。

RHSA OVAL 定義 は、個別に完全なパッケージとして利用可能であり、RedHat カスタマーポータルで新しいセキュリティーアドバイザリーが利用可能になってから 1 時間以内に更新されます。

各 OVAL パッチ定義は、Red Hat セキュリティーアドバイザリー (RHSA) と 1 対 1 にマッピングしています。RHSA には複数の脆弱性に対する修正が含まれるため、各脆弱性は、共通脆弱性識別子 (Common Vulnerabilities and Exposures (CVE)) 名ごとに表示され、公開バグデータベースの該当箇所へのリンクが示されます。

RHSA OVAL 定義は、システムにインストールされている RPM パッケージで脆弱なバージョンを確認するように設計されています。この定義は拡張でき、パッケージが脆弱な設定で使用されているかどうかを見つけるなど、さらに確認できるようにすることができます。この定義は、Red Hat が提供するソフトウェアおよび更新に対応するように設計されています。サードパーティーソフトウェアのパッチ状態を検出するには、追加の定義が必要です。



注記

コンテナーやコンテナーイメージのセキュリティーの脆弱性をスキャンするには、「コンテナーおよびコンテナーイメージの脆弱性スキャン」 を参照してください。

関連情報

- Red Hat and OVAL compatibility
- Red Hat and CVE compatibility
- 製品セキュリティーの概要 の 通知およびアドバイザリー
- Security Data Metrics
- 「コンテナーおよびコンテナーイメージの脆弱性スキャン」

8.2.2. システムの脆弱性のスキャン

oscap コマンドラインユーティリティーを使用すると、ローカルシステムをスキャンし、設定コンプライアンスコンテンツを検証し、スキャンおよび評価に基づいてレポートおよびガイドを生成します。このユーティリティーは、**OpenSCAP** ライブラリーのフロントエンドとしてサービスを提供し、その機能を処理する **SCAP** コンテンツのタイプに基づいてモジュール (サブコマンド) にグループ化します。

手順

1. **openscap-scanner** パッケージおよび **bzip2** パッケージをインストールします。

```
~]# yum install openscap-scanner bzip2
```

2. お使いのシステムに対応した最新の **RHSA OVAL** 定義ファイルをダウンロードします。以下に例を示します。

```
~]# wget -O - https://www.redhat.com/security/data/oval/v2/RHEL7/rhel-7.oval.xml.bz2 |
bzip2 --decompress > rhel-7.oval.xml
```

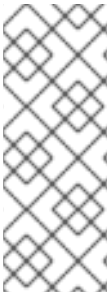
3. システムの脆弱性をスキャンし、**vulnerability.html** ファイルに結果を保存します。

```
~]# oscap oval eval --report vulnerability.html rhel-7.oval.xml
```

検証

1. 結果をブラウザーで確認します。以下に例を示します。

```
~]$ firefox vulnerability.html &
```



注記

CVE OVAL チェックは脆弱性を検索します。したがって、"True" いう結果は、システムに脆弱性があることを意味し、"False" という結果は、スキャンによって脆弱性が発見されなかったことを意味します。HTML レポートでは、これは結果の行の色によってさらに区別されます。

関連情報

- *man ページの oscap (8)*
- *Red Hat OVAL definitions の一覧*

8.2.3. リモートシステムの脆弱性のスキャン

SSH プロトコルで oscap-ssh ツールを使用して、OpenSCAP スキャナーでリモートシステムの脆弱性を確認することもできます。

前提条件

- *リモートシステムに openscap-scanner パッケージがインストールされている。*
- *リモートシステムで SSH サーバーが実行している。*

手順

1.

1. **openscap-utils パッケージおよび bzip2 パッケージをインストールします。**

```
~]# yum install openscap-utils bzip2
```

2. **システムに最新 RHSA OVAL 定義をダウンロードします。**

```
~]# wget -O - https://www.redhat.com/security/data/oval/v2/RHEL7/rhel-7.oval.xml.bz2 |
bzip2 --decompress > rhel-7.oval.xml
```

3. **脆弱性に対して、ホスト名 machine1、ポート 22 で実行する SSH、およびユーザー名 joesec でリモートシステムをスキャンし、結果を remote-vulnerability.html ファイルに保存します。**

```
~]# oscap-ssh joesec@machine1 22 oval eval --report remote-vulnerability.html rhel-
7.oval.xml
```

関連情報

- **man ページの oscap-ssh (8)**
- **Red Hat OVAL definitions の一覧**

8.3. 設定コンプライアンススキャン

8.3.1. RHEL 7 の設定コンプライアンス

設定コンプライアンススキャンを使用して、特定の組織で定義されているベースラインに準拠できます。たとえば、米国政府と協力している場合は、Operating System Protection Profile (OSPP) に準拠し、支払い処理業者の場合は Payment Card Industry Data Security Standard (PCI-DSS) に準拠しなければならない場合があります。設定コンプライアンススキャンを実行して、システムセキュリティーを強化することもできます。

Red Hat は、影響を受けるコンポーネントに対する Red Hat のベストプラクティスに従っているため、SCAP Security Guide パッケージで提供される Security Content Automation Protocol (SCAP) コンテンツに従うことを推奨します。

SCAP Security Guide パッケージは、SCAP 1.2 および SCAP 1.3 標準規格に準拠するコンテンツを提供します。openscap scanner ユーティリティーは、SCAP Security Guide パッケージで提供さ

れる SCAP 1.2 および SCAP 1.3 コンテンツの両方と互換性があります。

重要

設定コンプライアンススキャンを実行しても、システムが準拠しているとは限りません。

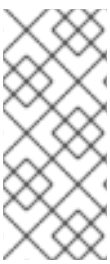
SCAP セキュリティーガイドスイートは、データストリームのドキュメント形式で、複数のプラットフォームのプロファイルを提供します。データストリームは、定義、ベンチマーク、プロファイル、および個々のルールが含まれるファイルです。各ルールでは、コンプライアンスの適用性と要件を指定します。RHEL 7 は、セキュリティーポリシーを扱う複数のプロファイルを提供します。Red Hat データストリームには、業界標準の他に、失敗したルールの修正に関する情報も含まれます。

コンプライアンススキャンリソースの構造

```
Data stream
  ├── xccdf
/     ├── benchmark
/           ├── profile
/               ├──rule
/                   ├── xccdf
/                       ├── oval reference
  ├── oval                  ├── ocil reference
  ├── ocil                  ├── cpe reference
  └── cpe                   └── remediation
```

このプロファイルは、OSPP (Operating System Protection Profile)、PCI-DSS (Payment Card Industry Data Security Standard) などのセキュリティーポリシーに基づく一連のルールです。これにより、セキュリティー標準規格に準拠するために、システムを自動で監査できます。

プロファイルを変更 (調整) して、パスワードの長さなどの特定のルールをカスタマイズできます。プロファイルの調整の詳細は、「SCAP Workbench を使用したセキュリティープロファイルのカスタマイズ」を参照してください



注記

コンテナーまたはコンテナーイメージをスキャンして設定のコンプライアンスを確認するには、「コンテナーおよびコンテナーイメージの脆弱性スキャン」 を参照してください。

8.3.2. OpenSCAP スキャン結果の例

システムのさまざまなプロパティーと、OpenSCAP スキャンに適用されるデータストリームおよびプロファイルによっては、ルールごとに固有の結果が生成されることがあります。以下は、考えられる結果のリストで、その意味を簡単に説明します。

表8.1 OpenSCAP スキャン結果の例

結果	説明
Pass	スキャンでは、このルールとの競合が見つかりませんでした。
Fail	スキャンで、このルールとの競合が検出されました。
Not checked	OpenSCAP はこのルールの自動評価を実行しません。システムがこのルールに手動で準拠しているかどうかを確認してください。
Not applicable	このルールは、現在の設定には適用されません。
Not selected	このルールはプロファイルには含まれません。OpenSCAP はこのルールを評価せず、結果にこのようなルールは表示されません。
Error	スキャンでエラーが発生しました。詳細は、--**verbose DEVEL オプションを指定して oscap- scanner** コマンドを入力できます。バグレポート を作成することを検討してください。
Unknown	スキャンで予期しない状況が発生しました。詳細は、--**verbose DEVEL オプションを指定して oscap- scanner** コマンドを入力できます。バグレポート を作成することを検討してください。

8.3.3. 設定コンプライアンスのプロファイルの表示

スキャンまたは修復にプロファイルを使用することを決定する前に、それらを一覧表示し、oscap info サブコマンドを使用して詳細な説明を確認できます。

前提条件

- *openscap-scanner パッケージおよび scap-security-guide パッケージがインストールされている。*

手順

1. *SCAP Security Guide プロジェクトが提供する設定コンプライアンスプロファイルで、利用可能なファイルをすべて一覧表示します。*

```
~]$ ls /usr/share/xml/scap/ssg/content/
ssg-firefox-cpe-dictionary.xml  ssg-rhel6-ocil.xml
ssg-firefox-cpe-oval.xml        ssg-rhel6-oval.xml
...
ssg-rhel6-ds-1.2.xml            ssg-rhel8-xccdf.xml
ssg-rhel6-ds.xml
...
```

2. *oscap info サブコマンドを使用して、選択したデータストリームに関する詳細情報を表示します。データストリームを含む XML ファイルは、名前に -ds 文字列で示されます。Profiles セクションで、利用可能なプロファイルとその ID の一覧を確認できます。*

```
~]$ oscap info /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
...
Profiles:
 Title: PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7
  Id: xccdf_org.ssgproject.content_profile_pci-dss
 Title: OSPP - Protection Profile for General Purpose Operating Systems v. 4.2.1
  Id: xccdf_org.ssgproject.content_profile_ospp
...
```

3. *データストリームファイルからプロファイルを選択し、選択したプロファイルに関する追加情報を表示します。これを行うには、--profile オプションと、前のコマンドの出力に表示される ID の接尾辞を指定して oscap info を使用します。たとえば、PCI-DSS プロファイルの ID は xccdf_org.ssgproject.content_profile_pci-dss で、--profile オプションの値は _pci-dss です。*

```
~]$ oscap info --profile _pci-dss /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
...
Profile
 Title: PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7
 Id: xccdf_org.ssgproject.content_profile_pci-dss

 Description: Ensures PCI-DSS v3.2.1 related security configuration settings are applied.
...
```

4. *または、GUI を使用する場合は scap-security-guide-doc パッケージをインストールし、Web ブラウザーで file:///usr/share/doc/scap-security-guide-doc-0.1.46/ssg-rhel7-guide-index.html ファイルを開きます。*Guide to the Secure Configuration of Red Hat Enterprise Linux 7 *ドキュメントの右上のフィールドで必要なプロファイルを選択すると、後続の評価のために、関連するコマンドに ID がすでに含まれていることを確認できます。*

関連情報

- man ページの scap-security-guide (8) には、プロファイルの一覧も含まれています。

8.3.4. 特定のベースラインによる設定コンプライアンスの評価

システムが特定のベースラインに準拠しているかどうかを確認するには、次の手順に従います。

前提条件

- openscap-scanner パッケージおよび scap-security-guide パッケージがインストールされている。
- システムが準拠する必要があるベースライン内のプロファイルの ID を知っている必要があります。ID を見つけるには、「設定コンプライアンスのプロファイルの表示」を参照してください。

