



Sobre Red Hat JBoss Enterprise Application Platform 7 [Esta é uma tradução automática] 7.1

Guia de Migração [Esta é uma tradução automática]

Para usar com Red Hat JBoss Enterprise Application Platform 7,1 [Esta é uma
tradução automática]

Sobre Red Hat JBoss Enterprise Application Platform 7 [Esta é uma tradução automática] 7.1 Guia de Migração [Esta é uma tradução automática]

Para usar com Red Hat JBoss Enterprise Application Platform 7,1 [Esta é uma tradução automática]

Nota Legal

Copyright © 2018 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

Resumo

Este guia fornece informações sobre como migrar seu aplicativo de versões anteriores de Red Hat JBoss Enterprise Application Platform.

Índice

CAPÍTULO 1. INTRODUÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	9
1.1. SOBRE RED HAT JBOSS ENTERPRISE APPLICATION PLATFORM 7 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	9
1.2. SOBRE O GUIA DE MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	9
1.3. SOBRE OS UPGRADES E AS MIGRAÇÕES [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	9
1.4. SOBRE O USO DO EAP_HOME NESTE DOCUMENTO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	10
CAPÍTULO 2. PREPARAÇÃO PARA A MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	12
2.1. VISÃO GERAL DA PREPARAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	12
2.2. REVISE OS RECURSOS DO JAVA EE 7 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	12
2.3. REVISE O QUE HÁ DE NOVO NO JBOSS EAP 7 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	12
Mudanças de Configuração de Mensagens no JBoss EAP 7.1 [Esta é uma tradução automática]	13
2.4. VERIFIQUE A LISTA DE RECURSOS PRETERIDOS E SEM SUPORTE [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	14
2.5. VERIFIQUE A DOCUMENTAÇÃO DE INTRODUÇÃO DO JBOSS EAP [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	14
2.6. ANÁLISE E PLANEJAMENTO DE MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	15
2.7. FAÇA UM BACKUP DOS DADOS IMPORTANTES E VERIFIQUE O ESTADO DO SERVIDOR [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	16
2.8. MIGRANDO UMA INSTALAÇÃO RPM [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	17
2.9. MIGRAR O JBOSS EAP PARA SER EXECUTADO COMO UM SERVIÇO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	17
CAPÍTULO 3. FERRAMENTAS PARA AUXILIAR A MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	18
3.1. USE O RED HAT APPLICATION MIGRATION TOOLKIT PARA ANALISAR APLICATIVOS PARA MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	18
3.2. UTILIZE A FERRAMENTA DE MIGRAÇÃO DO SERVIDOR JBOSS PARA MIGRAR AS CONFIGURAÇÕES DE SERVIDOR [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	18
CAPÍTULO 4. MUDANÇAS DE CONFIGURAÇÃO DE SERVIDOR [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	20
4.1. MUDANÇAS NA INSTALAÇÃO DO RPM [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	20
4.2. OPÇÕES DE MIGRAÇÃO DE CONFIGURAÇÃO DE SERVIDOR [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	20
Alterações de configurações de servidor EJB [Esta é uma tradução automática]	20
Operação de migração da CLI de gerenciamento [Esta é uma tradução automática]	20
4.3. OPERAÇÃO DE MIGRAÇÃO DA CLI DE GERENCIAMENTO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	20
Inicie o servidor e a CLI de gerenciamento	21
Migrar JacORB, Messaging e subsistemas Web	22
4.4. ALTERAÇÕES DE REGISTRO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	26
4.4.1. Alterações de prefixos dos registros de mensagens [Esta é uma tradução automática]	26
4.4.2. Alterações do manipulador de console do criador de logs raiz [Esta é uma tradução automática]	26
4.5. ALTERAÇÕES DE CONFIGURAÇÕES DO SERVIDOR WEB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	26
4.5.1. Substituição do subsistema web com Undertow [Esta é uma tradução automática]	26
4.5.2. Migrar as condições de regravação do JBoss Web [Esta é uma tradução automática]	27
4.5.3. Migrar propriedades de sistemas JBoss Web [Esta é uma tradução automática]	30
4.5.4. Atualizar o padrão de cabeçalho do log de acesso [Esta é uma tradução automática]	30
4.5.5. Migrar válvulas globais [Esta é uma tradução automática]	31
Migrar válvulas globais [Esta é uma tradução automática]	31
Procedimento de migração manual JDBCAccessLogValve	32
4.5.6. Alterações no comportamento do conjunto de cookies [Esta é uma tradução automática]	33

4.5.7. Alterações no comportamento de chamadas de método HTTP [Esta é uma tradução automática]	34
4.5.8. Mudanças no Comportamento do Módulo da Web Padrão no JBoss EAP 7.1 [Esta é uma tradução automática]	35
4.6. ALTERAÇÕES DE CONFIGURAÇÕES DO SERVIDOR JGROUPS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	35
4.6.1. JGroups é pre-definido para uma interface de rede privada [Esta é uma tradução automática]	35
4.6.2. Alterações de Canais JGroups [Esta é uma tradução automática]	36
4.7. ALTERAÇÕES DE CONFIGURAÇÃO DE SERVIDOR INFINISPAN [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	36
4.7.1. Alterações de configurações de cachês por padrão de Infinispan [Esta é uma tradução automática]	36
4.7.2. Alterações de estratégia de cachê Infinispan [Esta é uma tradução automática]	36
4.7.3. Configurando o Cache do Bean de Sessão de Estado Customizado para Passivação [Esta é uma tradução automática]	36
4.7.4. Alterações no Transporte do Contêiner de Cache Infinispan [Esta é uma tradução automática]	37
4.8. ALTERAÇÕES DE CONFIGURAÇÕES DE SERVIDOR EJB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	37
DuplicateServiceException no log do servidor [Esta é uma tradução automática]	38
4.9. ALTERAÇÕES DE CONFIGURAÇÃO DO SERVIDOR DE MENSAGENS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	38
4.9.1. Alterações de configuração do servidor de subsistema de mensagens [Esta é uma tradução automática]	38
Mudanças no gerenciamento de JMX [Esta é uma tradução automática]	39
Avisos para a operação de migração de subsistema de mensagem [Esta é uma tradução automática]	39
Alteração de comportamento de atributo forward-when-no-consumers	40
Alteração na diretiva de balanceamento de carga de cluster padrão	40
Alterações de configuração do servidor de subsistema de mensagens [Esta é uma tradução automática]	40
4.9.2. Migrar dados de mensagem [Esta é uma tradução automática]	40
4.9.2.1. Migrar dados de mensagens usando exportação e importação [Esta é uma tradução automática]	41
Exemplo: Access Log Format no JBoss EAP 6.4 [Esta é uma tradução automática]	41
Mudanças de Configuração de Mensagens no JBoss EAP 7.1 [Esta é uma tradução automática]	42
Migrar dados de mensagem [Esta é uma tradução automática]	43
Recuperando-se de uma falha de dados de mensagens de importação	44
4.9.2.2. Migrar dados de mensagem usando uma ponte JMS [Esta é uma tradução automática]	44
Configurar o servidor do JBoss EAP 6.4	44
Configurar o servidor do JBoss EAP 7.x de destino	45
Migrar dados de mensagem [Esta é uma tradução automática]	47
4.9.2.3. Mapeamento dos nomes da pasta de mensagens [Esta é uma tradução automática]	48
4.9.2.4. Fazendo backup dos dados da pasta de mensagens [Esta é uma tradução automática]	48
4.9.3. Migrar as destinações JMS [Esta é uma tradução automática]	49
4.9.4. Migrar interceptores de mensagem [Esta é uma tradução automática]	49
4.9.5. Substituir as configurações Netty Servlet [Esta é uma tradução automática]	50
4.9.6. Configurando um Adaptador de Recursos JMS Genérico [Esta é uma tradução automática]	50
4.9.7. Mudanças de Configuração de Mensagens no JBoss EAP 7.1 [Esta é uma tradução automática]	50
4.9.8. Alterações no Comportamento de Serialização do JMS entre Liberações [Esta é uma tradução automática]	50
4.10. MUDANÇAS NO GERENCIAMENTO DE JMX [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	52
4.11. ALTERAÇÕES DE CONFIGURAÇÕES DE SERVIDOR ORB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	53
4.12. MIGRAR A CONFIGURAÇÃO DE SUBSISTEMA DE THREADS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	56
4.13. MIGRAR A CONFIGURAÇÃO DE SUBSISTEMA REMOTO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	56
4.14. ALTERAÇÕES DE CONFIGURAÇÕES DE SERVIDOR WEBSOCKET [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	57
4.15. ALTERAÇÕES DO SERVIDOR SINGLE SIGN-ON [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	57

4.16. ALTERAÇÕES NA CONFIGURAÇÃO DA FONTE DE DADOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	58
4.16.1. Nome do Driver da Origem de Dados JDBC [Esta é uma tradução automática]	58
Driver contendo uma classe única	58
Driver contendo classes múltiplas	58
4.17. ALTERAÇÕES DE CONFIGURAÇÕES DE SERVIDOR DE SEGURANÇA [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	59
4.17.1. Mudanças no Comportamento de Segurança Legado entre o JBoss EAP 7.0 e o JBoss EAP 7.1 [Esta é uma tradução automática]	59
4.17.1.1. Alteração de status HTTP para domínios LDAP inacessíveis [Esta é uma tradução automática]	59
4.17.1.2. Habilitando o domínio de segurança do LDAP para analisar funções de um DN [Esta é uma tradução automática]	59
4.17.1.3. Mudanças no envio do certificado SSL do JBoss EAP para um servidor LDAP [Esta é uma tradução automática]	59
4.17.2. Mudanças no modo FIPS [Esta é uma tradução automática]	60
4.18. ALTERAÇÕES DO SUBSISTEMA DE TRANSAÇÕES [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	61
Alterações do subsistema de transações [Esta é uma tradução automática]	61
Alterações do subsistema de transações [Esta é uma tradução automática]	61
4.19. CHANGES TO MOD_CLUSTER CONFIGURATION [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	61
4.20. EXIBINDO ALTERAÇÕES NA CONFIGURAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	62
CAPÍTULO 5. ALTERAÇÕES NA MIGRAÇÃO DOS APLICATIVOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	63
5.1. ALTERAÇÕES NOS APLICATIVOS DE SERVIÇOS WEB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	63
5.1.1. Alterações de suporte JAX-RPC [Esta é uma tradução automática]	63
5.1.2. Alterações dos serviços web Apache CXF Spring [Esta é uma tradução automática]	64
Migrar interceptores de mensagem [Esta é uma tradução automática]	64
Recursos Apache CXF	64
Apache CXF HTTP Transport	65
5.1.3. Alterações WS-Security	66
5.1.4. Alterações de estruturas de módulos JBoss [Esta é uma tradução automática]	66
5.1.5. Alterações e requisitos de Bouncy Castle [Esta é uma tradução automática]	66
5.1.6. Estratégia de seleção de barramento Apache CXF [Esta é uma tradução automática]	67
5.1.7. Requisitos JAX-WS 2.2 para WebServiceRef [Esta é uma tradução automática]	67
5.1.8. Alterações de verificação CN IgnoreHttpsHost [Esta é uma tradução automática]	67
5.1.9. Configuração lado cliente e carregamento de classe [Esta é uma tradução automática]	67
5.1.10. O mecanismo de substituição de standards endossados Java está descontinuado. [Esta é uma tradução automática]	67
5.1.11. Especificação de descritor no arquivo EAR [Esta é uma tradução automática]	68
5.2. ATUALIZE A PORTA E CONECTOR URL REMOTO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	68
5.3. ALTERAÇÕES DE APLICATIVOS DE MENSAGENS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	68
5.3.1. Substituir ou atualizar descritores de implantação JMS [Esta é uma tradução automática]	68
5.3.2. Atualizar clientes externo JMS [Esta é uma tradução automática]	69
5.3.3. Substitua a API HornetQ [Esta é uma tradução automática]	70
5.4. ALTERAÇÕES DE APLICATIVOS JAX-RS E RESTEASY [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	70
5.4.1. Classes preteridas de RESTEasy [Esta é uma tradução automática]	70
Classes de corpo de mensagem e interceptores	70
API Cliente	72
StringConverter	74
5.4.2. Classes de RESTEasy removidas ou protegidas [Esta é uma tradução automática]	74
Métodos de adição de ResteasyProviderFactory	74
Classes suplementares removidas do RESTEasy 3	74
5.4.3. Outras alterações de RESTEasy [Esta é uma tradução automática]	74
SignedInput e SignedOutput	74

Alterações WS-Security	74
Filtros do lado do cliente	74
Suporte HTTP assíncrono	74
Cache do lado do servidor	74
Alterações do provedor Jackson [Esta é uma tradução automática]	75
Charset padrão UTF-8 no cabeçalho Content-Type	75
SerializableProvider	75
Solicitações correspondentes aos métodos de recursos	75
5.4.4. Alterações SPI RESTEasy [Esta é uma tradução automática]	76
Exceções SPI	76
InjectorFactory e Registro	76
5.4.5. Alterações do provedor Jackson [Esta é uma tradução automática]	76
5.4.6. Alterações à integração de Spring RESTEasy [Esta é uma tradução automática]	76
5.4.7. Alterações ao fornecedor RESTEasy Jettison JSON [Esta é uma tradução automática]	77
5.5. ALTERAÇÕES AOS APLICATIVOS CDI 1.2 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	77
Arquivos Bean	77
Esclarecimento de resolução de conversação	78
Resolução por observação	78
5.6. MIGRAR AS DEPENDÊNCIAS DE MÓDULO EXPLICITO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	79
Verifique dependências para disponibilidade	79
Dependências que requerem scan de anotação	79
5.7. ALTERAÇÕES DE MIGRAÇÃO JPA E HIBERNATE [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	79
5.7.1. Hibernate ORM 3.0 [Esta é uma tradução automática]	79
5.7.2. Hibernate ORM 4.0 - 4.3 [Esta é uma tradução automática]	80
5.7.3. Migrando para Hibernate ORM 5 [Esta é uma tradução automática]	80
Classes preteridas de RESTEasy [Esta é uma tradução automática]	80
Outras alterações às classes e pacotes	80
Type Handling	81
Alterações do subsistema de transações [Esta é uma tradução automática]	81
Alterações no Hibernate Search [Esta é uma tradução automática]	82
5.7.4. Migrando do Hibernate ORM 5.0 para o Hibernate ORM 5.1 [Esta é uma tradução automática]	82
Hibernate ORM 3.0 [Esta é uma tradução automática]	83
Mudanças no gerenciamento de JMX [Esta é uma tradução automática]	83
Mudanças no Cliente EJB no JBoss EAP 7 [Esta é uma tradução automática]	83
Mudanças de Configuração de Mensagens no JBoss EAP 7.1 [Esta é uma tradução automática]	83
5.8. ALTERAÇÕES NO HIBERNATE SEARCH [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	83
Alterações no Hibernate Search [Esta é uma tradução automática]	84
Indexing of id Fields of Embedded Relations	84
Alterações de migração JPA e Hibernate [Esta é uma tradução automática]	84
Alterações no Hibernate Search [Esta é uma tradução automática]	84
Alterações no Hibernate Search [Esta é uma tradução automática]	86
Lucene - Classes renomeadas e reempacotadas	86
Alterações no Hibernate Search [Esta é uma tradução automática]	86
Alterações no Hibernate Search [Esta é uma tradução automática]	86
Alterações no Hibernate Search [Esta é uma tradução automática]	87
Alterações no Hibernate Search [Esta é uma tradução automática]	87
Alterações no Hibernate Search [Esta é uma tradução automática]	87
Alterações no Hibernate Search [Esta é uma tradução automática]	87
Alterações no Hibernate Search [Esta é uma tradução automática]	88
Alterações que impactam integradores avançados	88
5.9. MIGRAR DE BEANS DE ENTIDADE PARA JPA [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	89
5.10. ALTERAÇÕES DE PROPRIEDADES DE PERSISTÊNCIA JPA [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	91

5.10.1. Mudanças de propriedade de persistência do JPA no JBoss EAP 7.1 [Esta é uma tradução automática]	91
5.11. MIGRAR O CÓDIGO DE CLIENTE EJB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	92
5.11.1. Mudanças no Cliente EJB no JBoss EAP 7 [Esta é uma tradução automática]	92
5.11.1.1. Atualize a porta de conexão remota padrão [Esta é uma tradução automática]	92
5.11.1.2. Atualizar o conector padrão [Esta é uma tradução automática]	93
5.11.2. Migrar código de cliente de nomeação remota [Esta é uma tradução automática]	93
5.11.3. Alterações Adicionais do Cliente EJB Introduzidas no JBoss EAP 7.1 [Esta é uma tradução automática]	94
5.12. MIGRAR CLIENTES PARA USAR O ARQUIVO DE CONFIGURAÇÃO DO WILDFLY [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	97
5.13. MIGRAR CONFIGURAÇÕES DE PLANOS DE IMPLEMENTAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	98
5.14. MIGRAR VÁLVULAS DE APLICATIVOS PERSONALIZADAS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	99
Migrar configurações SSL [Esta é uma tradução automática]	99
Migrar válvulas do autenticador [Esta é uma tradução automática]	100
5.15. ALTERAÇÕES DE APLICATIVOS DE SEGURANÇA [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	100
5.15.1. Migrar válvulas do autenticador [Esta é uma tradução automática]	100
5.15.2. Alterações de PicketLink [Esta é uma tradução automática]	100
5.15.3. Outras alterações de aplicativos de segurança [Esta é uma tradução automática]	100
5.16. ALTERAÇÕES DE JBOSS LOGGING [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	100
5.17. ALTERAÇÕES AO CÓDIGO JAVASERVER FACES (JSF) [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	101
Suporte para JSF 1.2 descontinuado	101
Problema de compatibilidade entre JSF 2.1 e JSF 2.2	101
5.18. ALTERAÇÕES AO CARREGAMENTO DE CLASSES DE MÓDULOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	102
5.19. ALTERAÇÕES À CLUSTERIZAÇÃO DE APLICATIVOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	103
5.19.1. Visão geral de novas funcionalidades de clusterização [Esta é uma tradução automática]	103
5.19.2. Alterações nas Web Session Clustering [Esta é uma tradução automática]	103
5.19.3. Alterações em Stateful Session EJB Clustering [Esta é uma tradução automática]	105
5.19.4. Alterações nos serviços de clustering [Esta é uma tradução automática]	106
5.19.5. Migrar Clustering HA Singleton [Esta é uma tradução automática]	106
CAPÍTULO 6. ALTERAÇÕES DIVERSAS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	107
6.1. ALTERAÇÕES NO MODO DE ENTREGA DO JBOSS EAP NATIVES E SERVIDOR HTTP APACHE [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	107
6.2. ALTERAÇÕES ÀS IMPLEMENTAÇÕES NO AMAZON EC2 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	108
6.3. REMOÇÃO DE IMPLEMENTAÇÃO DE APLICATIVOS QUE INCLUEM MÓDULOS COMPARTILHADOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	108
6.4. ALTERAÇÕES NOS SCRIPTS DO JBOSS EAP [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	109
6.5. REMOÇÃO DE SUPORTE OSGI [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	109
CAPÍTULO 7. MIGRANDO PARA O ELYTRON NO JBOSS EAP 7.1 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	110
7.1. VISÃO GERAL DE ELYTRON [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	110
7.2. MIGRAÇÃO DE COFRES E PROPRIEDADES SEGURAS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	111
7.2.1. Migrar Vaults para Proteger o Armazenamento de Credenciais [Esta é uma tradução automática]	111
Migrar clientes para usar o arquivo de configuração do WildFly [Esta é uma tradução automática]	111
Migrar Vaults para Proteger o Armazenamento de Credenciais [Esta é uma tradução automática]	112
Migrar Vaults para Proteger o Armazenamento de Credenciais [Esta é uma tradução automática]	112
7.2.2. Migrar propriedades de segurança para o Elytron [Esta é uma tradução automática]	114
7.3. MIGRAR CONFIGURAÇÃO DE AUTENTICAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	115