手順

1. 選択したプロファイルでそのシステムがどのように複雑であるかを評価し、スキャン内容を保存すると、以下のように HTML ファイル (report.html) に結果が表示されます。

```
~]$ sudo oscap xccdf eval --report report.html --profile ospp
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

2. オプション: ホスト名 machine1、ポート 22 で実行されている SSH、およびユーザー名 joesec を使用してリモートシステムをスキャンし、脆弱性を確認して、結果を remote-report.html ファイルに保存します。

```
~]$ oscap-ssh joesec@machine1 22 xccdf eval --report remote_report.html --profile ospp
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

関連情報

- scap-security-guide (8)の man ページ
- file:///usr/share/doc/scap-security-guide-doc-0.1.46/ ディレクトリーにインストールされている SCAP セキュリティーガイド のドキュメント。
-

scap-security-guide-doc パッケージでインストールされている*Guide to the Secure Configuration of Red Hat Enterprise Linux 7*

8.4. 特定のベースラインに合わせたシステムの修復

この手順を使用して、特定のベースラインに合わせて RHEL 7 システムを修正します。この例では、OSPP (Protection Profile for General Purpose Operating Systems) の保護プロファイルを使用します。

警告

Remediate オプションを有効にしてシステム評価を慎重に実行しないと、システムが機能しなくなることがあります。Red Hat は、セキュリティーを強化した修正で加えられた変更を元に戻す自動手段は提供していません。修復は、デフォルト設定の RHEL システムで対応しています。インストール後にシステムが変更した場合は、修正を実行しても、必要なセキュリティープロファイルに準拠しない場合があります。

前提条件

- RHEL 7 システムに、*scap-security-guide* パッケージがインストールされている。

手順

1. *oscap* コマンドに *--remediate* オプションを指定して使用します。

```
~]$ sudo oscap xccdf eval --profile ospp --remediate /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

2. システムを再起動します。

検証

1. システムの OSPP プロファイルへのコンプライアンスを評価し、スキャン結果を *ospp_report.html* ファイルに保存します。

```
~]$ oscap xccdf eval --report ospp_report.html --profile ospp
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

関連情報

- *scap-security-guide (8) および oscap (8)の man ページ*

8.5. SSG ANSIBLE PLAYBOOK を使用して特定のベースラインに合わせるようにシステムを修正

この手順を使用して、SCAP Security Guide プロジェクトの Ansible Playbook ファイルを使用して、特定のベースラインでシステムを修正します。この例では、OSPP (Protection Profile for General Purpose Operating Systems) の保護プロファイルを使用します。

警告

Remediate オプションを有効にしてシステム評価を慎重に実行しないと、システムが機能しなくなることがあります。Red Hat は、セキュリティーを強化した修正で加えられた変更を元に戻す自動手段は提供していません。修復は、デフォルト設定の RHEL システムで対応しています。インストール後にシステムが変更した場合は、修正を実行しても、必要なセキュリティープロファイルに準拠しない場合があります。

前提条件

- RHEL 7 システムに、scap-security-guide パッケージがインストールされている。
- ansible パッケージがインストールされている。詳細は、Ansible インストールガイドを参照してください。

手順

1. Ansible を使用して、OSPP に合わせてシステムを修正します。

```
~]# ansible-playbook -i localhost, -c local /usr/share/scap-security-guide/ansible/ssg-rhel7-
role-ospp.yml
```

2. システムを再起動します。

検証

1. システムの OSPP プロファイルへのコンプライアンスを評価し、スキャン結果を ospp_report.html ファイルに保存します。

```
~]# oscap xccdf eval --profile ospp --report ospp_report.html
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

関連情報

- scap-security-guide (8) および oscap (8)の man ページ
- Ansible ドキュメント

8.6. システムを特定のベースラインに合わせるための修復用 ANSIBLE PLAYBOOK の作成

以下の手順に従って、必要な修復のみを含む Ansible Playbook を作成し、システムを特定のベースラインに合わせます。この例では、OSPP (Protection Profile for General Purpose Operating Systems) の保護プロファイルを使用します。この手順では、すでに対応済みの要件は含めずに、より小規模な Playbook を作成します。以下の手順に従うと、システムを変更せずに、後のアプリケーション用にファイルの準備を行うだけです。

前提条件

- システムに、scap-security-guide パッケージがインストールされている。

手順

1. システムをスキャンして結果を保存します。

```
~]# oscap xccdf eval --profile ospp --results ospp-results.xml
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

2. 前の手順で生成されたファイルに基づいて Ansible Playbook を生成します。

```
~]# oscap xccdf generate fix --fix-type ansible --profile ospp --output ospp-
remediations.yml ospp-results.xml
```

3. *ospp-remediations.yml ファイルには、手順 1 で実行されたスキャン中に失敗したルールの Ansible 修復が含まれます。この生成されたファイルを確認したら、ansible-playbook ospp-remediations.yml コマンドを使用して適用できます。*

検証

1. *お使いのテキストエディターで、手順 1 で実行したスキャンで失敗したルールが ospp-remediations.yml ファイルに含まれていることを確認します。*

関連情報

- *scap-security-guide (8) および oscap (8)の man ページ*
- *Ansible ドキュメント*

8.7. SCAP WORKBENCH を使用したカスタムプロファイルでシステムのスキャン

SCAP Workbench は、単一のローカルシステムまたはリモートシステムで設定スキャンを実行し、システムの修復を実行し、スキャン評価に基づいてレポートを生成できるグラフィカルユーティリティーです。SCAP Workbench には、oscap コマンドラインユーティリティーと比較して、機能が限定されていることに注意してください。SCAP Workbench は、データストリームファイルの形式でセキュリティーコンテンツを処理します。

8.7.1. SCAP Workbench を使用したシステムのスキャンおよび修復

選択したセキュリティーポリシーに照らしてシステムを評価するには、次の手順を使用します。

前提条件

- *scap-workbench パッケージがシステムにインストールされている。*

手順

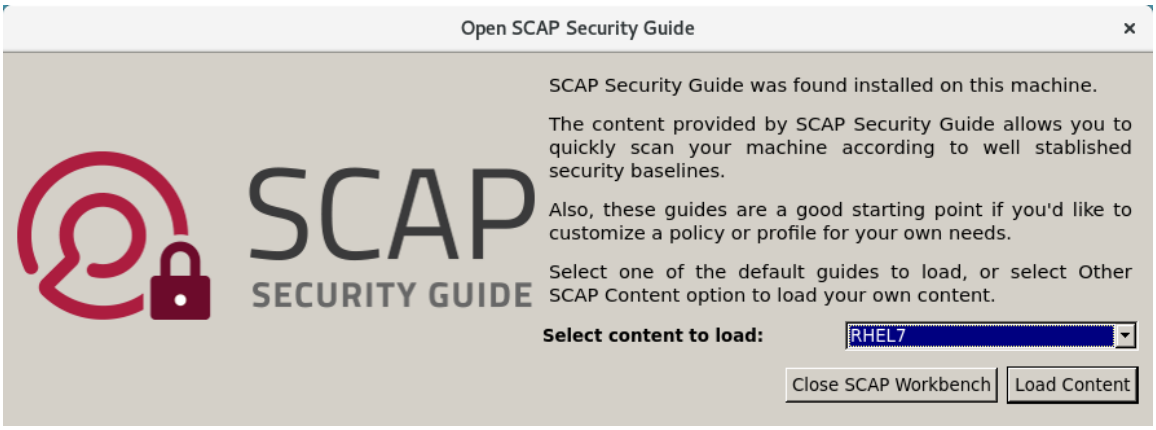
1. *GNOME クラシック デスクトップ環境から SCAP Workbench を実行するには、Super キーを押してアクティビティーの 概要に入り、scap-workbench と 入力 して Enter キーを押*

します。または、次のコマンドを実行します。

```
~]$ scap-workbench &
```

2. *次のオプションのいずれかを使用して、セキュリティーポリシーを選択します。*

   - *開始ウィンドウの **Load Content** ボタン*
   - ***Open content from SCAP Security Guide***
   - ***File** メニューで **Other Content** を開き、該当する **XCCDF**、**SCAP RPM**、またはデータストリームファイルを検索します。*

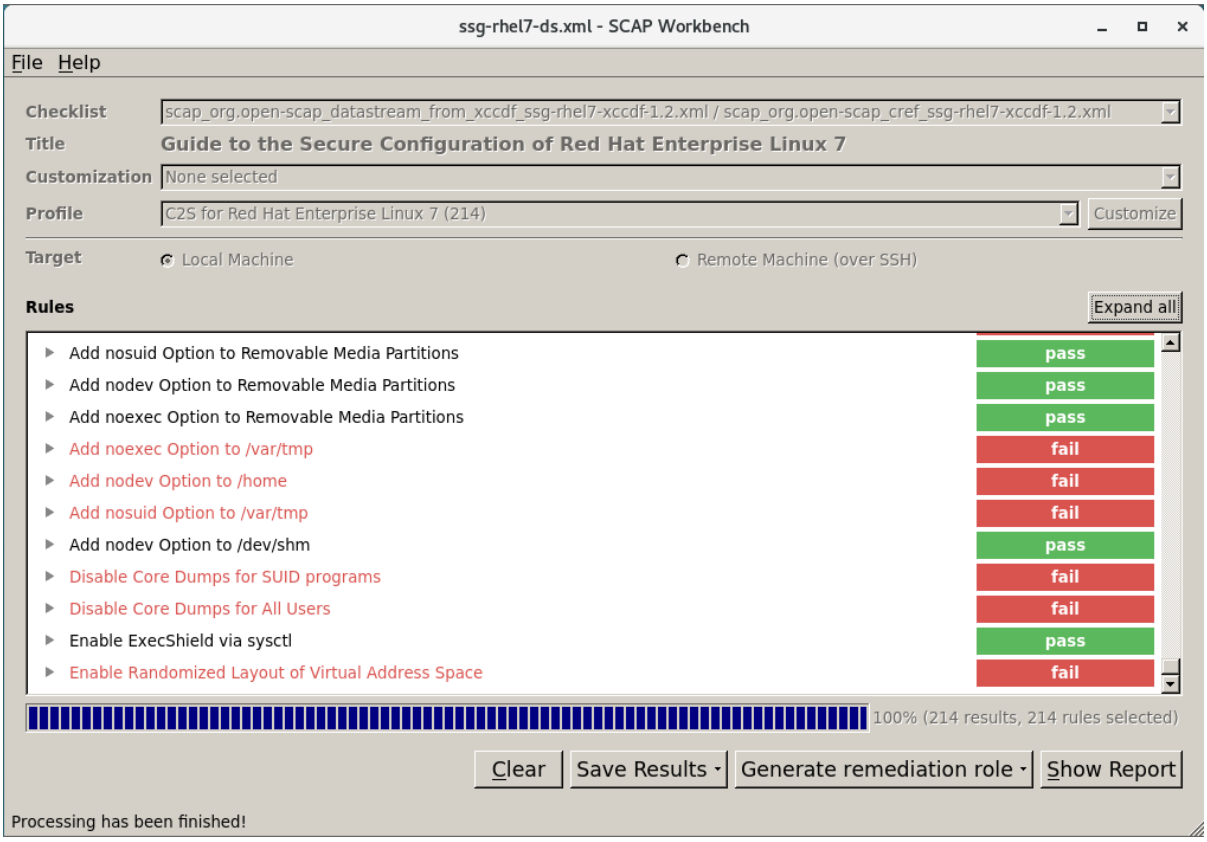


3. ***Remediate** チェックボックスを選択して、システム設定の自動修正を有効にすることができます。このオプションを有効にすると、**SCAP Workbench** は、ポリシーにより適用されるセキュリティールールに従ってシステム設定の変更を試みます。このプロセスは、システムスキャン中に失敗した関連チェックを修正しようとします。*

警告

Remediate オプションを有効にしてシステム評価を慎重に実行しないと、システムが機能しなくなることがあります。Red Hat は、セキュリティーを強化した修正で加えられた変更を元に戻す自動手段は提供していません。修復は、デフォルト設定の RHEL システムで対応しています。インストール後にシステムが変更した場合は、修正を実行しても、必要なセキュリティープロファイルに準拠しない場合があります。

4. *Scan ボタンをクリックして、選択したプロファイルでシステムをスキャンします。*



5. *スキャン結果を XCCDF ファイル、ARF ファイル、または HTML ファイルの形式で保存するには、Save Results コンボボックスをクリックします。人間が読める形式のスキャンレポートを作成するには、HTML Report オプションを選択します。XCCDF 形式および ARF (データストリーム) 形式は、追加の自動処理に適しています。3 つのオプションはすべて繰り返し選択できます。*

6. *結果ベースの修復をファイルにエクスポートするには、Generate remediation role ポップアップメニューを使用します。*

8.7.2. SCAP Workbench を使用したセキュリティープロファイルのカスタマイズ

セキュリティープロファイルをカスタマイズするには、特定のルール (パスワードの最小長など) のパラメーターを変更し、別の方法で対象とするルールを削除し、追加のルールを選択して内部ポリシーを実装できます。プロファイルをカスタマイズして新しいルールの定義はできません。

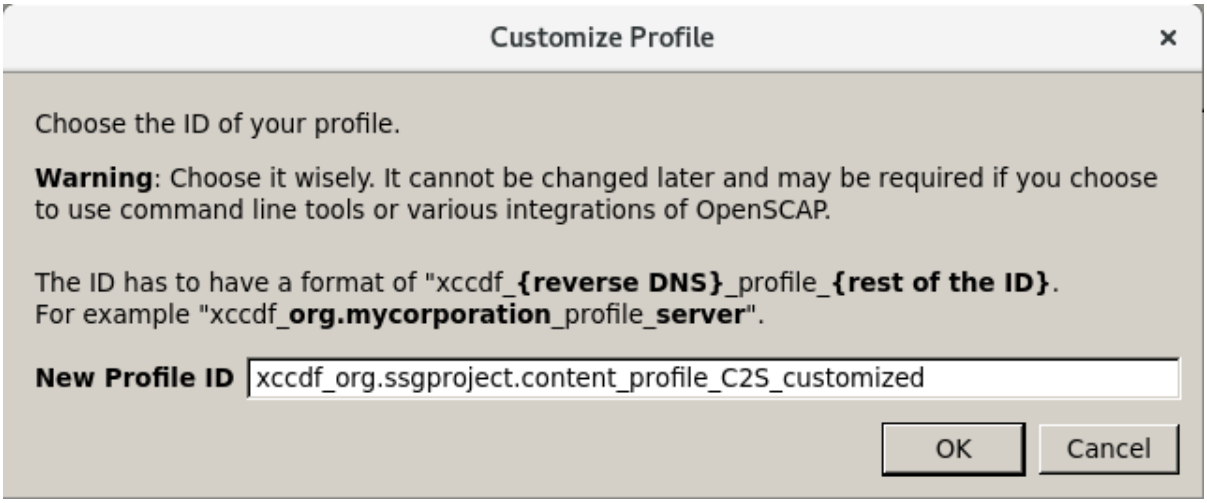
以下の手順では、SCAP Workbench を使用してプロファイルをカスタマイズ（調整）します。oscap コマンドラインユーティリティーで使用するように調整されたプロファイルを保存することもできます。

手順

1. SCAP Workbench を実行し、Open content from SCAP Security Guide または File メニューの Open Other Content を使用してカスタマイズするプロファイルを選択します。

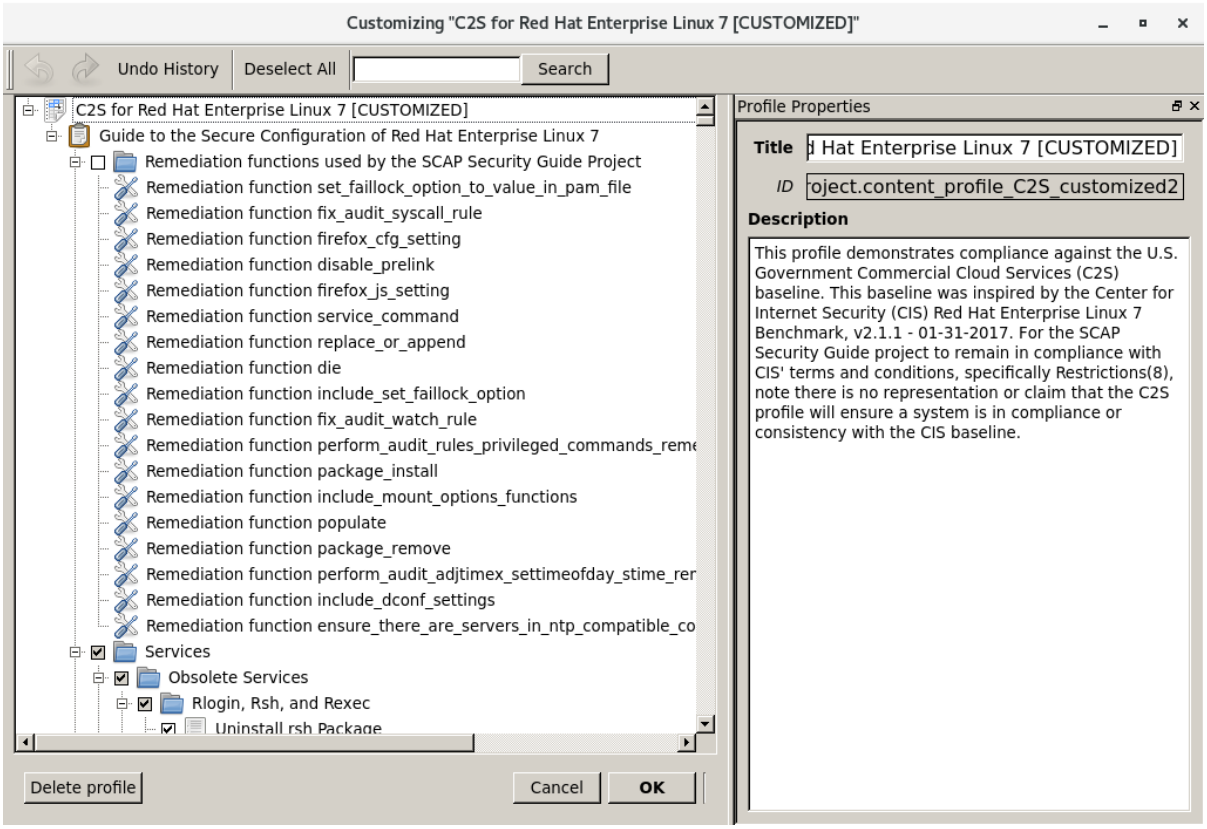
2. 選択したセキュリティープロファイルを必要に応じて調整するには、Customize ボタンをクリックします。

   これにより、オリジナルの XCCDF ファイルを変更することなく、現在選択されている XCCDF プロファイルを変更することができる新しいカスタマイズウィンドウが開きます。新しいプロファイル ID を選択します。



3. 論理グループに分類されたルールを持つツリー構造を使用するか、Search フィールドを使用して変更するルールを検索します。

4. ツリー構造のチェックボックスを使用した include ルールまたは exclude ルール、または必要に応じてルールの値を変更します。



5. *OK ボタンをクリックして変更を確認します。*

6. *変更内容を永続的に保存するには、以下のいずれかのオプションを使用します。*

- *File メニューの Save Customization Only を使用して、カスタマイズファイルを別々に保存します。*

- *File メニューの Save All を使用して、すべてのセキュリティーコンテンツを一度に保存します。*

Into a directory オプションを選択すると、SCAP Workbench は XCCDF またはデータストリームファイルとカスタマイズファイルの両方を指定された場所に保存します。これはバックアップソリューションとして使用できます。

As RPM オプションを選択すると、SCAP Workbench に、データストリームファイルおよびカスタマイズファイルを含む RPM パッケージを作成できます。これは、リモートでスキャンできないシステムにセキュリティーコンテンツを配布したり、詳細な処理のためにコンテンツを配信するのに便利です。

注記

SCAP Workbench は、カスタマイズしたプロファイルに対して結果ベースの修正をサポートしていないため、oscap コマンドラインユーティリティーでエクスポートした修復を使用します。

8.7.3. 関連情報

- scap-workbench(8) man page
- SCAP Workbench User Manual
- Deploy customized SCAP policies with Satellite 6.x - 調整スクリプトに関するナレッジベースの記事

8.8. インストール直後にセキュリティープロファイルに準拠するシステムのデプロイメント

OpenSCAP スイートを使用して、インストールプロセスの直後に、OSPP や PCI-DSS などのセキュリティープロファイルに準拠する RHEL システムをデプロイできます。このデプロイメント方法を使用すると、修正スクリプトを使用して後で適用できない特定のルール (パスワードの強度とパーティション化のルールなど) を適用できます。

8.8.1. グラフィカルインストールを使用したベースライン準拠の RHEL システムのデプロイメント

この手順を使用して、特定のベースラインに合わせた RHEL システムをデプロイします。この例では、OSPP (Protection Profile for General Purpose Operating System) を使用します。

前提条件

- グラフィカル インストールプログラムで起動している。OSCAP Anaconda Add-on はテキストのみのインストールをサポートしていないことに注意してください。
- インストール概要 画面にアクセスしている。

手順

1. インストール概要 画面 から、ソフトウェアの選択 を クリックします。ソフトウェアの選

択 画面 が開きます。

2. *ベース環境 ペインから、サーバー 環境を選択 し ます。ベース環境は、1 つだけ選択できます。*

3. *完了 をクリックし て設定を適用し、インストール概要 画面 に戻ります。*

4. ***Security Policy** をクリックします。セキュリティーポリシー 画面が 開きます。*

5. *システムでセキュリティーポリシーを有効にするには、セキュリティーポリシーの適用 を **ON** に切り替えます。*

6. *プロファイルペインから **Protection Profile for General Purpose Operating Systems** を選択します。*

7. *プロファイルの選択 をクリックして選択を確定します。*

8. *ウィンドウの下部に表示される変更 または完了 ペインの変更を確認します。残りの手動変更を完了します。*

9. ***OSPP** には厳密なパーティション要件があるため、**/boot**、**/home**、**/var**、**/var/log**、**/var/tmp**、および**/var/ log /audit** 用に個別のパーティションを作成します。*

10. *グラフィカルインストールプロセスを完了します。*

注記

*グラフィカルインストールプログラムは、インストールに成功すると、対応するキックスタートファイルを自動的に作成します。**/root/anaconda-ks.cfg** ファイルを使用して、**OSPP** 準拠のシステムを自動的にインストールできます。*