7.3.1. Migrar Autenticação e Autorização Baseadas em Propriedades para o Elytron [Esta é uma tradução automática]	115
7.3.1.1. Migrar a configuração baseada em propriedades do PicketBox para Elytron [Esta é uma tradução automática]	115
Migrar parcialmente expondo o domínio de segurança do PicketBox para Elytron	116
Migrar Autenticação e Autorização Baseadas em Propriedades para o Elytron [Esta é uma tradução automática]	118
7.3.1.2. Migrar a configuração baseada em propriedades herdadas para o Elytron [Esta é uma tradução automática]	120
7.3.2. Migrar a configuração de autenticação LDAP para o Elytron [Esta é uma tradução automática]	123
7.3.2.1. Migre a Autenticação LDAP Legada para Elytron [Esta é uma tradução automática]	126
7.3.3. Migrar a configuração de autenticação de banco de dados para o Elytron [Esta é uma tradução automática]	127
7.3.3.1. Migre a Autenticação de Banco de Dados Legada para Elytron [Esta é uma tradução automática]	129
7.3.4. Migração da autenticação Kerberos para o Elytron [Esta é uma tradução automática]	129
Migração da autenticação Kerberos para o Elytron [Esta é uma tradução automática]	130
Migração da autenticação Kerberos para o Elytron [Esta é uma tradução automática]	132
Migração da autenticação Kerberos para o Elytron [Esta é uma tradução automática]	134
Migração da autenticação Kerberos para o Elytron [Esta é uma tradução automática]	134
7.3.5. Migrar lojas compostas para Elytron [Esta é uma tradução automática]	136
Exemplo: Configuração do PicketBox LoginModule [Esta é uma tradução automática]	136
Exemplo: Comandos de Configuração do Reino de Segurança Legado [Esta é uma tradução automática]	137
Exemplo: Configuração do Realm de Segurança LDAP [Esta é uma tradução automática]	138
7.3.6. Migrar os domínios de segurança que usam o cache para Elytron [Esta é uma tradução automática]	140
Exemplo: Configuração do Domínio de Segurança em Cache do PicketBox [Esta é uma tradução automática]	140
Exemplo: Configuração do Domínio de Segurança em Cache do Elytron [Esta é uma tradução automática]	141
7.3.7. Migre a Segurança do JACC para o Elytron [Esta é uma tradução automática]	143
7.4. MIGRAR CLIENTES DE APLICATIVOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	143
7.4.1. Migrar uma configuração de cliente de nomeação para Elytron [Esta é uma tradução automática]	144
7.4.1.1. Migrar o cliente de nomeação usando a abordagem do arquivo de configuração [Esta é uma tradução automática]	144
7.4.1.2. Migrar o cliente de nomeação usando a abordagem programática [Esta é uma tradução automática]	145
7.4.2. Migrar um cliente EJB para Elytron [Esta é uma tradução automática]	146
7.4.2.1. Migrar o cliente EJB usando a abordagem do arquivo de configuração [Esta é uma tradução automática]	146
7.4.2.2. Migrar o cliente EJB usando a abordagem programática [Esta é uma tradução automática]	147
7.5. MIGRAR CONFIGURAÇÕES SSL [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	148
7.5.1. Migrar uma configuração SSL simples para o Elytron [Esta é uma tradução automática]	148
7.5.2. Migrar Autenticação SSL de Cliente-Certificado para Elytron [Esta é uma tradução automática]	150
CAPÍTULO 8. MIGRANDO A PARTIR DE VERSÕES MAIS ANTIGAS DO JBOSS EAP [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	154
8.1. MIGRANDO DO JBOSS EAP 5 PARA O JBOSS EAP 7 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	154
8.2. RESUMO DAS ALTERAÇÕES FEITAS EM CADA LANÇAMENTO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	154
8.3. REVISE O CONTEÚDO NOS GUIAS DE MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	155
8.4. REFERÊNCIA DE ATUALIZAÇÃO DO COMPONENTE JBOSS EAP 5 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	155
APÊNDICE A. MATERIAL DE REFERÊNCIA [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	163

A.1. AVISOS DE OPERAÇÃO DE MIGRAÇÃO DE SUBSISTEMA JACORB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	163
A.2. AVISOS PARA A OPERAÇÃO DE MIGRAÇÃO DE SUBSISTEMA DE MENSAGEM [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	164
Substituir os atributos preteridos broadcast-group ou discovery-group	167
A.3. AVISOS PARA OPERAÇÃO DE MIGRAÇÃO DE SUBSISTEMA WEB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	168
Avisos para operação de migração de subsistema web [Esta é uma tradução automática]	172
Atributos do Conector SSL da Web	172
Atributos de recursos estáticos da Web	172
Atributos de Recurso SSO da Web	173
Mapeamento dos atributos de mensagens [Esta é uma tradução automática]	173
Atributos do Conector da Web	174
A.4. MIGRAR REFERÊNCIA DE PROPRIEDADES DO SISTEMA JBOSS WEB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	174
A.5. COMPATIBILIDADE E INTEROPERABILIDADE ENTRE OS LANÇAMENTOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]	184
EJB remoto via IIOP	184
EJB remoto usando JNDI	184
EJB remoto utilizando @WebService	184
Cliente autônomo de mensagem	184
Migrar dados de mensagem [Esta é uma tradução automática]	185
Pontes JMS	185

CAPÍTULO 1. INTRODUÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

1.1. SOBRE RED HAT JBOSS ENTERPRISE APPLICATION PLATFORM 7 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Red Hat JBoss Enterprise Application Platform 7 (JBoss EAP) é uma plataforma de middleware baseada em padrões abertos e compatível com a especificação Java Enterprise Edition 7.

O JBoss EAP oferece uma estrutura modular que permite a habilitação de serviços apenas quando necessário, melhorando a velocidade da inicialização.

O Console de Gerenciamento e a CLI (Interface de Linha de Comando) de Gerenciamento tornam as edições dos arquivos de configuração XML desnecessárias e agregam a habilidade de utilizar script e automatizar tarefas.

O JBoss EAP fornece dois modos de operação para as instâncias do JBoss EAP: o servidor autônomo ou o domínio gerenciado. O modo de operação do servidor autônomo representa o JBoss EAP em execução como uma instância de servidor único. O modo de operação do domínio gerenciado permite o gerenciamento de múltiplas instâncias do JBoss EAP a partir de um ponto de controle único.

Além disso, o JBoss EAP inclui APIs e estruturas para o desenvolvimento rápido de aplicativos Java EE seguros e escaláveis.

1.2. SOBRE O GUIA DE MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

A finalidade deste guia é documentar as alterações necessárias para executar e implantar com êxito Red Hat JBoss Enterprise Application Platform 6 aplicações em Red Hat JBoss Enterprise Application Platform 7. Ele fornece informações sobre os novos recursos disponíveis nesta liberação, os recursos reprovados e não suportados e quaisquer atualizações de configuração de aplicativo e servidor que possam ser necessárias para evitar mudanças no comportamento do aplicativo.

Também fornece informações sobre ferramentas que podem ajudar na migração, como [Kit de Ferramentas de Migração de Aplicativos Red Hat](#), que simplifica a migração de aplicativos Java e [Ferramenta de Migração do JBoss Server](#), que atualiza a configuração do servidor.

Uma vez que o aplicativo foi implementado com êxito e esta sendo executado, pode-se fazer planos para atualizar componentes individuais para utilizar as novas funções e recursos de JBoss EAP 7.

Se você planeja migrar seus aplicativos do JBoss EAP 5 diretamente para o JBoss EAP 7, veja [Migrando de versões mais antigas do JBoss EAP](#).

1.3. SOBRE OS UPGRADES E AS MIGRAÇÕES [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Lançamentos Principais [Esta é uma tradução automática]

Uma grande atualização ou migração é necessária quando um aplicativo é movido de um lançamento principal para outro, por exemplo, do JBoss EAP 6.4 para o JBoss EAP 7.0. Se um aplicativo seguir as especificações do Java EE, não acessar APIs reprovadas e não contiver código proprietário, pode ser

possível executar o aplicativo no JBoss EAP 7 sem nenhuma alteração no código do aplicativo. No entanto, a configuração do servidor foi alterada no JBoss EAP 7 e requer migração. Esse tipo de migração é abordado neste guia.

Updates de Manutenção [Esta é uma tradução automática]

O JBoss EAP fornece periodicamente lançamentos pontuais, que são pequenas atualizações que incluem correções de bugs, correções de segurança e novos recursos. Informações sobre as alterações feitas em uma liberação pontual estão documentadas neste guia e no [Notas de versão 7.1.0](#).

Você pode usar a Ferramenta de Migração do JBoss Server para atualizar automaticamente de um lançamento pontual para outro, por exemplo, do JBoss EAP 7.0 para o JBoss EAP 7.1. Para obter informações sobre como configurar e executar a ferramenta, consulte [Usando a Ferramenta de Migração do JBoss Server](#).

Se preferir, você pode executar uma atualização manual da configuração do servidor. Instruções sobre como realizar uma atualização manual estão documentadas em [Atualizando o JBoss EAP](#) no JBoss EAP *Guia de patch e atualização*.

Patches Cumulativos [Esta é uma tradução automática]

O JBoss EAP também fornece periodicamente patches cumulativos que contêm correções de bug e segurança. Os patches cumulativos aumentam a liberação pelo último dígito, por exemplo, de 7.1.0 para 7.1.1. A instalação do patch é endereçada no JBoss EAP [Guia de patch e atualização](#).

1.4. SOBRE O USO DO EAP_HOME NESTE DOCUMENTO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Neste documento, a variável **EAP_HOME** é usado para denotar o caminho para a instalação do JBoss EAP. Substitua esta variável pelo caminho atual para a sua instalação do JBoss EAP.

- Se você instalou o JBoss EAP usando o método de instalação ZIP, o diretório de instalação é o **jboss-eap-7.1** diretório onde você extraiu o arquivo ZIP.
- Se você instalou o JBoss EAP usando o método de instalação RPM, o diretório de instalação **/opt/rh/eap7/root/usr/share/wildfly/**.
- Se você usou o instalador para instalar o JBoss EAP, o caminho padrão para **EAP_HOME** é **\${user.home}/EAP-7.1.0**:
 - Para o Red Hat Enterprise Linux, Solaris e HP-UX: **/home/USER_NAME/EAP-7.1.0/**
 - Para o Microsoft Windows: **C:\Users\USER_NAME\EAP-7.1.0**
- Se você usou o instalador do JBoss Developer Studio para instalar e configurar o servidor JBoss EAP, o caminho padrão para **EAP_HOME** é **\${user.home}/jbdevstudio/runtimes/jboss-eap**:
 - Para o Red Hat Enterprise Linux: **/home/USER_NAME/jbdevstudio/runtimes/jboss-eap/**
 - Para o Microsoft Windows: **C:\Users\USER_NAME\jbdevstudio\runtimes\jboss-eap** ou **C:\Documents and Settings\USER_NAME\jbdevstudio\runtimes\jboss-eap**



NOTA

EAP_HOME não é uma variável de ambiente. ***JBoss_HOME*** é a variável de ambiente usada em scripts.