検証

1. *インストール完了後にシステムの現在のステータスを確認するには、システムを再起動し*

て新しいスキャンを開始します。

```
~]# oscap xccdf eval --profile ospp --report eval_postinstall_report.html
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

関連情報

- パーティション設定の詳細は、グラフィカルユーザーインターフェイスを使用した RHEL のインストールを参照してください。

8.8.2. キックスタートを使用したベースライン準拠の RHEL システムのデプロイメント

この手順を使用して、特定のベースラインに合わせた RHEL システムをデプロイします。この例では、OSPP (Protection Profile for General Purpose Operating System) を使用します。

前提条件

- システムに、scap-security-guide パッケージがインストールされている。

手順

1. キックスタートファイル /usr/share/scap-security-guide/kickstart/ssg-rhel7-ospp-ks.cfg を、選択したエディターで開きます。

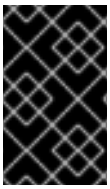
2. 設定要件を満たすように、パーティション設定スキームを更新します。OSPP に準拠するには、/boot、/home、/var、/var /log、/var/tmp、および/var/ log /audit 用の個別のパーティションを保持する必要があります。ただし、これらのパーティションのサイズは変更できます。

警告

OSCAP Anaconda Add-on はテキストのみのインストールをサポートしていないため、キックスタートファイルの text オプションを使用しないでください。詳細は RHBZ#1674001 を参照してください。

3.

キックスタートインストールを開始する方法は、キックスタートインストールの開始を参照してください。



重要

OSPP 要件では、ハッシュ形式のパスワードは確認できません。

検証

1. インストール完了後にシステムの現在のステータスを確認するには、システムを再起動して新しいスキャンを開始します。

```
~]# oscap xccdf eval --profile ospp --report eval_postinstall_report.html
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

関連情報

- 詳細は、OSCAP Anaconda Add-on プロジェクトページを参照してください。

8.9. コンテナーおよびコンテナーイメージの脆弱性スキャン

これらの手順を使用して、コンテナーまたはコンテナーイメージのセキュリティーの脆弱性を検索します。

oscap-docker コマンドラインユーティリティーまたは atomic scan コマンドラインユーティリティーのいずれかを使用して、コンテナーまたはコンテナーイメージのセキュリティー脆弱性を見つけることができます。

oscap-docker を使用すると、oscap プログラムを使用してコンテナーイメージとコンテナーをスキャンできます。

atomic scan を使用すると、OpenSCAP スキャン機能を使用して、システム上のコンテナーイメージとコンテナーをスキャンできます。既知の CVE の脆弱性と設定コンプライアンスをスキャンできます。さらに、コンテナーイメージを指定されたポリシーに修正できます。

8.9.1. oscap-dockerを使用したコンテナーイメージとコンテナーの脆弱性のスキャン

oscap-docker ユーティリティーを使用して、コンテナーおよびコンテナーイメージをスキャンでき

ます。

注記

oscap-docker コマンドには root 権限が必要で、コンテナーの ID は 2 つ目の引数になります。

前提条件

- *openscap-containers パッケージがインストールされます。*

手順

1. *コンテナーまたはコンテナーイメージの ID を取得します。以下に例を示します。*

```
~]# docker images
REPOSITORY                          TAG      IMAGE ID       CREATED       SIZE
registry.access.redhat.com/ubi7/ubi   latest   096cae65a207   7 weeks ago   239 MB
```

2. *コンテナーまたはコンテナーイメージで脆弱性をスキャンし、結果を* **vulnerability.html** *ファイルに保存します。*

```
~]# oscap-docker image-cve 096cae65a207 --report vulnerability.html
```

重要

コンテナーをスキャンするには、引数 image-cve を container-cve に置き換えてください。

検証

1. *任意のブラウザーで結果を調べます。以下に例を示します。*

```
~]$ firefox vulnerability.html &
```

関連情報

- *詳細は、oscap-docker (8)および oscap (8) の man ページを参照してください。*

8.9.2. atomic scanを使用したコンテナーイメージとコンテナーの脆弱性のスキャン

atomic scan ユーティリティーを使用すると、Red Hat がリリースする CVE OVAL 定義で定義されている既知のセキュリティー脆弱性について コンテナーおよびコンテナーイメージをスキャンできます。atomic scan コマンドの形式は次のとおりです。

```
~]# atomic scan [OPTIONS] [ID]
```

ここで、ID は、スキャンするコンテナーイメージまたはコンテナーの ID になります。

警告

アトミックスキャン 機能は非推奨となり、OpenSCAP コンテナーイメージは新しい脆弱性に対して更新されなくなりました。したがって、脆弱性スキャンの目的で oscap-docker ユーティリティーを使用することが推奨されます。

ユースケース

- すべてのコンテナーイメージをスキャンするには、--images ディレクティブを使用します。
- すべてのコンテナーをスキャンするには、--containers ディレクティブを使用します。
- 両方のタイプをスキャンするには、--all ディレクティブを使用します。
- 利用可能なすべてのコマンドラインオプションを一覧表示するには、atomic scan --help コマンドを使用します。

atomic scan コマンドのデフォルトのスキャンタイプは CVE scan です。Red Hat がリリースする CVE OVAL 定義 で定義されている既知のセキュリティー脆弱性のターゲットをチェックするために使用します。

前提条件

- ***atomic install rhel7/openscap* コマンドを使用して、*Red Hat Container Catalog (RHCC)* から *OpenSCAP* コンテナーイメージをダウンロードしてインストール している。**

手順

1. **最新の *OpenSCAP* コンテナーイメージがあることを確認し、定義が最新であることを確認します。**

```
~]# atomic help registry.access.redhat.com/rhel7/openscap | grep version
```

2. **いくつかの既知のセキュリティー脆弱性がある *RHEL 7.2* コンテナーイメージをスキャンします。**

```
~]# atomic scan registry.access.redhat.com/rhel7:7.2
docker run -t --rm -v /etc/localtime:/etc/localtime -v /run/atomic/2017-11-01-14-49-36-
614281:/scanin -v /var/lib/atomic/openscap/2017-11-01-14-49-36-614281:/scanout:rw,Z -v
/etc/oscapd:/etc/oscapd:ro registry.access.redhat.com/rhel7/openscap oscapd-evaluate scan
--no-standard-compliance --targets chroots-in-dir:///scanin --output /scanout

registry.access.redhat.com/rhel7:7.2 (98a88a8b722a718)

The following issues were found:

 RHSA-2017:2832: nss security update (Important)
 Severity: Important
  RHSA URL: https://access.redhat.com/errata/RHSA-2017:2832
  RHSA ID: RHSA-2017:2832-01
  Associated CVEs:
    CVE ID: CVE-2017-7805
    CVE URL: https://access.redhat.com/security/cve/CVE-2017-7805
...
```

関連情報

- ***Red Hat Enterprise Linux Atomic Host* の製品ドキュメント には、*atomic* コマンドの使用方法およびコンテナーの詳細な説明が記載されています。**

- ***Red Hat* カスタマーポータルでは、*Atomic* コマンドラインインターフェイス *(CLI)* のガイド を提供しています。**

8.10. 特定のベースラインを使用したコンテナーまたはコンテナーイメージの設定コンプライアンスの評価

以下の手順に従い、**Operating System Protection Profile (OSPP)** や **Payment Card Industry Data Security Standard (PCI-DSS)** などの特定のセキュリティーベースラインのあるコンテナーまたはコンテナーイメージのコンプライアンスを評価します。

前提条件

- **openscap-utils** パッケージおよび **scap-security-guide** パッケージがインストールされている。

手順

1. コンテナーまたはコンテナーイメージの **ID** を取得します。以下に例を示します。

```
~]# docker images
REPOSITORY                         TAG      IMAGE ID       CREATED       SIZE
registry.access.redhat.com/ubi7/ubi   latest   096cae65a207   7 weeks ago   239 MB
```

2. **OSPP** プロファイルでコンテナーイメージのコンプライアンスを評価し、スキャン結果を **report.html** *HTML* ファイルに保存します。

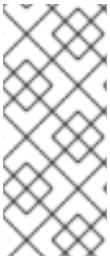
```
~]$ sudo oscap-docker 096cae65a207 xccdf eval --report report.html --profile ospp
/usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

PCI-DSS ベースラインで設定コンプライアンスを評価する場合は、**096cae65a207** をコンテナーイメージの **ID** に、**ospp** の値を **pci-dss** に置き換えてください。

検証

1. 結果をブラウザーで確認します。以下に例を示します。

```
~]$ firefox report.html &
```



注記

notapplicable が付いているルールは、コンテナー化されたシステムには適用されないルールです。これらのルールは、ベアメタルシステムまたは仮想化システムにのみ適用されます。

関連情報

- 詳細は、oscap-docker (8)および scap-security-guide (8) の man ページを参照してください。
- file:///usr/share/doc/scap-security-guide-doc-0.1.46/ ディレクトリーにインストールされている SCAP セキュリティーガイド のドキュメント。

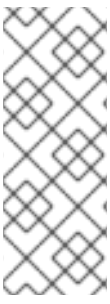
8.11. ATOMIC SCANを使用したコンテナーイメージとコンテナーの設定コンプライアンスのスキャンと修正

8.11.1. atomic scanを使用したコンテナーイメージとコンテナーの設定コンプライアンスのスキャン

このタイプのスキャンを使用して、OpenSCAP コンテナーイメージ内にバンドルされた SCAP Security Guide (SSG)によって提供される SCAP コンテンツで Red Hat Enterprise Linux ベースのコンテナーイメージとコンテナーを評価します。これにより、SCAP セキュリティーガイドで提供されるあらゆるプロファイルに対するスキャンが可能になります。

警告

アトミックスキャン 機能は非推奨となり、OpenSCAP コンテナーイメージは新しいセキュリティーコンプライアンスコンテンツで更新されなくなりました。したがって、セキュリティーコンプライアンススキャンの目的で oscap-docker ユーティリティーを使用することが推奨されます。



注記

atomic コマンドとコンテナーの使用方法の詳細については、Product Documentation for Red Hat Enterprise Linux Atomic Host 7 を参照してください。Red Hat カスタマーポータルでは、 atomic コマンドラインインターフェイス(CLI)のガイド も提供しています。

前提条件

- *atomic install rhel7/openscap コマンドを使用して、Red Hat Container Catalog (RHCC) から OpenSCAP コンテナーイメージをダウンロードしてインストール している。*

手順

1. *configuration_compliance スキャン用に OpenSCAP イメージによって提供される SCAP コンテンツを一覧表示します。*

```
~]# atomic help registry.access.redhat.com/rhel7/openscap
```

最新の Red Hat Enterprise Linux 7 コンテナーイメージの米国国防情報システム局のセキュリティー技術実装ガイド (DISA STIG) ポリシーへの準拠を検証し、スキャンから HTML レポートを生成します。

```
~]# atomic scan --scan_type configuration_compliance --scanner_args xccdf-
id=scap_org.open-scap_cref_ssg-rhel7-xccdf-
1.2.xml,profile=xccdf_org.ssgproject.content_profile_stig-rhel7-disa,report
registry.access.redhat.com/rhel7:latest
```

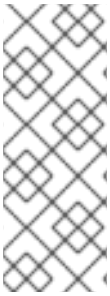
上記のコマンドの出力の最後には、スキャンに関連するファイルの情報が含まれています。

```
.............

Files associated with this scan are in /var/lib/atomic/openscap/2017-11-03-13-35-34-296606.

~]# tree /var/lib/atomic/openscap/2017-11-03-13-35-34-296606
/var/lib/atomic/openscap/2017-11-03-13-35-34-296606
├── db7a70a0414e589d7c8c162712b329d4fc670fa47ddde721250fb9fcdbed9cc2
│   ├── arf.xml
│   ├── fix.sh
│   ├── json
│   └── report.html
└── environment.json

1 directory, 5 files
```

atomic scan は、/var/lib/atomic/openscap/ ディレクトリーのスキャンからすべての結果とレポートを含むサブディレクトリーを生成します。設定準拠のため、スキャンごとに結果を

含む arf.xml ファイルが生成されます。人間が判読可能な HTML 形式のレポートファイルを生成するには、report サブオプシ ョンを --scanner_args オプションに追加してください。

2. *オプション： DISA STIG Viewer が読み取り可能な XCCDF 結果を生成するには、stig-viewer サブオプションを --scanner_args オプションに追加します。この結果は、stig.xml に配置されます。*



注記

--scanner_args オプションの xccdf-id サブオプションが省略された場合、スキャナーは、選択されたデータストリームファイルの最初の XCCDF コンポーネントでプロファイルを検索します。データストリームファイルの詳細については、「RHEL 7 の設定コンプライアンス」 を参照してください。

8.11.2. atomic scan を使用したコンテナーイメージとコンテナーの設定コンプライアンスの修正

オリジナルのコンテナーイメージに対して設定コンプライアンススキャンを実行し、DISA STIG ポリシーへの準拠を確認することができます。スキャン結果に基づき、失敗したスキャン結果に対して、bash による修復を含む修正スクリプトが生成されます。その後、修正スクリプトを元のコンテナーイメージに適用します。これを修復と呼びます。修復の結果、設定が変更されたコンテナーイメージが作成され、元のコンテナーイメージの上に新しいレイヤーとして追加されます。

重要

元のコンテナーイメージは変更されず、新しいレイヤーがその上に作成されるだけである点に注意してください。修復プロセスでは、すべての設定の改善を含む新しいコンテナーイメージが構築されます。このレイヤーの内容は、スキャニングのセキュリティーポリシー (前の例では DISA STIG ポリシー) によって定義されます。これは、修復されたコンテナーイメージが Red Hat によって署名されなくなったことも意味します。これは想定内であり、修復されたレイヤーが含まれているという点で、元のコンテナーイメージとは異なることが原因となっています。

警告

アトミックスキャン 機能は非推奨となり、OpenSCAP コンテナーイメージは新しいセキュリティーコンプライアンスコンテンツで更新されなくなりました。したがって、セキュリティーコンプライアンススキャンの目的で oscap-docker ユーティリティーを使用することが推奨されます。

前提条件

- ***atomic install rhel7/openscap** コマンドを使用して、**Red Hat Container Catalog (RHCC)** から **OpenSCAP** コンテナーイメージをダウンロードしてインストール している。*

手順

1. ***configuration_compliance** スキャン用に **OpenSCAP** イメージによって提供される **SCAP** コンテンツを一覧表示します。*

```
~]# atomic help registry.access.redhat.com/rhel7/openscap
```

2. ***指定したポリシーにコンテナーイメージを修復するには、設定コンプライアンスをスキャンするときに **--remediate** オプションを **atomic scan** コマンドに追加します。次のコマンドは、**Red Hat Enterprise Linux 7** コンテナーイメージから **DISA STIG** ポリシーに準拠した新しい修復済みコンテナーイメージをビルドします。*

```
~]# atomic scan --remediate --scan_type configuration_compliance --scanner_args
profile=xccdf_org.ssgproject.content_profile_stig-rhel7-disa,report
registry.access.redhat.com/rhel7:latest

registry.access.redhat.com/rhel7:latest (db7a70a0414e589)

The following issues were found:
.............
  Configure Time Service Maxpoll Interval
  Severity: Low
   XCCDF result: fail

  Configure LDAP Client to Use TLS For All Transactions
  Severity: Moderate
   XCCDF result: fail
.............
Remediating rule 43/44: 'xccdf_org.ssgproject.content_rule_chronyd_or_ntpd_set_maxpoll'
Remediating rule 44/44: 'xccdf_org.ssgproject.content_rule_ldap_client_start_tls'

Successfully built 9bbc7083760e
Successfully built remediated image 9bbc7083760e from
db7a70a0414e589d7c8c162712b329d4fc670fa47ddde721250fb9fcdbed9cc2.

Files associated with this scan are in /var/lib/atomic/openscap/2017-11-06-13-01-42-785000.
```

3. ***オプション： **atomic scan** コマンドの出力は、修復されたイメージ **ID** を報告します。イメージを覚えやすくするために、以下のような名前を付けてください。*