CAPÍTULO 2. PREPARAÇÃO PARA A MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

2.1. VISÃO GERAL DA PREPARAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

No JBoss EAP 7, foi feito um esforço para fornecer compatibilidade retroativa para os aplicativos do JBoss EAP 6. No entanto, se seu aplicativo usar recursos que foram reprovados ou a funcionalidade que foi removida do JBoss EAP 7, talvez seja necessário fazer alterações no código do aplicativo.

Além disso, várias coisas mudaram nesta versão que podem impactar a implantação de aplicativos do JBoss EAP 7. É recomendável que você faça alguma pesquisa e planejamento antes de tentar migrar seu aplicativo.

- Familiarize-se com o [recursos do Java EE 7](#).
- Reveja [O que há de novo no JBoss EAP 7](#).
- Revise a lista de [recursos obsoletos e sem suporte](#).
- Revise o material no JBoss EAP 7 [Guia de Introdução](#).
- Dê uma olhada no [ferramentas que podem ajudar com tarefas de migração](#).

Depois que você se familiarizar com as alterações dos recursos, com os materiais de desenvolvimento e as ferramentas que podem auxiliar os seus esforços de migração, você pode começar a avaliar os seus aplicativos e a configuração do seu servidor para determinar as alterações necessárias para a execução do JBoss EAP 7.

2.2. REVISE OS RECURSOS DO JAVA EE 7 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

O Java EE 7 inclui diversas melhorias para facilitar o desenvolvimento e a execução dos aplicativos com muitos recursos em nuvens públicas e privadas. Ele incorpora novos recursos e os padrões mais recentes, como HTML5, WebSocket, JSON, Batch e Concurrency Utilities. As atualizações incluem JPA 2.1, JAX-RS 2.0, Servlet 3.1, Expression Language 3.0, JMS 2.0, JSF 2.2, EJB 3.2, CDI 1.2 e Bean Validation 1.1.

Você pode encontrar mais informações sobre o Java EE 7, incluindo tutoriais, no site da Oracle: [Java EE em um relance](#)

2.3. REVISE O QUE HÁ DE NOVO NO JBOSS EAP 7 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

O JBoss EAP 7 inclui algumas melhorias e upgrades importantes com relação ao último lançamento.

Java EE 7

O JBoss EAP 7 é uma implementação certificada do Java EE 7, atendendo tanto ao Perfil da Web quanto às especificações completas da plataforma. Também inclui suporte para as últimas iterações do CDI 1.2 e do Web Sockets 1.1.

Undertow

O Undertow é um servidor web eficaz, flexível e leve que faz parte do JBoss EAP 7, substituindo o JBoss Web. Escrito em Java, ele foi designado para máxima escalabilidade e produtividade e fornece suporte às tecnologias da web mais recentes, como o novo padrão HTTP/2.

Apache ActiveMQ Artemis

O Apache ActiveMQ Artemis é o novo provedor de mensagens interno do JBoss EAP 7. Baseado na doação de código do HornetQ, este subprojeto de Apache fornece um desempenho excelente baseado em uma arquitetura não bloqueadora comprovada.

IronJacamar 1.2

O IronJacamar mais recente fornece um suporte estável e com muitos recursos ao JCA e DataSources.

JBossWS 5

A quinta geração do JBossWS é um grande avanço, trazendo novos recursos e melhorias de desempenho para os serviços web do JBoss EAP 7.

Alterações SPI RESTEasy [Esta é uma tradução automática]

O JBoss EAP 7 inclui a última geração de RESTEasy. Ele vai além das APIs REST Java EE padrão (JAX-RS 2.0), fornecendo várias extensões úteis, como JSON Web Encryption, Jackson, JSON-P e Jettison.

OpenJDK ORB

O JBoss EAP 7 substituiu a implementação do JacORB IIOP com uma ramificação downstream do OpenJDK ORB, gerando melhor interoperabilidade para o JVM ORB e o Java EE RI.

Clusterização Repleta de Recursos

O suporte de clusterização foi altamente refatorado no JBoss EAP 7 e inclui várias APIs públicas para o acesso de aplicativos.

Introdução [Esta é uma tradução automática]

Ao utilizar o upgrade do HTTP, o JBoss EAP 7 mudou praticamente todos os protocolos para serem multiplexados via duas portas HTTP apenas: uma porta de gerenciamento (9990) e uma porta do aplicativo (8080).

Registro em Log Avançado

Agora a API de gerenciamento fornece suporte para a habilidade de listar e visualizar os arquivos de log disponíveis em um servidor, ou até mesmo definir formatadores personalizados além do formatador padrão. A configuração do registro em log da implantação também foi amplamente melhorada.

Para uma lista completa dos novos recursos introduzidos no JBoss EAP 7.0, veja [Novos recursos e aprimoramentos](#) no JBoss EAP *Notas de versão 7.0.0*.

Mudanças de Configuração de Mensagens no JBoss EAP 7.1 [Esta é uma tradução automática]

Élito

O Elytron, baseado no projeto WildFly Elytron, é o novo framework de segurança do JBoss EAP 7.1. Ele é projetado para unificar a segurança em todo o servidor de aplicativos.

Mudanças no gerenciamento de JMX [Esta é uma tradução automática]

O console de gerenciamento foi aprimorado para fornecer a capacidade de configurar mais subsistemas, fornecer **transaction** subsistema e métricas de recurso de transação, além de gerenciar muitas configurações adicionais.

Mudanças no gerenciamento de JMX [Esta é uma tradução automática]

A CLI de gerenciamento fornece suporte aprimorado para resposta e anexos de arquivo, configuração de módulo e suporte a depuração através do **echo-command** argumento.

Para a lista completa dos novos recursos introduzidos no JBoss EAP 7.1, veja [Novos recursos e aprimoramentos](#) no *Notas de versão 7.1.0* no Portal do Cliente Red Hat.

2.4. VERIFIQUE A LISTA DE RECURSOS PRETERIDOS E SEM SUPORTE [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Antes de migrar seu aplicativo para o JBoss EAP 7, esteja ciente de que alguns recursos que estavam disponíveis em versões anteriores do JBoss EAP podem ser preteridos ou não mais suportados.

O suporte para algumas tecnologias foi removido devido ao alto custo de manutenção, pouco interesse da comunidade e soluções alternativas muito melhores. O que segue é um breve resumo de alguns dos recursos sem suporte.

Migrar de beans de entidade para JPA [Esta é uma tradução automática]

EJB entity beans não são mais suportados. Caso seu aplicativo utilizar EJB entity beans, você deve migrar o código para usar JPA, que oferece uma API muito mais eficiente e flexível.

JAX-RPC

Como JAX-WS oferece uma solução muito mais precisa e completa, o código escrito para JAX-RPC deve ser migrado para utilizar JAX-WS.

JSR-88

A especificação de API de implantação de aplicativos Java EE (JSR-88), que definia um contrato para habilitar ferramentas de múltiplos fornecedores para configurar e implantar aplicativos em qualquer produto de plataforma Java EE, não foi amplamente adotada. Você deve utilizar outra opção do JBoss EAP com suporte para implantação de aplicativos, como o console de gerenciamento, a CLI de gerenciamento, o escâner de implantação ou Maven.

Configurando um Adaptador de Recursos JMS Genérico [Esta é uma tradução automática]

- No JBoss EAP 7.0, a capacidade de configurar um adaptador de recursos JMS genérico para conectar-se a um provedor JMS não é suportada.
- No JBoss EAP 7.1, a capacidade de configurar um adaptador de recursos JMS genérico para conectar-se a um provedor JMS é suportada.

Para obter uma lista completa de recursos obsoletos e não suportados no JBoss EAP 7.0, consulte [Funcionalidade não suportada e descontinuada](#) no JBoss EAP *Notas de versão 7.0.0* no Portal do Cliente Red Hat.

Para obter uma lista completa de recursos obsoletos e não suportados no JBoss EAP 7.1, consulte [Funcionalidade não suportada e descontinuada](#) no JBoss EAP *Notas de versão 7.1.0* no Portal do Cliente Red Hat.

2.5. VERIFIQUE A DOCUMENTAÇÃO DE INTRODUÇÃO DO JBOSS EAP [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Certifique-se de revisar o JBoss EAP [Guia de Introdução](#). Contém as seguintes informações importantes:

- Como baixar e instalar o JBoss EAP 7
- Como baixar e instalar o Red Hat JBoss Developer Studio

- Como configurar o Maven para seu ambiente de desenvolvimento, gerenciar dependências de projetos e configurar seus projetos para utilizar artefatos da Estrutura de Produtos (BOM) do JBoss EAP
- Como baixar e executar os aplicativos de exemplo de início rápido que são enviados com o produto.

2.6. ANÁLISE E PLANEJAMENTO DE MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Cada configuração de aplicativo e servidor é exclusiva e você deve entender completamente os componentes e a arquitetura do aplicativo existente e da plataforma do servidor antes de tentar a migração. Seu plano de migração deve incluir um roteiro detalhado para testes e distribuição para produção que leve em consideração as informações a seguir.

Identifica as pessoas responsáveis pela migração

Identifique as partes interessadas, gerentes de projeto, desenvolvedores, administradores e outras pessoas responsáveis pela migração.

Verifica a configuração da plataforma do servidor de aplicativos e hardware

Analisa o servidor de aplicativos existente e configurações de plataforma para determinar como eles são impactados pelas alterações de recursos no JBoss EAP 7. A análise deve incluir os seguintes itens:

- Sistemas operacionais e versões
- Banco de dados utilizados pelos aplicativos
- Servidores da Web
- Arquitetura de segurança
- Número e tipo de processadores
- Quantidade de memória
- Quantidade de armazenamento no disco físico
- Migrar dados de mensagem [Esta é uma tradução automática]
- Outros componentes que podem ser afetados pela migração

Verifique o ambiente de produção atual

Você deve planejar a criação do ambiente de produção o mais semelhante quanto possível para testar e preparar o processo de migração.

- Leve em conta quaisquer configurações de cluster. Veja [Atualizando um Cluster](#) no JBoss EAP *Guia de patch e atualização* para obter mais informações sobre como migrar clusters.
- Caso você esteja atualmente executando em um domínio gerenciado amplo, considere uma abordagem gradual de migração.
- Determine se você precisa migrar algum banco de dados ou dados de mensagens

Examine e entenda os aplicativos existentes

Examine minuciosamente o aplicativo JBoss EAP 6 existente. Esteja totalmente familiarizado com sua arquitetura, funções, recursos e componentes, incluindo:

- A versão JVM
- Integração com outros componentes middleware de servidores dos aplicativos da Red Hat
- Integração com software proprietário de terceiros
- Utilização de recursos preteridos que precisarão ser substituídos
- Configuração de aplicativos incluindo descritor de implementação, JNDI, persistência, configuração JDBC e pooling, tópicos JMS e filas, e registro em log.