```
~]# docker tag 9bbc7083760e rhel7_disa_stig
```

8.12. RHEL 7 で対応する SCAP セキュリティーガイドプロファイル

RHEL の特定のマイナーリリースで提供される SCAP コンテンツのみを使用します。これは、強化に参加するコンポーネントが新機能で定期的に更新されるためです。SCAP コンテンツは、この更新を反映するように変更されますが、常に後方互換性があるわけではありません。

以下の表は、RHEL の各マイナーバージョンで提供されるプロファイルと、プロファイルが適合するポリシーのバージョンを検出できます。

表8.2 RHEL 7.9 で対応する SCAP セキュリティーガイドプロファイル

プロファイル名	プロファイル ID	ポリシーバージョン
CIS Red Hat Enterprise Linux 7 Benchmark for Level 2 - Server	**xccdf_org.ssgproject.content_profile_cis**	RHEL 7.9.9 以前：2.2.0 RHEL 7.9.10 以降：3.1.1
CIS Red Hat Enterprise Linux 7 Benchmark for Level 1 - Server	**xccdf_org.ssgproject.content_profile_cis_server_l1**	RHEL 7.9.10 以降：3.1.1
CIS Red Hat Enterprise Linux 7 Benchmark for Level 1 - Workstation	**xccdf_org.ssgproject.content_profile_cis_workstation_l1**	RHEL 7.9.10 以降：3.1.1
CISRed Hat Enterprise Linux 7 ベンチマーク for Level 2 - Workstation	**xccdf_org.ssgproject.content_profile_cis_workstation_l2**	RHEL 7.9.10 以降：3.1.1
Security of Information Systems (ANSSI) BP-028 Enhanced Level	**xccdf_org.ssgproject.content_profile_anssi_nt28_enhanced**	RHEL 7.9.4 以前：ドラフト RHEL 7.9.5 から RHEL 7.9.24:1.2 RHEL 7.9.25 以降：2.0
French National Agency for the Security of Information Systems (ANSSI) BP-028 High Level	**xccdf_org.ssgproject.content_profile_anssi_nt28_high**	RHEL 7.9.6 以前：ドラフト RHEL 7.9.7 から RHEL 7.9.24:1.2 RHEL 7.9.25 以降：2.0
French National Agency for the Security of Information Systems (ANSSI) BP-028 Intermediary Level	**xccdf_org.ssgproject.content_profile_anssi_nt28_intermediary**	RHEL 7.9.4 以前：ドラフト RHEL 7.9.5 から RHEL 7.9.24:1.2 RHEL 7.9.25 以降：2.0
French National Agency for the Security of Information Systems (ANSSI) BP-028 Minimal Level	**xccdf_org.ssgproject.content_profile_anssi_nt28_minimal**	RHEL 7.9.4 以前：ドラフト RHEL 7.9.5 から RHEL 7.9.24:1.2 RHEL 7.9.25 以降：2.0

プロファイル名	プロファイル ID	ポリシーバージョン
C2S for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_C2S**	バージョン付けなし
Criminal Justice Information Services (CJIS) Security Policy	**xccdf_org.ssgproject.content_profile_cjis**	5.4
Unclassified Information in Non-federal Information Systems and Organizations (NIST 800-171)	**xccdf_org.ssgproject.content_profile_cui**	r1
Australian Cyber Security Centre (ACSC) Essential Eight	**xccdf_org.ssgproject.content_profile_e8**	バージョン付けなし
Health Insurance Portability and Accountability Act (HIPAA)	**xccdf_org.ssgproject.content_profile_hipaa**	バージョン付けなし
NIST National Checklist Program Security Guide	**xccdf_org.ssgproject.content_profile_ncp**	バージョン付けなし
OSPP: 汎用オペレーティングシステム v4.2.1 の保護プロファイル	**xccdf_org.ssgproject.content_profile_ospp**	4.2.1
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss_centric**	RHEL 7.9.12 以前：3.2.1 7.9.13 以降のバージョンで削除されました。詳細は、RHBZ#2038165 を参照してください。
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss**	3.2.1
[ドラフト] DISA STIG for Red Hat Enterprise Linux Virtualization Host (RHELH)	**xccdf_org.ssgproject.content_profile_rhelh-stig**	ドラフト
VPP - Protection Profile for Virtualization v.1.0 for Red Hat Enterprise Linux Hypervisor (RHELH)	**xccdf_org.ssgproject.content_profile_rhelh-vpp**	1.0
Red Hat Corporate Profile for Certified Cloud Providers (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	バージョン付けなし
Standard System Security Profile for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_standard**	バージョン付けなし

プロファイル名	プロファイル ID	ポリシーバージョン
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig**	RHEL 7.9.0 および 7.9.1: 1.4 RHEL 7.9.2 から 7.9.4: V3R1 RHEL 7.9.5 および 7.9.6:V3R2 RHEL 7.9.7 から RHEL 7.9.9:V3R3 RHEL7.9.10 および RHEL7.9.11:V3R5 RHEL 7.9.12 および RHEL 7.9.13:V3R6 RHEL 7.9.14 to RHEL 7.9.16:V3R7 RHEL 7.9.17 to RHEL 7.9.20:V3R8 RHEL 7.9.21 to RHEL 7.9.24:V3R10 RHEL 7.9.25 以降：V3R12
DISA STIG with GUI for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig_gui**	RHEL 7.9.7 から RHEL 7.9.9:V3R3 RHEL7.9.10 および RHEL7.9.11:V3R5 RHEL 7.9.12 および RHEL 7.9.13:V3R6 RHEL 7.9.14 to RHEL 7.9.16:V3R7 RHEL 7.9.17 to RHEL 7.9.20:V3R8 RHEL 7.9.21 to RHEL 7.9.24:V3R10 RHEL 7.9.25 以降：V3R12

表8.3 RHEL 7.8 で対応する SCAP セキュリティーガイドプロファイル

プロファイル名	プロファイル ID	ポリシーバージョン
ドラフト - ANSSI DAT-NT28 (拡張)	**xccdf_org.ssgproject.content_profile_anssi_nt28_enhanced**	ドラフト
ドラフト - ANSSI DAT-NT28 (高)	**xccdf_org.ssgproject.content_profile_anssi_nt28_high**	ドラフト
ドラフト - ANSSI DAT-NT28 (中間)	**xccdf_org.ssgproject.content_profile_anssi_nt28_intermediary**	ドラフト

プロファイル名	プロファイル ID	ポリシーバージョン
ドラフト - ANSSI DAT-NT28 (最小)	**xccdf_org.ssgproject.content_profile_anssi_nt28_minimal**	ドラフト
C2S for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_C2S**	バージョン付けなし
Criminal Justice Information Services (CJIS) Security Policy	**xccdf_org.ssgproject.content_profile_cjis**	5.4
Unclassified Information in Non-federal Information Systems and Organizations (NIST 800-171)	**xccdf_org.ssgproject.content_profile_cui**	r1
Australian Cyber Security Centre (ACSC) Essential Eight	**xccdf_org.ssgproject.content_profile_e8**	バージョン付けなし
Health Insurance Portability and Accountability Act (HIPAA)	**xccdf_org.ssgproject.content_profile_hipaa**	バージョン付けなし
NIST National Checklist Program Security Guide	**xccdf_org.ssgproject.content_profile_ncp**	バージョン付けなし
OSPP: 汎用オペレーティングシステム v4.2.1 の保護プロファイル	**xccdf_org.ssgproject.content_profile_ospp**	4.2.1
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss_centric**	3.2.1
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss**	3.2.1
[ドラフト] DISA STIG for Red Hat Enterprise Linux Virtualization Host (RHELH)	**xccdf_org.ssgproject.content_profile_rhelh-stig**	ドラフト
VPP - Protection Profile for Virtualization v.1.0 for Red Hat Enterprise Linux Hypervisor (RHELH)	**xccdf_org.ssgproject.content_profile_rhelh-vpp**	1.0
Red Hat Corporate Profile for Certified Cloud Providers (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	バージョン付けなし
Standard System Security Profile for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_standard**	バージョン付けなし

プロファイル名	プロファイル ID	ポリシーバージョン
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig**	1.4

表8.4 RHEL 7.7 で対応する SCAP セキュリティーガイドプロファイル

プロファイル名	プロファイル ID	ポリシーバージョン
C2S for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_C2S**	バージョン付けなし
Criminal Justice Information Services (CJIS) Security Policy	**xccdf_org.ssgproject.content_profile_cjis**	5.4
Health Insurance Portability and Accountability Act (HIPAA)	**xccdf_org.ssgproject.content_profile_hipaa**	バージョン付けなし
Unclassified Information in Non-federal Information Systems and Organizations (NIST 800-171)	**xccdf_org.ssgproject.content_profile_nist-800-171-cui**	r1
OSPP: 汎用オペレーティングシステム v. の保護プロファイル4.2	**xccdf_org.ssgproject.content_profile_ospp42**	4.2
米国政府共通設定基準	**xccdf_org.ssgproject.content_profile_ospp**	3.9
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss_centric**	3.2.1
PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss**	3.2.1
VPP - Protection Profile for Virtualization v.1.0 for Red Hat Enterprise Linux Hypervisor (RHELH)	**xccdf_org.ssgproject.content_profile_rhelh-vpp**	1.0
Red Hat Corporate Profile for Certified Cloud Providers (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	バージョン付けなし
Standard System Security Profile for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_standard**	バージョン付けなし
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig-rhel7-disa**	1.4

表8.5 RHEL 7.6 で対応する SCAP セキュリティーガイドプロファイル

プロファイル名	プロファイル ID	ポリシーバージョン
C2S for Red Hat Enterprise Linux 7	xccdf_org.ssgproject.content_profile_**C2S**	バージョン付けなし
Criminal Justice Information Services (CJIS) Security Policy	**xccdf_org.ssgproject.content_profile_cjis**	5.4
Health Insurance Portability and Accountability Act (HIPAA)	**xccdf_org.ssgproject.content_profile_hipaa**	バージョン付けなし
Unclassified Information in Non-federal Information Systems and Organizations (NIST 800-171)	**xccdf_org.ssgproject.content_profile_nist-800-171-cui**	r1
OSPP: 汎用オペレーティングシステム v. の保護プロファイル4.2	**xccdf_org.ssgproject.content_profile_ospp42**	4.2
米国政府共通設定基準	**xccdf_org.ssgproject.content_profile_ospp**	3.9
PCI-DSS v3 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss_centric**	3.1
PCI-DSS v3 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss**	3.1
Red Hat Corporate Profile for Certified Cloud Providers (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	バージョン付けなし
Standard System Security Profile for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_standard**	バージョン付けなし
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig-rhel7-disa**	1.4

表8.6 RHEL 7.5 で対応する SCAP セキュリティーガイドプロファイル

プロファイル名	プロファイル ID	ポリシーバージョン
C2S for Red Hat Enterprise Linux	**xccdf_org.ssgproject.content_profile_C2S**	バージョン付けなし
Criminal Justice Information Services (CJIS) Security Policy	**xccdf_org.ssgproject.content_profile_cjis-rhel7-server**	5.4

プロファイル名	プロファイル ID	ポリシーバージョン
汎用システムの共通プロファイル	**xccdf_org.ssgproject.content_profile_**common	バージョン付けなし
標準的な Docker ホストのセキュリティープロファイル	**xccdf_org.ssgproject.content_profile_**docker-host	バージョン付けなし
Unclassified Information in Non-federal Information Systems and Organizations (NIST 800-171)	**xccdf_org.ssgproject.content_profile_**nist-800-171-cui	r1
米国政府共通設定基準 (USGCB / STIG) - ドラフト	**xccdf_org.ssgproject.content_profile_**ospp-rhel7	3.9
PCI-DSS v3 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_**pci-dss_centric	3.1
PCI-DSS v3 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_**pci-dss	3.1
Red Hat Corporate Profile for Certified Cloud Providers (RH CCP)	**xccdf_org.ssgproject.content_profile_**rht-ccp	バージョン付けなし
標準システムセキュリティープロファイル	**xccdf_org.ssgproject.content_profile_**standard	バージョン付けなし
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_**stig-rhel7-disa	1.4
STIG for Red Hat Virtualization Hypervisor	**xccdf_org.ssgproject.content_profile_**stig-rhevh-upstream	1.4

表8.7 RHEL 7.4 で対応する SCAP セキュリティーガイドプロファイル

プロファイル名	プロファイル ID	ポリシーバージョン
C2S for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_**C2S	バージョン付けなし
Criminal Justice Information Services (CJIS) Security Policy	**xccdf_org.ssgproject.content_profile_**cjis-rhel7-server	5.4
汎用システムの共通プロファイル	**xccdf_org.ssgproject.content_profile_**common	バージョン付けなし
標準的な Docker ホストのセキュリティープロファイル	**xccdf_org.ssgproject.content_profile_**docker-host	バージョン付けなし

プロファイル名	プロファイル ID	ポリシーバージョン
Unclassified Information in Non-federal Information Systems and Organizations (NIST 800-171)	**xccdf_org.ssgproject.content_profile_nist-800-171-cui**	r1
米国政府共通設定基準 (USGCB / STIG) - ドラフト	**xccdf_org.ssgproject.content_profile_ospp-rhel7**	3.9
PCI-DSS v3 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss_centric**	3.1
PCI-DSS v3 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_pci-dss**	3.1
Red Hat Corporate Profile for Certified Cloud Providers (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	バージョン付けなし
標準システムセキュリティープロファイル	**xccdf_org.ssgproject.content_profile_standard**	バージョン付けなし
DISA STIG for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_stig-rhel7-disa**	1.4
STIG for Red Hat Virtualization Hypervisor	**xccdf_org.ssgproject.content_profile_stig-rhevh-upstream**	

表8.8 RHEL 7.3 で対応する SCAP セキュリティーガイドプロファイル

プロファイル名	プロファイル ID	ポリシーバージョン
C2S for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_C2S**	バージョン付けなし
Criminal Justice Information Services (CJIS) Security Policy	**xccdf_org.ssgproject.content_profile_cjis-rhel7-server**	5.4
汎用システムの共通プロファイル	**xccdf_org.ssgproject.content_profile_common**	バージョン付けなし
CNSSI 1253 Low/Low/Low Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_nist-cl-il-al**	バージョン付けなし
米国政府共通設定基準 (USGCB / STIG)	**xccdf_org.ssgproject.content_profile_ospp-rhel7-server**	バージョン付けなし

プロファイル名	プロファイル ID	ポリシーバージョン
PCI-DSS v3 Control Baseline for Red Hat Enterprise Linux 7	**xccdf_org.ssgproject.content_profile_**pci-dss	3.1
Red Hat Corporate Profile for Certified Cloud Providers (RH CCP)	**xccdf_org.ssgproject.content_profile_**rht-ccp	バージョン付けなし
標準システムセキュリティープロファイル	**xccdf_org.ssgproject.content_profile_**standard	バージョン付けなし
GUI を実行している STIG for Red Hat Enterprise Linux 7 Server	**xccdf_org.ssgproject.content_profile_**stig-rhel7-server-gui-upstream	1.4
STIG for Red Hat Enterprise Linux 7 Server	**xccdf_org.ssgproject.content_profile_**stig-rhel7-server-upstream	1.4
STIG for Red Hat Enterprise Linux 7 Workstation	**xccdf_org.ssgproject.content_profile_**stig-rhel7-workstation-upstream	1.4

表8.9 RHEL 7.2 で対応する SCAP セキュリティーガイドプロファイル

プロファイル名	プロファイル ID	ポリシーバージョン
汎用システムの共通プロファイル	**xccdf_org.ssgproject.content_profile_**common	バージョン付けなし
PCI-DSS v3 Control Baseline for Red Hat Enterprise Linux 7 のドラフト	**xccdf_org.ssgproject.content_profile_**pci-dss	ドラフト
Red Hat Corporate Profile for Certified Cloud Providers (RH CCP)	**xccdf_org.ssgproject.content_profile_**rht-ccp	バージョン付けなし
標準システムセキュリティープロファイル	**xccdf_org.ssgproject.content_profile_**standard	バージョン付けなし
STIG for Red Hat Enterprise Linux 7 Server のプレリリースのドラフト	**xccdf_org.ssgproject.content_profile_**stig-rhel7-server-upstream	ドラフト

表8.10 RHEL 7.1 で対応する SCAP セキュリティーガイドプロファイル

プロファイル名	プロファイル ID	ポリシーバージョン
Red Hat Corporate Profile for Certified Cloud Providers (RH CCP)	**xccdf_org.ssgproject.content_profile_rht-ccp**	バージョン付けなし

関連情報

- *RHEL 8 のプロファイルの詳細は、RHEL 8 で対応している SCAP Security Guide プロファイルを参照してください。*

8.13. 関連情報

- *Supported versions of the SCAP Security Guide - この記事には、さまざまなバージョンの RHEL でサポートされている SCAP セキュリティーガイドのバージョンが記載されています。*
- *OpenSCAP プロジェクトページ: OpenSCAP プロジェクトのホームページでは、oscap ユーティリティーおよび SCAP に関連するその他のコンポーネントおよびプロジェクトに関する詳細情報が提供されています。*
- *SCAP Workbench プロジェクトページ - SCAP Workbench プロジェクトのホームページでは、scap-workbench アプリケーションの詳細情報が提供されています。*
- *SCAP Security Guide (SSG) プロジェクトページ - SSG プロジェクトのホームページでは、Red Hat Enterprise Linux 向けの最新セキュリティーコンテンツが提供されています。*
- *Red Hat Security Demos: Creating Customized Security Policy Content to Automate Security Compliance - Red Hat Enterprise Linux に含まれるツールを使用してセキュリティーコンプライアンスを自動化し、業界標準のセキュリティーポリシーとカスタムセキュリティーポリシーの両方に準拠するためのハンズオンラボ。トレーニングや、このラボ演習へのアクセスを希望する場合は、Red Hat アカウントチームにお問い合わせください。*
- *Red Hat Security Demos: Defend Yourself with RHEL Security Technologies - OpenSCAP を含む Red Hat Enterprise Linux で利用可能な主要なセキュリティー技術を使用して、RHEL システムの全レベルでセキュリティーを実装する方法を学ぶためのハンズオンラボ。トレーニングや、このラボ演習へのアクセスを希望する場合は、Red Hat アカウントチームにお問い合わせください。*
-

National Institute of Standards and Technology (NIST) SCAP のページ - このページでは、SCAP の出版物、仕様、SCAP 検出プログラムなどの SCAP 関連の資料が多数提供されます。

- *National Vulnerability Database (NVD) - このページは、SCAP コンテンツおよびその他の SCAP 規格ベースの脆弱性管理データに関する最大のリポジトリーです。*

- *Red Hat OVAL content repository - Red Hat Enterprise Linux システムの脆弱性に関する OVAL 定義を含むリポジトリーです。このページは、脆弱性の情報を得るために確認が推奨されるページです。*

- *MITRE CVE - これは、MITRE corporation が提供する既知のセキュリティー脆弱性のデータベースです。RHEL の場合は、Red Hat が提供する OVAL CVE コンテンツを使用することが推奨されます。*

- *MITRE OVAL - このページでは、MITRE corporation が提供する OVAL 関連のプロジェクトが紹介されています。OVAL の関連情報、たとえば OVAL 言語の最新バージョン、数千にもなる OVAL 定義が用意された OVAL コンテンツのリポジトリーがあります。RHEL のスキャンには、Red Hat が提供する OVAL CVE コンテンツを使用することが推奨されます。*

- *Red Hat Satellite ドキュメント - このガイドセットでは、OpenSCAP を使用して複数のシステムでシステムセキュリティーを維持する方法などが説明されています。*

第9章 米連邦政府の標準および規制

連邦政府および業界のセキュリティー仕様、標準、規制に準拠するよう組織が努力することで、セキュリティーレベルを維持することが可能です。本章では、これらの標準と規制の一部について説明します。

9.1. FIPS (FEDERAL INFORMATION PROCESSING STANDARD)

連邦情報処理標準 (FIPS) 140-2 は、U.S により開発されたコンピューターセキュリティー標準です。暗号化モジュールの品質を検証する政府および業界の作業グループ。NIST Computer Security Resource Center で公式の FIPS 公開を参照してください。

FIPS 140-2 標準は、暗号化ツールがアルゴリズムを適切に実装できるようにします。これらのレベルや FIPS 規格の他の仕様の詳細については、http://dx.doi.org/10.6028/NIST.FIPS.140-2 の完全な FIPS 140-2 規格を参照してください。

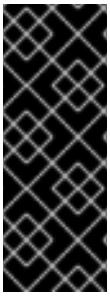
コンプライアンス要件については、Red Hat Government Standards ページを参照してください。

9.1.1. FIPS モードの有効化

Red Hat Enterprise Linux を連邦情報処理標準 (FIPS: Federal Information Processing Standard) 140-2 に準拠させるには、認定された暗号モジュールが使用されるようにいくつかの変更を行う必要があります。FIPS モードは、システムのインストール中またはインストール後に有効にできます。

システムのインストール時

FIPS 140-2 への厳密な準拠 を満たすには、システムのインストール中にカーネルコマンドラインに fips=1 カーネルオプションを追加します。このオプションを使用すると、すべてのキーの生成は FIPS 承認のアルゴリズムで行われ、継続的な監視テストが実施されます。インストール後、システムは FIPS モードで自動的に起動するように設定されます。



重要

インストール作業中は、マウスを動かしたり、キーストロークを多く押したりして、システムに十分なエントロピーがあることを確認してください。キーストロークの推奨量は 256 以上です。256 未満のキーストロークは、一意でないキーを生成する可能性があります。

システムインストールの後

インストール後にシステムのカーネル空間とユーザー空間を FIPS モードにするためには、以下の手順で行います。

1. dracut-fips パッケージをインストールします。

```
~]# yum install dracut-fips
```

AES New Instructions (AES-NI) をサポートしている CPU の場合は、dracut-fips-aesni パッケージもインストールしてください。

```
~]# yum install dracut-fips-aesni
```

2. initramfs ファイルを再生成します。

```
~]# dracut -v -f
```

モジュール内の整合性検証を有効にし、カーネル起動時に必要なすべてのモジュールが存在するようにするには、initramfs ファイルを再生成する必要があります。

警告

この操作は、既存の initramfs ファイルを上書きします。

3. ブートローダーの設定を変更します。

FIPS モードで起動するには、ブートローダーのカーネルコマンドラインに fips=1 オプションを追加してください。/boot または /boot/EFI/ パーティションが別のパーティションにある場合は、boot= <partition > （ここでの < partition > は /bootを表します）パラメーターをカーネルコマンドラインに追加します。

ブートパーティションを特定するには、次のコマンドを入力します。

```
~]$ df /boot
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/sda1               495844     53780    416464  12% /boot
```

ブート間でデバイスの名前が変更された場合でも **boot=** 設定オプションが機能するようにするには、次のコマンドを実行して、パーティションのユニバーサル一意識別子 (**UUID**) を識別します。

```
~]$ blkid /dev/sda1
/dev/sda1: UUID="05c000f1-f899-467b-a4d9-d5ca4424c797" TYPE="ext4"
```

UUID をカーネルコマンドラインに追加します。

```
boot=UUID=05c000f1-f899-467b-a4d9-d5ca4424c797
```

お使いのブートローダーによっては、以下のように変更してください。

- **GRUB 2**

  /etc/default /grub ファイルの **GRUB_CMDLINE_LINUX** キーに **fips=1** および **boot=<partition of /boot>** オプションを追加します。**/etc/default/grub** に変更を適用するには、以下のように **grub.cfg** ファイルを再構築します。

  - **BIOS** ベースのマシンでは、**root** で以下のコマンドを入力します。

    ```
    ~]# grub2-mkconfig -o /etc/grub2.cfg
    ```

  - **UEFI** ベースのマシンでは、**root** で以下のコマンドを入力します。

    ```
    ~]# grub2-mkconfig -o /etc/grub2-efi.cfg
    ```

- **zipl (IBM z Systems** アーキテクチャーのみ)

fips=1 および boot=<partition of /boot > オプションを /etc/zipl.conf のカーネルコマンドラインに追加し、次のように入力して変更を適用します。