Identificar quaisquer incompatibilidades de código ou configuração que necessitarão modificações durante a migração para JBoss EAP 7.

Crie um plano teste detalhado

- O plano deve incluir teste de regressão e requisitos de critérios de aceitação.
- Deve também incluir teste de desempenho.
- Estabeleça um ambiente de teste tão semelhante quanto possível do ambiente de produção para testar a migração antes do lançamento para produção.
- Tenha certeza de que criou um backup e um plano de backup!

Revise o conteúdo nos Guias de Migração [Esta é uma tradução automática]

- Avalie as competências da equipe de desenvolvimento e planeje treinamento ou auxílio de consultoria suplementar.
- Esteja ciente de que serão necessários hardwares suplementares e outros recursos para preparar e testar durante o processo de migração até que a tarefa esteja completa.
- Determine se qualquer treinamento formal será necessário. Caso afirmativo, adicione ao cronograma.

Execute o plano

Reúna os recursos necessários e implemente o plano de migração.



IMPORTANTE

Antes de fazer quaisquer modificações em seus aplicativos, tenha certeza de que criou uma cópia de backup.

2.7. FAÇA UM BACKUP DOS DADOS IMPORTANTES E VERIFIQUE O ESTADO DO SERVIDOR [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

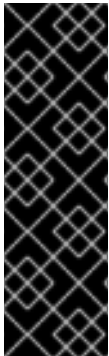
Antes de migrar seu aplicativo, você precisa estar ciente dos possíveis problemas:

- A migração pode remover pastas temporárias. Qualquer implementação armazenada no **data/content/** O diretório deve ser submetido a backup antes da migração e restaurado após a conclusão. Caso contrário, o servidor não será iniciado devido ao conteúdo ausente.

- Antes da migração, manipule todas as transações abertas e exclua **data/tx-object-store/** diretório de transação.
- Os dados do temporizador persistente **data/timer-service-data** deve ser verificado para determinar se é compatível. Antes da migração, revise o **deployment-*** arquivos nesse diretório para determinar quais cronômetros ainda estão em uso.

Certifique-se também de fazer o backup das configurações do servidor atual e aplicativos antes de começar.

2.8. MIGRANDO UMA INSTALAÇÃO RPM [ESTA É UMA TRADUÇÃO AUTOMÁTICA]



IMPORTANTE

Não é fornecido suporte para mais de uma instância de RPM instalada de JBoss EAP em um único Servidor Red Hat Enterprise Linux. Como resultado, nós recomendamos que você migre sua instalação do JBoss EAP para uma nova máquina quando migrar para JBoss EAP 7.

Quando migrar uma instalação RPM de JBoss EAP a partir do JBoss EAP 6 para JBoss EAP 7, certifique-se que JBoss EAP 7 esteja instalado em uma máquina que não tem uma instalação RPM de JBoss EAP existente.

Para instalar o JBoss EAP 7 usando RPMs, veja o JBoss EAP [Guia de instalação](#).

O aviso de migração neste guia também se aplica à migração de instalações RPM do JBoss EAP, mas você pode precisar alterar algumas etapas (como iniciar o JBoss EAP) para adequar-se a uma instalação RPM em comparação a uma instalação ZIP ou de instalação.

2.9. MIGRAR O JBOSS EAP PARA SER EXECUTADO COMO UM SERVIÇO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Se você executar o JBoss EAP 6 como um serviço, não deixe de revisar [Configurando o JBoss EAP para Executar como um Serviço](#) no JBoss EAP [Guia de instalação](#) para obter instruções de configuração atualizadas para o JBoss EAP 7.

CAPÍTULO 3. FERRAMENTAS PARA AUXILIAR A MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

3.1. USE O RED HAT APPLICATION MIGRATION TOOLKIT PARA ANALISAR APLICATIVOS PARA MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

O Red Hat Application Migration Toolkit (RHAMT) é um conjunto de ferramentas extensível e personalizável baseado em regras que ajuda a simplificar a migração de aplicativos Java. Ele analisa as APIs, tecnologias e arquiteturas usadas pelos aplicativos que você planeja migrar e fornece relatórios detalhados de migração para cada aplicativo. Esses relatórios fornecem as informações a seguir.

- Explicações detalhadas das alterações necessárias para migração
- Se as alterações relatadas são obrigatórias ou opcionais
- Se as alterações relatadas são complexas ou triviais
- Links para códigos que necessitam de alterações para migração
- Dicas e links de informação sobre como fazer as alterações necessárias
- Uma estimativa do nível de esforço para cada problema de migração encontrado e o esforço total estimado para migrar o aplicativo

Você pode usar o RHAMT para analisar o código e a arquitetura de seus aplicativos do JBoss EAP 6 antes de migrá-los para o JBoss EAP 7. O conjunto de regras RHAMT para migração do JBoss EAP 6 para o JBoss EAP 7 relata descritores XML e código e parâmetros específicos da aplicação. precisam ser substituídos por uma configuração alternativa ao migrar para o JBoss EAP 7.

Para obter mais informações sobre como usar o Red Hat Application Migration Toolkit para analisar seus aplicativos do JBoss EAP 6, consulte o Red Hat Application Migration Toolkit [Guia de Introdução](#).

3.2. UTILIZE A FERRAMENTA DE MIGRAÇÃO DO SERVIDOR JBOSS PARA MIGRAR AS CONFIGURAÇÕES DE SERVIDOR [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

A Ferramenta de Migração do JBoss Server é o método preferido para atualizar a configuração do seu servidor para incluir os novos recursos e configurações no JBoss EAP 7 enquanto mantém sua configuração existente. A Ferramenta de Migração do JBoss Server lê seus arquivos de configuração existentes do servidor JBoss EAP e adiciona configurações para quaisquer novos subsistemas, atualiza as configurações de subsistema existentes com novos recursos e remove quaisquer configurações de subsistema obsoletas.

Utilize a ferramenta de migração do servidor JBoss para migrar as configurações de servidor [Esta é uma tradução automática]

Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

A Ferramenta de Migração do JBoss Server vem com o JBoss EAP 7.1, portanto, não é necessário nenhum download separado ou instalação. Você executa a ferramenta executando o **jboss-server-migration** roteiro localizado no **EAP_HOME/bin** diretório. Para obter mais informações sobre como configurar e executar a ferramenta, consulte [Usando a Ferramenta de Migração do JBoss Server](#).

É recomendado que você use esta versão da Ferramenta de Migração do JBoss Server para migrar a configuração do seu servidor para o JBoss EAP 7.1, já que esta versão da ferramenta é [suportado](#).

Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

Para migração para o JBoss EAP 7.0, você deve baixar a última distribuição binária da Ferramenta de Migração do JBoss Server a partir do [Ferramenta de Migração do Servidor GitHub](#). Para obter informações sobre como executar a ferramenta, consulte a Ferramenta de Migração do JBoss Server. [Guia de usuario](#).



IMPORTANTE

Esta versão da Ferramenta de Migração do JBoss Server não é suportada. Em vez de usar esta versão para migrar a configuração do seu servidor para o JBoss EAP 7.0, é recomendado que você use o [versão suportada da ferramenta](#) para migrar sua configuração de servidor diretamente para o JBoss EAP 7.1.

CAPÍTULO 4. MUDANÇAS DE CONFIGURAÇÃO DE SERVIDOR [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

4.1. MUDANÇAS NA INSTALAÇÃO DO RPM [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

No JBoss EAP 6, o caminho padrão para a instalação do RPM era o `/usr/share/jbossas/` diretório.

O JBoss EAP 7 foi construído para [Biblioteca de coleções de software](#) convenções. O diretório raiz de coleções de software é normalmente localizado no `/opt/` diretório para evitar possíveis conflitos entre as Coleções de Software ea instalação do sistema básico. O uso do `/opt/` O diretório é recomendado pelo Filesystem Hierarchy Standard (FHS). Como resultado, o caminho padrão para a instalação do RPM foi alterado para `/opt/rh/eap7/root/usr/share/wildfly/` no JBoss EAP 7.

4.2. OPÇÕES DE MIGRAÇÃO DE CONFIGURAÇÃO DE SERVIDOR [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Para migrar a configuração do servidor do JBoss EAP 6 para o JBoss EAP 7, você pode usar o [Ferramenta de Migração do JBoss Server](#) ou você pode executar uma migração manual com a ajuda do CLI de gerenciamento `migrate` Operação.

Alterações de configurações de servidor EJB [Esta é uma tradução automática]

A Ferramenta de Migração do JBoss Server é o método preferido para atualizar sua configuração para incluir os novos recursos e configurações no JBoss EAP 7, enquanto mantém sua configuração existente. Para obter informações sobre como configurar e executar a ferramenta, consulte [Usando a Ferramenta de Migração do JBoss Server](#).

Operação de migração da CLI de gerenciamento [Esta é uma tradução automática]

Você pode usar o CLI de gerenciamento `migrate` operação para atualizar o `jacorb`, `messaging`, e `web` subsistemas no arquivo de configuração do servidor do JBoss EAP 6 para permitir que eles sejam executados no novo release, mas esteja ciente de que o resultado não é uma configuração completa do JBoss EAP 7. Por exemplo:

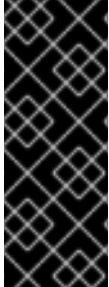
- A operação não atualiza o original `remote` configurações de protocolo e porta para o novo `http-remoting` e novas configurações de porta agora usadas no JBoss EAP 7.
- A configuração não inclui os novos subsistemas de JBoss EAP ou funcionalidades tais como implantações singleton clusterizadas ou desligamento ordenado.
- A configuração não inclui os novos recursos de Java EE 7 como processamento batch.
- o `migrate` operação não migra o `ejb3` configuração do subsistema. Para obter informações sobre possíveis problemas de migração do EJB, consulte [Alterações na Configuração do Servidor EJB](#).

Para mais informações sobre como usar o `migrate` operação para migrar a configuração do servidor, consulte [Operação de Migração do CLI de Gerenciamento](#).

4.3. OPERAÇÃO DE MIGRAÇÃO DA CLI DE GERENCIAMENTO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Você pode usar a CLI de gerenciamento para atualizar seus arquivos de configuração do servidor JBoss

EAP 6 para executar no JBoss EAP 7. O CLI de gerenciamento **migrate** operação para atualizar automaticamente **jacorb**, **messaging**, e **web** subsistemas do release anterior para a nova configuração. Você também pode executar o **describe-migration** operação para o **jacorb**, **messaging**, e **web** subsistemas para revisar as alterações de configuração de migração propostas antes de executar a migração. Não há substitutos para o **cmp**, **jaxr**, ou **threads** subsistemas e devem ser removidos da configuração do servidor.



IMPORTANTE

Veja [Opções de Migração de Configuração do Servidor](#) para limitações do **migrate** Operação. A Ferramenta de Migração do JBoss Server é o método preferido para atualizar sua configuração para incluir os novos recursos e configurações no JBoss EAP 7, enquanto mantém sua configuração existente. Para obter informações sobre como configurar e executar a ferramenta, consulte [Usando a Ferramenta de Migração do JBoss Server](#).

Tabela 4.1. Migração de subsistemas e operação de Cli de gerenciamento [Esta é uma tradução automática]

Subsistema JBoss EAP 6	Subsistema JBoss EAP 7	Operação de migração da CLI de gerenciamento [Esta é uma tradução automática]
cmp	sem substituição	remove
jacorb	iiop-openjdk	migrate
jaxr	sem substituição	remove
messaging	messaging-activemq	migrate
threads	sem substituição	remove
web	undertow	migrate

Inicie o servidor e a CLI de gerenciamento

Siga as etapas abaixo para atualizar sua configuração do servidor JBoss EAP 6 para executar em JBoss EAP 7.

1. Antes de começar, revise [Faça backup de dados importantes e revise o estado do servidor](#). Ele contém informações importantes sobre como garantir que o servidor esteja em bom estado e que os arquivos apropriados sejam armazenados em backup.
2. Inicie o servidor JBoss EAP 7 com a configuração do JBoss EAP 6.
 - a. Faça o backup dos arquivos do servidor JBoss EAP 7.
 - b. Copie o arquivo de configuração da versão anterior no diretório do JBoss EAP 7.

```
$ cp EAP6_HOME/standalone/configuration/standalone-full.xml
EAP7_HOME/standalone/configuration
```

- c. Navegue até o diretório de instalação do JBoss EAP 7 e inicie o servidor com o **--start-mode=admin-only** argumento.

```
$ bin/standalone.sh -c standalone-full.xml --start-mode=admin-only
```

NOTA

Você verá o seguinte **org.jboss.as.controller.management-operation** ERROS no log do servidor quando você inicia o servidor. Esses erros são esperados e indicam que as configurações do subsistema legado devem ser removidas ou migradas para o JBoss EAP 7.

- WFLYCTL0402: Subsystems [cmp] provided by legacy extension 'org.jboss.as.cmp' are not supported on servers running this version. Both the subsystem and the extension must be removed or migrated before the server will function.
- WFLYCTL0402: Subsystems [jacob] provided by legacy extension 'org.jboss.as.jacob' are not supported on servers running this version. Both the subsystem and the extension must be removed or migrated before the server will function.
- WFLYCTL0402: Subsystems [jaxr] provided by legacy extension 'org.jboss.as.jaxr' are not supported on servers running this version. Both the subsystem and the extension must be removed or migrated before the server will function.
- WFLYCTL0402: Subsystems [messaging] provided by legacy extension 'org.jboss.as.messaging' are not supported on servers running this version. Both the subsystem and the extension must be removed or migrated before the server will function.
- WFLYCTL0402: Subsystems [threads] provided by legacy extension 'org.jboss.as.threads' are not supported on servers running this version. Both the subsystem and the extension must be removed or migrated before the server will function.
- WFLYCTL0402: Subsystems [web] provided by legacy extension 'org.jboss.as.web' are not supported on servers running this version. Both the subsystem and the extension must be removed or migrated before the server will function.

3. Abra um novo terminal, navegue até o diretório de instalação do JBoss EAP 7 e inicie a CLI de gerenciamento usando o **--controller=remote://localhost:9999** argumentos.

```
$ bin/jboss-cli.sh --connect --controller=remote://localhost:9999
```

Migrar JacORB, Messaging e subsistemas Web

1. Para revisar as alterações de configuração que serão feitas no subsistema antes de executar a migração, execute o **describe-migration** Operação.
o **describe-migration** operação usa a seguinte sintaxe.

```
/subsystem=SUBSYSTEM_NAME:describe-migration
```

O exemplo a seguir descreve as mudanças de configuração feitas no JBoss EAP 6.4 **standalone-full.xml** arquivo de configuração quando ele é migrado para o JBoss EAP 7. As entradas foram removidas da saída para melhorar a legibilidade e economizar espaço.

Exemplo: operação **describe-migration** [Esta é uma tradução automática]

```
/subsystem=messaging:describe-migration
{
  "outcome" => "success",
  "result" => {
    "migration-warnings" => [],
    "migration-operations" => [
      {
        "operation" => "add",
        "address" => [{"extension" =>
"org.wildfly.extension.messaging-activemq"}],
        "module" => "org.wildfly.extension.messaging-
activemq"
      },
      {
        "operation" => "add",
        "address" => [{"subsystem" => "messaging-
activemq"}]
      },
      <!-- *** Entries removed for readability *** -->
      {
        "operation" => "remove",
        "address" => [{"subsystem" => "messaging"}]
      },
      {
        "operation" => "remove",
        "address" => [{"extension" =>
"org.jboss.as.messaging"}]
      }
    ]
  }
}
```

2. Execute o **migrate** operação para migrar a configuração do subsistema para o subsistema que a substitui no JBoss EAP 7. A operação usa a seguinte sintaxe.

```
/subsystem=SUBSYSTEM_NAME:migrate
```



NOTA

o **messaging** subsistema **describe-migration** e **migrate** operações permitem que você passe um argumento para configurar o acesso por clientes legados. Para obter mais informações sobre a sintaxe de comando, consulte [Migração de Subsistema de Mensagens e Compatibilidade de Encaminhamento](#).

3. Verifique o desfecho e o resultado do comando. Certifique-se que a operação foi completada com êxito e não há entradas de "migration-warning". Isto significa que a configuração de migração para o subsistema está completa.

Exemplo: operação de migração com êxito sem alertas [Esta é uma tradução automática]

```
/subsystem=messaging:migrate
{
  "outcome" => "success",
  "result" => {"migration-warnings" => []}
}
```

Se você vir entradas de "avisos de migração" no log, isso indica que a migração da configuração do servidor foi concluída com êxito, mas não conseguiu migrar todos os elementos e atributos. Você deve seguir as sugestões fornecidas pelos "avisos de migração" e executar comandos CLI de gerenciamento adicionais para modificar essas configurações. O seguinte é um exemplo de um **migrate** operação que retorna "avisos de migração".

Exemplo: operação de migração com alertas [Esta é uma tradução automática]

```
/subsystem=messaging:migrate
{
  "outcome" => "success",
  "result" => {"migration-warnings" => [
    "WFLYMSG0080: Could not migrate attribute group-address from
resource [
  (\\"subsystem\\" => \\"messaging-activemq\\"),
  (\\"server\\" => \\"default\\"),
  (\\"broadcast-group\\" => \\"groupB\\")
]. Use instead the socket-binding attribute to configure this
broadcast-group.",
    "WFLYMSG0080: Could not migrate attribute group-port from
resource [
  (\\"subsystem\\" => \\"messaging-activemq\\"),
  (\\"server\\" => \\"default\\"),
  (\\"broadcast-group\\" => \\"groupB\\")
]. Use instead the socket-binding attribute to configure this
broadcast-group.",
    "WFLYMSG0080: Could not migrate attribute local-bind-address
from resource [
  (\\"subsystem\\" => \\"messaging-activemq\\"),
  (\\"server\\" => \\"default\\"),
  (\\"broadcast-group\\" => \\"groupA\\")
]. Use instead the socket-binding attribute to configure this
broadcast-group.",
    "WFLYMSG0080: Could not migrate attribute local-bind-port
from resource [
  (\\"subsystem\\" => \\"messaging-activemq\\"),
  (\\"server\\" => \\"default\\"),
  (\\"broadcast-group\\" => \\"groupA\\")
]. Use instead the socket-binding attribute to configure this
broadcast-group.",
    "WFLYMSG0080: Could not migrate attribute group-address from
resource [
```

```

        ("subsystem\" => \"messaging-activemq\"),
        ("server\" => \"default\"),
        ("broadcast-group\" => \"groupA\")
    ]. Use instead the socket-binding attribute to configure this
    broadcast-group.",
        "WFLYMSG0080: Could not migrate attribute group-port from
    resource [
        ("subsystem\" => \"messaging-activemq\"),
        ("server\" => \"default\"),
        ("broadcast-group\" => \"groupA\")
    ]. Use instead the socket-binding attribute to configure this
    broadcast-group."
    ]}
}
```

NOTA

A lista de **migrate** e **describe-migration** avisos para cada subsistema estão localizados no [Material de referência](#) no final deste guia.

- [Avisos de Operação da Migração do Subsistema Jacorb](#)
- Avisos para a operação de migração de subsistema de mensagem [Esta é uma tradução automática]
- Avisos para a operação de migração de subsistema de mensagem [Esta é uma tradução automática]

4. Revise o arquivo de configuração de servidor para verificar se extensão, subsistema e espaço de nomes foram atualizados e as configurações dos subsistemas existentes foram migradas para o JBoss EAP 7.

NOTA

Você deve repetir este processo para cada um dos **jacorb**, **messaging**, e **web** subsistemas usando os seguintes comandos.

```

/subsystem=jacorb:migrate
/subsystem=messaging:migrate
/subsystem=web:migrate
```

5. Remova o **cmp**, **jaxr**, e **threads** subsistemas e extensões da configuração do servidor. Ainda no prompt do CLI de gerenciamento, remova o obsoleto **cmp**, **jaxr**, e **threads** subsistemas executando os seguintes comandos.

```

/subsystem=cmp:remove
/extension=org.jboss.as.cmp:remove
/subsystem=jaxr:remove
/extension=org.jboss.as.jaxr:remove
/subsystem=threads:remove
/extension=org.jboss.as.threads:remove
```



IMPORTANTE

Você deve migrar o **messaging**, **jacorb**, e **web** subsistemas e remova o **cmp**, **jaxr**, e **threads** extensões e subsistemas antes que você possa reiniciar o servidor para operação normal. Se você precisar reiniciar o servidor antes de concluir esse processo, inclua o **--start-mode=admin-only** argumento na linha de comando do início do servidor. Isso permite que você continue com as alterações de configuração.