```
~]# zipl
```

4. *プレリンクが無効になっていることを確認してください。*

   モジュール内の整合性検証を適切に動作させるには、ライブラリーとバイナリーの事前リンクを無効にする必要があります。事前リンクは、デフォルトではインストールされていない prelink パッケージによって実行されます。prelink がインストールされていない限り、この手順は必要ありません。事前リンクを無効にするには、/etc/sysconfig/prelink 設定ファイルで PRELINKING=no オプションを設定します。すべてのシステムファイルで既存の事前リンクを無効にするには、prelink -u -a コマンドを使用します。

5. *システムを再起動します。*

コンテナーでの FIPS モードの有効化

ホストも FIPS140-2 モードに設定されていて、次のいずれかの要件が満たされている場合は、コンテナーを FIPS140-2 モードに切り替えることができます。

- *dracut-fips パッケージがコンテナーにインストールされます。*
- */etc/system-fips ファイルは、ホストからコンテナーにマウントされます。*

9.2. NISPOM (NATIONAL INDUSTRIAL SECURITY PROGRAM OPERATING MANUAL)

NISPOM (DoD 5220.22-M とも呼ばれる) は、米国国家産業セキュリティープログラム (NISP) の設定要素として、すべての政府契約者に対し、機密情報に関する一連の手順と要件を定めています。現行の NISPOM は 2006 年 2 月 28 日付で、2013 年 3 月 28 日からは主要な変更が組み込まれています。NISPOM ドキュメント は、http://www.nispom.org/NISPOM-download.html の URL からダウンロードできます。

9.3. PCI DSS (PAYMENT CARD INDUSTRY DATA SECURITY STANDARD)

https://www.pcisecuritystandards.org/about/index.shtml を参照: PCI Security Standards Council は、2006 年に発足したオープンなグローバルフォーラムで、Data Security Standard (DSS) を含む PCI Security Standards の開発、管理、教育、啓発を担っています。

PCI DSS 標準は、https://www.pcisecuritystandards.org/security_standards/pci_dss.shtml からダウンロードできます。

9.4. セキュリティー技術実装ガイド

セキュリティー技術実装ガイド (STIG) は、コンピューターのソフトウェアやハードウェアの安全なインストールと保守を標準化するための方法論です。

STIG の詳細は、https://public.cyber.mil/stigs/ を参照してください。

付録A 暗号の標準

A.1. 同期式の暗号

A.1.1. 高度暗号化標準 — AES

暗号化における Advanced Encryption Standard (AES) は、米国政府が採用する暗号標準です。この標準は、AES-128、AES-192、および AES-256 の 3 つのブロック暗号で設定されており、元々は Rijndael として公開されていたより大きなコレクションから採用されています。各 AES 暗号のブロックサイズは 128 ビットで、キーサイズはそれぞれ 128 ビット、192 ビット、256 ビットです。AES 暗号は、その前身である Data Encryption Standard (DES) と同様に、多くの解析が行われ、現在世界中で利用されています。[3]

A.1.1.1. AES の歴史

AES は、5 年間の標準化プロセスを経て、2001 年 11 月 26 日に National Institute of Standards and Technology (NIST) によって米国 FIPS PUB 197 (FIPS 197) として発表されました。15 種類のデザインが提示され、評価された結果、Rijndael が最も適したものとして選択されました。2002 年 5 月 26 日に標準として有効となりました。これは、さまざまな暗号化パッケージで利用できます。AES は、NSA が最高機密情報用として承認した、一般にアクセス可能でオープンな最初の暗号です (AES に関する Wikipedia の記事のセキュリティーの項を参照ください)。[4]

Rijndael 暗号は、ベルギー人の暗号家 Joan Daemen と Vincent Rijmen の 2 人によって開発され、AES の選考プロセスに提出されました。Rijndael は、2 人の発明者の名前の混成語です。[5]

A.1.2. データ暗号化標準 — DES

DES(Data Encryption Standard) は、ブロック暗号 (共有秘密暗号) の一種で、1976 年に米国国立標準局 (National Bureau of Standards) が米国の公式な FIPS(Federal Information Processing Standard) に選定したもので、その後、国際的にも広く使用されています。56 ビットの鍵を使用する対称鍵アルゴリズムをベースにしています。このアルゴリズムは、設計要素が機密であること、鍵長が比較的短いこと、国家安全保障局 (NSA) のバックドアが疑われることなどから、当初は物議をかもしていました。その結果、DES は、ブロック暗号とその暗号解析の最新の理解を動機付ける厳しい学術的精査を受けました。[6]

A.1.2.1. DES の歴史

DES は現在、多くのアプリケーションで安全でないと考えられています。これは、鍵のサイズが 56 ビットと小さすぎることが主な原因です。1999 年 1 月には、distributed.net と Electronic Frontier Foundation が共同で、DES の鍵を 22 時間 15 分で破っています。また、実際にはマウントすることは不可能ですが、暗号の理論上の弱点を示す分析結果もいくつかあります。このアルゴリズムは、理論

的な攻撃はあるものの、Triple DES という形で実質的に安全であると考えられています。最近では、この暗号は Advanced Encryption Standard (AES) に取って代わられています。[7]

一部のドキュメントでは、標準としての DES と、DEA (Data Encryption Algorithm) と呼ばれるアルゴリズムとしての DES が区別されています。[8]

A.2. 公開鍵の暗号化

公開鍵暗号は、多くの暗号アルゴリズムや暗号システムで採用されている暗号化アプローチで、対称鍵アルゴリズムの代わりに、あるいは対称鍵アルゴリズムに加えて、非対称鍵アルゴリズムを使用することを特徴としています。公開鍵 - 秘密鍵暗号の手法を用いることで、これまで知られていなかった通信の保護やメッセージの認証の方法が数多く実用化されました。対称鍵アルゴリズムを使用する場合に必要とされる、1 つ以上の秘密鍵のセキュアな初期交換は必要ありません。これは、デジタル署名の作成にも使用できます。[9]

公開鍵暗号は、世界中で広く使われている基本的な技術であり、Transport Layer Security (TLS) (SSL の後継)、PGP、GPG などのインターネット標準の基礎となるアプローチです。[10]

公開鍵暗号化で使用される際立った手法は、非対称鍵アルゴリズムの使用です。この場合、メッセージの暗号化に使用される鍵は、メッセージの復号化に使用される鍵と同じではありません。各ユーザーには、公開鍵と秘密鍵の暗号鍵のペアがあります。秘密鍵は秘密にされ、公開鍵は広く配布されます。メッセージは受信者の公開鍵で暗号化され、対応する秘密鍵によってのみ復号化されます。鍵は数学的に関連付けられていますが、秘密鍵を公開鍵から実現可能な形で (つまり、実際のまたは予測された実践で) 導出することはできません。このようなアルゴリズムの発見により、1970 年代半ば以降、暗号技術に革命的な変化が起きました。[11]

一方、対称鍵アルゴリズムは、そのバリエーションが数千年にわたって使用されており、暗号化と復号化の両方に、送信者と受信者が共有する単一の秘密鍵 (これも非公開にする必要があるため、一般的な用語のあいまいさを考慮します) を使用します。対称暗号化スキームを使用するには、送信側と受信側が事前に鍵を安全に共有する必要があります。[12]

対称鍵アルゴリズムは、ほとんどの場合、計算量がはるかに少ないため、鍵交換アルゴリズムを使用して鍵を交換し、その鍵と対称鍵アルゴリズムを使用してデータを送信することが一般的となっています。たとえば、PGP およびスキームの SSL/TLS ファミリーはこれを行うため、ハイブリッド暗号化システムと呼ばれます。[13]

A.2.1. Diffie-Hellman

Diffie-Hellman 鍵交換 (D-H) は、互いに関する予備知識がない 2 者が、安全でない通信チャネルを

介して共有秘密鍵を共同で確立できるようにする暗号化プロトコルです。その後、この鍵を使用し、対称鍵暗号を使用する後続の通信を暗号化できます。[14]

A.2.1.1. Diffie-Hellman の歴史

このスキームは、1976 年に Whitfield Diffie と Martin Hellman によって初めて公開されました。しかし、これとは別にその数年前に、英国の信号情報機関である GCHQ 内で Malcolm J. Williamson が考案していたことが後に判明しましたが、機密扱いとなっていました。2002 年、Hellman は、公開鍵暗号の発明に貢献した Ralph Merkle の功績を称えて、このアルゴリズムを Diffie-Hellman-Merkle 鍵交換と呼ぶことを提案しました (Hellman, 2002)。[15]

Diffie-Hellman 鍵合意自体は匿名 (非認証) の鍵合意プロトコルですが、さまざまな認証済みプロトコルの基礎を提供し、Transport Layer Security の一時的モード (暗号スイートに応じて EDH または DHE と呼ばれる) では、Perect Forward Secrecy を提供するために使用されています。[16]

米国現在失効している米国特許 4,200,770 号には、アルゴリズムに関する説明のほか、Hellman、Diffie、および Merkle の名前が発明者として記載されています。[17]

A.2.2. RSA

暗号化において、RSA (最初に公発した Rivest、Shamir、Adleman の 3 氏の略) は公開鍵暗号化のアルゴリズムの 1 つです。これは、署名と暗号化に適していることが確認された最初のアルゴリズムであり、公開鍵暗号化における最初の大きな進歩の 1 つでした。RSA は、電子商取引プロトコルで広く使用されており、十分に長いキーと最新の実装を使用すれば安全であると考えられています。

A.2.3. DSA

DSA (Digital Signature Algorithm) は、デジタル署名の標準で、米国連邦政府が策定したデジタル署名の標準です。DSA は署名のみを目的としており、暗号化アルゴリズムではありません。[18]

A.2.4. SSL/TLS

Transport Layer Security (TLS) とその前進である Secure Sockets Layer (SSL) は、インターネットなどのネットワーク上の通信にセキュリティーを提供する暗号プロトコルである。TLS と SSL は、トランスポート層でネットワーク接続のセグメントをエンドツーエンドで暗号化します。

このプロトコルのいくつかのバージョンは、Web ブラウジング、電子メール、インターネットファクス、インスタントメッセージ、VoIP (Voice-over-IP) などのアプリケーションで広く使用されていま

す。[19]

A.2.5. Cramer-Shoup 暗号システム

Cramer-Shoup システムは、非対称鍵暗号化アルゴリズムであり、標準的な暗号化の仮定を使用して、適応的選択暗号文攻撃に対して安全であることが証明された最初の効率的なスキームでした。その安全性は、決定論的な Diffie-Hellman の仮定が計算不可能であること (広く仮定されているが、証明されていない) に基づいています。1998 年に Ronald Cramer と Victor Shoup によって開発されたこの暗号システムは、ElGamal 暗号システムを拡張したものです。非常に順応性が高い ElGamal とは対照的に、Cramer-Shoup は、賢い攻撃者に対しても非順応性を確保するために要素を追加しています。この非順応性は、衝突耐性のあるハッシュ関数と追加の計算を使用することで実現され、ElGamal の 2 倍の暗号文が生成されます。[20]

A.2.6. ElGamal 暗号

暗号化では、ElGamal 暗号化システムは、Diffie-Hellman キー合意に基づく公開キー暗号化用の非対称キー暗号化アルゴリズムです。これは、1985 年に Taher ElGamal によって記述されました。ElGamal 暗号は、無料の GNU Privacy Guard ソフトウェア、最近の PGP バージョン、およびその他の暗号システムで使用されています。[21]

---

[3] "Advanced Encryption Standard." Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

[4] "Advanced Encryption Standard." Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

[5] "Advanced Encryption Standard." Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

[6] "Data Encryption Standard." Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Data_Encryption_Standard

[7] "Data Encryption Standard." Wikipedia. 14 November 2009 http://en.wikipedia.org/wiki/Data_Encryption_Standard

[8] "Data Encryption Standard." Wikipedia. 14 November 2009

http://en.wikipedia.org/wiki/Data_Encryption_Standard

[9]
公開鍵の暗号化 Wikipedia.*14 November 2009* *http://en.wikipedia.org/wiki/Public-key_cryptography*

[10]
公開鍵の暗号化 Wikipedia.*14 November 2009* *http://en.wikipedia.org/wiki/Public-key_cryptography*

[11]
公開鍵の暗号化 Wikipedia.*14 November 2009* *http://en.wikipedia.org/wiki/Public-key_cryptography*

[12]
公開鍵の暗号化 Wikipedia.*14 November 2009* *http://en.wikipedia.org/wiki/Public-key_cryptography*

[13]
公開鍵の暗号化 Wikipedia.*14 November 2009* *http://en.wikipedia.org/wiki/Public-key_cryptography*

[14]
"Diffie-Hellman." Wikipedia.*2009 年 11 月 14 日* *http://en.wikipedia.org/wiki/Diffie-Hellman*

[15]
"Diffie-Hellman." Wikipedia.*2009 年 11 月 14 日* *http://en.wikipedia.org/wiki/Diffie-Hellman*

[16]
"Diffie-Hellman." Wikipedia.*2009 年 11 月 14 日* *http://en.wikipedia.org/wiki/Diffie-Hellman*

[17]
"Diffie-Hellman." Wikipedia.*2009 年 11 月 14 日* *http://en.wikipedia.org/wiki/Diffie-Hellman*

[18]
"DSA." Wikipedia.*24 February 2010* *http://en.wikipedia.org/wiki/Digital_Signature_Algorithm*

[19]
"TLS/SSL." Wikipedia.*24 February 2010* *http://en.wikipedia.org/wiki/Transport_Layer_Security*

[20]
"Cramer-Shoup 暗号システム" Wikipedia.*24 February 2010* *http://en.wikipedia.org/wiki/Cramer–Shoup_cryptosystem*

[21]
"ElGamal encryption" Wikipedia.*24 February 2010*
http://en.wikipedia.org/wiki/ElGamal_encryption

付録B 更新履歴

改訂 1-43 Fri Feb 7 2020 Jan Fiala
コンプライアンスと脆弱性スキャンの章を更新した非同期リリースです。

改訂 1-42 Fri Aug 9 2019 Mirek Jahoda
7.7 GA 公開用バージョン

改訂 1-41 Sat Oct 20 2018 Mirek Jahoda
7.6 GA リリースのバージョン

改訂 1-32 Wed Apr 4 2018 Mirek Jahoda
7.5 GA 公開用バージョン

改訂 1-30 Thu Jul 27 2017 Mirek Jahoda
7.4 GA 公開用バージョン

改訂 1-24 Mon Feb 6 2017 Mirek Jahoda
misc. 更新を含む非同期リリースで、特に firewalld セクションの更新を行いました。

改訂 1-23 Tue Nov 1 2016 Mirek Jahoda
7.3 GA リリースのバージョン

改訂 1-19 Mon Jul 18 2016 Mirek Jahoda
スマートカードのセクションの追加

改訂 1-18 Mon Jun 27 2016 Mirek Jahoda
OpenSCAP デーモンおよび Atomic Scan セクションの追加

改訂 1-17 Fri Jun 3 2016 Mirek Jahoda
さまざまな更新が追加された非同期リリース

改訂 1-16 Tue Jan 5 2016 Robert Krátký
7.2 GA 後の修正

改訂 1-15 Tue Nov 10 2015 Robert Krátký
7.2 GA リリース向けのバージョン。

改訂 1-14.18 Mon Nov 09 2015 Robert Krátký
さまざまな更新が追加された非同期リリース

改訂 1-14.17 Wed Feb 18 2015 Robert Krátký
7.1 GA リリース向けバージョン。

改訂 1-14.15 Fri Dec 06 2014 Robert Krátký
Red Hat カスタマーポータルでの並べ替えのための更新

改訂 1-14.13 Thu Nov 27 2014 Robert Krátký
POODLE vuln を反映する更新

改訂 1-14.12 Tue Jun 03 2014 Tomáš Čapek
7.0 GA リリース向けバージョン。