4.4. ALTERAÇÕES DE REGISTRO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

4.4.1. Alterações de prefixos dos registros de mensagens [Esta é uma tradução automática]

Os registros de mensagens são prefixados com o código do projeto para o subsistema que relata a mensagem. Os prefixos para todos os registros de mensagens foram alterados no JBoss EAP 7.

Para uma lista completa dos novos prefixos de código de projeto de mensagem de log usados no JBoss EAP 7, veja [Códigos de Projeto Utilizados no JBoss EAP](#) no JBoss EAP *Guia de desenvolvimento*.

4.4.2. Alterações do manipulador de console do criador de logs raiz [Esta é uma tradução automática]

O criador de logs raiz do JBoss EAP 7.0 incluiu um manipulador de logs do console para todos os perfis de servidor de domínio e para todos os perfis independentes padrão, exceto o **standalone-full-ha** perfil. O criador de logs raiz do JBoss EAP 7.1 não inclui mais um manipulador de logs do console para os perfis de domínio gerenciados. O controlador host e o controlador de processo registram no console por padrão. Para obter a mesma funcionalidade que foi fornecida no JBoss EAP 7.0, veja [Configurar um manipulador de log do console](#) no *Guia de configuração* para o JBoss EAP.

4.5. ALTERAÇÕES DE CONFIGURAÇÕES DO SERVIDOR WEB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

4.5.1. Substituição do subsistema web com Undertow [Esta é uma tradução automática]

Undertow substitui o JBoss Web como o servidor web no JBoss EAP 7. Isso significa que o legado **web** A configuração do subsistema deve ser migrada para o novo JBoss EAP 7 **undertow** configuração do subsistema.

- o **urn:jboss:domain:web:2.2** o namespace de configuração do subsistema no arquivo de configuração do servidor foi substituído pelo **urn:jboss:domain:undertow:4.0** namespace.
- o **org.jboss.as.web** módulo de extensão, localizado em **EAP_HOME/modules/system/layers/base/**, foi substituído pelo **org.wildfly.extension.undertow** módulo de extensão.

Você pode usar o CLI de gerenciamento **migrate** operação para migrar o **web** subsistema para **undertow** no arquivo de configuração do servidor. No entanto, esteja ciente de que esta operação não é capaz de migrar todas as configurações do subsistema Web do JBoss. Se você vir entradas de "aviso

de migração", deverá executar comandos CLI de gerenciamento adicionais para migrar essas configurações para Undertow. Para mais informações sobre o CLI de gerenciamento **migrate** operação, consulte [Operação de Migração do CLI de Gerenciamento](#).

O seguinte é um exemplo do padrão **web** configuração do subsistema no JBoss EAP 6.

```
<subsystem xmlns="urn:jboss:domain:web:2.2" default-virtual-
server="default-host" native="false">
  <connector name="http" protocol="HTTP/1.1" scheme="http" socket-
binding="http"/>
  <virtual-server name="default-host" enable-welcome-root="true">
    <alias name="localhost"/>
    <alias name="example.com"/>
  </virtual-server>
</subsystem>
```

O seguinte é um exemplo do padrão **undertow** configuração do subsistema no JBoss EAP 7.

```
<subsystem xmlns="urn:jboss:domain:undertow:4.0">
  <buffer-cache name="default"/>
  <server name="default-server">
    <http-listener name="default" socket-binding="http" redirect-
socket="https"/>
    <host name="default-host" alias="localhost">
      <location name="/" handler="welcome-content"/>
      <filter-ref name="server-header"/>
      <filter-ref name="x-powered-by-header"/>
    </host>
  </server>
  <servlet-container name="default">
    <jsp-config/>
    <websockets/>
  </servlet-container>
  <handlers>
    <file name="welcome-content" path="${jboss.home.dir}/welcome-
content"/>
  </handlers>
  <filters>
    <response-header name="server-header" header-name="Server" header-
value="JBoss-EAP/7"/>
    <response-header name="x-powered-by-header" header-name="X-
Powered-By" header-value="Undertow/1"/>
  </filters>
</subsystem>
```

4.5.2. Migrar as condições de regravação do JBoss Web [Esta é uma tradução automática]

O CLI de gerenciamento **migrate** operação não é capaz de migrar automaticamente as condições de reescrita. Eles são relatados como "avisos de migração" e você deve migrá-los manualmente. Você pode criar a configuração equivalente no JBoss EAP 7 usando [Atributos e manipuladores de predicados inferiores](#).

O seguinte é um exemplo de um **web** configuração do subsistema no JBoss EAP 6 que inclui **rewrite** configuração.

```
<subsystem xmlns="urn:jboss:domain:web:2.2" default-virtual-
server="default" native="false">
  <virtual-server name="default" enable-welcome-root="true">
    <alias name="localhost"/>
    <rewrite name="test" pattern="(.)/toberewritten/(.)"
substitution="$1/rewritten/$2" flags="NC"/>
    <rewrite name="test2" pattern="(.)" substitution="-" flags="F">
      <condition name="get" test="%{REQUEST_METHOD}"
pattern="GET"/>
      <condition name="andCond" test="%{REQUEST_URI}"
pattern=".*index.html" flags="NC"/>
    </rewrite>
  </virtual-server>
</subsystem>
```

Segue o [Operação de Migração do CLI de Gerenciamento](#) instruções para iniciar seu servidor e a CLI de gerenciamento, depois migrar **web** arquivo de configuração do subsistema usando o seguinte comando.

```
/subsystem=web:migrate
```

Os seguintes "avisos de migração" são relatados quando você executa o **migrate** operação na configuração acima.

```
/subsystem=web:migrate
{
  "outcome" => "success",
  "result" => {"migration-warnings" => [
    "WFLYWEB0002: Could not migrate resource {
\"pattern\" => \"(.*)\",
\"substitution\" => \"-\",
\"flags\" => \"F\",
\"operation\" => \"add\",
\"address\" => [
  (\"subsystem\" => \"web\"),
  (\"virtual-server\" => \"default-host\"),
  (\"rewrite\" => \"test2\")
]
}"),
    "WFLYWEB0002: Could not migrate resource {
\"test\" => \"%{REQUEST_METHOD}\",
\"pattern\" => \"GET\",
\"flags\" => undefined,
\"operation\" => \"add\",
\"address\" => [
  (\"subsystem\" => \"web\"),
  (\"virtual-server\" => \"default-host\"),
  (\"rewrite\" => \"test2\"),
  (\"condition\" => \"get\")
]
}"),
    "WFLYWEB0002: Could not migrate resource {
\"test\" => \"%{REQUEST_URI}\",
```



```

    \"pattern\" => \".*index.html\",
    \"flags\" => \"NC\",
    \"operation\" => \"add\",
    \"address\" => [
        (\"subsystem\" => \"web\"),
        (\"virtual-server\" => \"default-host\"),
        (\"rewrite\" => \"test2\"),
        (\"condition\" => \"andCond\")
    ]
}
}]
}

```

Revise o arquivo de configuração do servidor e você verá a seguinte configuração para o **undertow** subsistema.



NOTA

Configuração lado cliente e carregamento de classe [Esta é uma tradução automática]

```

<subsystem xmlns="urn:jboss:domain:undertow:4.0">
  <buffer-cache name="default"/>
  <server name="default-server">
    <http-listener name="http" socket-binding="http"/>
    <host name="default-host" alias="localhost, example.com">
      <location name="/" handler="welcome-content"/>
    </host>
  </server>
  <servlet-container name="default">
    <jsp-config/>
  </servlet-container>
  <handlers>
    <file name="welcome-content" path="${jboss.home.dir}/welcome-content"/>
  </handlers>
</subsystem>

```

Use a CLI de gerenciamento para criar o filtro para substituir a configuração de reconfiguração no **undertow** subsistema. Você deve ver "{" resultado "=>" sucesso "}" para cada comando.

```

# Create the filters
/subsystem=undertow/configuration=filter/expression-
filter="test1":add(expression="path('(.*)/toberewritten/(.*)') ->
rewrite('$1/rewritten/$2')")
/subsystem=undertow/configuration=filter/expression-
filter="test2":add(expression="method('GET') and path('.*index.html') ->
response-code(403)")

# Add the filters to the default server
/subsystem=undertow/server=default-server/host=default-host/filter-
ref="test1":add
/subsystem=undertow/server=default-server/host=default-host/filter-
ref="test2":add

```

Revise o arquivo de configuração do servidor atualizado. O subsistema Web JBoss está agora completamente migrado e configurado no **undertow** subsistema.

```
<subsystem xmlns="urn:jboss:domain:undertow:4.0">
  <buffer-cache name="default"/>
  <server name="default-server">
    <http-listener name="http" socket-binding="http"/>
    <host name="default-host" alias="localhost, example.com">
      <location name="/" handler="welcome-content"/>
      <filter-ref name="test1"/>
      <filter-ref name="test2"/>
    </host>
  </server>
  <servlet-container name="default">
    <jsp-config/>
  </servlet-container>
  <handlers>
    <file name="welcome-content" path="${jboss.home.dir}/welcome-content"/>
  </handlers>
  <filters>
    <expression-filter name="test1"
expression="path('(.*)/toberewritten/(.*)') ->
rewrite('$1/rewritten/$2')"/>
    <expression-filter name="test2" expression="method('GET') and
path('.*index.html') -> response-code(403)"/>
  </filters>
</subsystem>
```

Para obter mais informações sobre como configurar filtros e manipuladores usando a CLI de gerenciamento, consulte [Configurando o Servidor da Web](#) no JBoss EAP 7 *Guia de configuração*.

4.5.3. Migrar propriedades de sistemas JBoss Web [Esta é uma tradução automática]

Na versão anterior do JBoss EAP, as propriedades do sistema poderiam ser usadas para modificar o comportamento padrão do JBoss. Para obter informações sobre como configurar o mesmo comportamento no Undertow, consulte [Referência de Migração de Propriedades do Sistema JBoss Web](#)

4.5.4. Atualizar o padrão de cabeçalho do log de acesso [Esta é uma tradução automática]

Quando você migra do JBoss EAP 6.4 para o JBoss EAP 7.1, você pode achar que os logs de acesso não gravam mais os valores esperados "Referer" e "User-agent". Isto é porque o JBoss Web, que foi incluído no JBoss EAP 6.4, usou um padrão de `%{headername}i` no **access-log** para registrar um cabeçalho de entrada.

Exemplo: Access Log Format no JBoss EAP 6.4 [Esta é uma tradução automática]

```
<access-log pattern="%h %l %u %t &quot;%T sec&quot; &quot;%r&quot; %s %b
&quot;%{Referer}i&quot; &quot;%{User-agent}i&quot;"/>
```

Com a mudança para usar o Undertow no JBoss EAP 7.1, o padrão para um cabeçalho de entrada foi alterado para `%{i, headername}`.

Exemplo: Acesse o Formato Header no JBoss EAP 7.1 [Esta é uma tradução automática]

```
<access-log pattern="%h %l %u %t &quot;%T sec&quot; &quot;%r&quot; %s %b
&quot;%{i,Referer}&quot; &quot;%{i,User-Agent}&quot;"/>
```

4.5.5. Migrar válvulas globais [Esta é uma tradução automática]

As versões anteriores do JBoss EAP tinham suporte de válvulas. Válvulas são classes personalizadas inseridas na pipeline de processamento de solicitações para um aplicativo antes de filtros servlet para fazer alterações às solicitações ou executar processamento adicional.

- Válvula globais são inseridas na pipeline de processamento de solicitações de todos os aplicativos implantados e são configuradas no arquivo de configuração do servidor.
- Válvulas do autenticador autenticam as credenciais da solicitação.
- As válvulas de aplicação personalizadas foram criadas ao estender **org.apache.catalina.valves.ValveBase** classe e configurado no **<valve>** elemento do **jboss-web.xml** arquivo descritor. Essas válvulas devem ser migradas manualmente.

Esta seção descreve como migrar válvulas globais. A migração de válvulas personalizadas e autenticadoras é coberta [Migrar válvulas de aplicativos personalizados](#) seção deste guia.

Undertow, que substitui JBoss Web no JBoss EAP 7, não suporta válvulas globais; porém, você pode conseguir funcionalidade semelhante utilizando os manipuladores Undertow. Undertow inclui um número de manipuladores integrados que fornecem funcionalidade comum. Também fornece a habilidade de criar manipuladores personalizados, que podem ser utilizados para substituir funcionalidade de válvula personalizada.

Se o seu aplicativo utilizar válvulas, você deve substituí-las com o código de manipulador Undertow adequado para obter a mesma funcionalidade quando você migrar para o JBoss EAP 7.

Para obter mais informações sobre como configurar manipuladores, consulte [Configurando manipuladores](#) no JBoss EAP 7 *Guia de configuração*.

Para obter mais informações sobre como configurar filtros, consulte [Configurando Filtros](#) no JBoss EAP 7 *Guia de configuração*.

Migrar válvulas globais [Esta é uma tradução automática]

A tabela a seguir lista as válvulas que foram fornecidas pelo JBoss Web no release anterior do JBoss EAP e o manipulador integrado Undertow correspondente. As válvulas do JBoss Web estão localizadas no **org.apache.catalina.valves** pacote.

Tabela 4.2. Mapeamento das válvulas para manipuladores [Esta é uma tradução automática]

Válvula	Manipulador
AccessLogValve	io.undertow.server.handlers.accesslog.AccessLogHandler
CrawlerSessionManagerValve	io.undertow.servlet.handlers.CrawlerSessionManagerHandler
ExtendedAccessLogValve	io.undertow.server.handlers.accesslog.AccessLogHandler

Válvula	Manipulador
JDBCAccessLogValve	Veja o JDBCAccessLogValve Procedimento de Migração Manual abaixo para instruções.
RemoteAddrValve	io.undertow.server.handlers.IPAddressAccessControlHandler
RemoteHostValve	io.undertow.server.handlers.AccessControlListHandler
RemoteIpValve	io.undertow.server.handlers.ProxyPeerAddressHandler
RequestDumperValve	io.undertow.server.handlers.RequestDumpingHandler
RewriteValve	Vejo Migrar as condições do JBoss Web Rewrite para instruções de como migrar essas válvulas manualmente.
StuckThreadDetectionValve	io.undertow.server.handlers.StuckThreadDetectionHandler

Você pode usar o CLI de gerenciamento **migrate** operação para migrar automaticamente válvulas globais que atendam aos seguintes critérios:

- Elas estão limitadas às válvulas listadas na tabela anterior que não necessitam de processamento manual.
- Eles devem ser definidos no **web** subsistema do arquivo de configuração do servidor.

Para mais informações sobre o CLI de gerenciamento **migrate** operação, consulte [Operação de Migração do CLI de Gerenciamento](#).

Procedimento de migração manual JDBCAccessLogValve

o `org.apache.catalina.valves.JDBCAccessLogValve` válvula é uma exceção à regra e não pode ser migrada automaticamente para `io.undertow.server.handlers.JDBCLogHandler`. Siga as etapas abaixo para migrar a seguinte válvula de exemplo.

```
<valve name="jdbc" module="org.jboss.as.web" class-
name="org.apache.catalina.valves.JDBCAccessLogValve">
  <param param-name="driverName" param-value="com.mysql.jdbc.Driver" />
  <param param-name="connectionName" param-value="root" />
  <param param-name="connectionPassword" param-value="password" />
  <param param-name="connectionURL" param-
value="jdbc:mysql://localhost:3306/wildfly?
zeroDateTimeBehavior=convertToNull" />
  <param param-name="format" param-value="combined" />
</valve>
```

1. Crie um módulo de driver para o banco de dados que irá armazenar as entradas de registro.
2. Configure a fonte de dados para o banco de dados e adicione o driver à lista de drivers disponíveis no **datasources** subsistema.

```
<datasources>
  <datasource jndi-name="java:jboss/datasources/accessLogDS" pool-
```

```

name="accessLogDS" enabled="true" use-java-context="true">
  <connection-url>jdbc:mysql://localhost:3306/wildfly?
zeroDateTimeBehavior=convertToNull</connection-url>
  <driver>mysql</driver>
  <security>
    <user-name>root</user-name>
    <password>Password1!</password>
  </security>
</datasource>
...
<drivers>
  <driver name="mysql" module="com.mysql">
    <driver-class>com.mysql.jdbc.Driver</driver-class>
  </driver>
...
</drivers>
</datasources>

```

3. Configurar um **expression-filter** no **undertow** subsistema com a seguinte expressão: **jdbc-access-log(datasource=DATASOURCE_JNDI_NAME)**.

```

<filters>
  <expression-filter name="jdbc-access" expression="jdbc-access-
log(datasource='java:jboss/datasources/accessLogDS')" />
  ...
</filters>

```

4.5.6. Alterações no comportamento do conjunto de cookies [Esta é uma tradução automática]

Especificações anteriores para **Set-Cookie** A sintaxe do cabeçalho de resposta HTTP, por exemplo, RFC2109 e RFC2965, permitia espaço em branco e outros caracteres separadores no valor do cookie quando o valor do cookie era citado. O JBoss Web no JBoss EAP 6.4 estava de acordo com as especificações anteriores e automaticamente citava um valor de cookie quando continha quaisquer caracteres separadores.

o [RFC6265](#) especificação para **Set-Cookie** A sintaxe do cabeçalho de resposta HTTP indica que os valores de cookie no **Set-Cookie** O cabeçalho de resposta deve estar em conformidade com as restrições gramaticais específicas. Por exemplo, eles devem ser caracteres US-ASCII, mas não podem incluir CTRLs (controles), espaço em branco, aspas duplas, vírgulas, ponto e vírgula ou caracteres de barra invertida.

No JBoss EAP 7.0, antes do patch cumulativo [Red Hat JBoss Enterprise Application Platform 7.0 atualização 08](#), Undertow não restringe esses caracteres inválidos e não cita cookies que continham os caracteres excluídos. Se você aplicar esse patch cumulativo ou um patch cumulativo mais recente, poderá ativar a validação de cookie compatível com RFC6265 definindo **io.undertow.cookie.DEFAULT_ENABLE_RFC6265_COOKIE_VALIDATION** propriedade do sistema para **true**.

No JBoss EAP 7.1, por padrão, Undertow não habilita a validação de cookie compatível com RFC6265. Ele cita cookies que contêm os caracteres excluídos. No JBoss EAP 7.1, você não pode usar o **io.undertow.cookie.DEFAULT_ENABLE_RFC6265_COOKIE_VALIDATION** propriedade do sistema para ativar a validação de cookie compatível com RFC6265. Em vez disso, você ativa a validação de

cookie compatível com RFC6265 para um ouvinte HTTP, HTTPS ou AJP configurando o **rfc6265-cookie-validation** atributo de ouvinte para **true**. O valor padrão para este atributo é **false**. O exemplo a seguir ativa a validação de cookie compatível com RFC6265 para o ouvinte HTTP.

```
/subsystem=undertow/server=default-server/http-listener=default:write-attribute(name=rfc6265-cookie-validation,value=true)
```

4.5.7. Alterações no comportamento de chamadas de método HTTP [Esta é uma tradução automática]

O JBoss EAP 6.4, que incluía o JBoss Web como servidor web, permitia **TRACE** chamadas de método por padrão.

Undertow, que substitui o JBoss Web como o servidor web no JBoss EAP 7, não permite HTTP **TRACE** chamadas de método por padrão. Esta configuração é configurada usando o **disallowed-methods** atributo do **http-listener** elemento no **undertow** subsistema. Isso pode ser confirmado pela análise da saída dos seguintes **read-resource** comando. Observe que o valor para o **disallowed-methods** atributo é **["TRACE"]**.

```
/subsystem=undertow/server=default-server/http-listener=default:read-resource
{
  "outcome" => "success",
  "result" => {
    "allow-encoded-slash" => false,
    "allow-equals-in-cookie-value" => false,
    "always-set-keep-alive" => true,
    "buffer-pipelined-data" => false,
    "buffer-pool" => "default",
    "certificate-forwarding" => false,
    "decode-url" => true,
    "disallowed-methods" => ["TRACE"],
    ...
  }
}
```

Para ativar o HTTP **TRACE** chamadas de método no JBoss EAP 7 e posterior, você deve remover a entrada "TRACE" do **disallowed-methods** lista de atributos executando o seguinte comando.

```
/subsystem=undertow/server=default-server/http-listener=default:list-remove(name=disallowed-methods,value="TRACE")
```

Quando você executa o **read-resource** comando novamente, você vai notar o **TRACE** chamada de método não está mais na lista de métodos não permitidos.

```
/subsystem=undertow/server=default-server/http-listener=default:read-resource
{
  "outcome" => "success",
  "result" => {
    "allow-encoded-slash" => false,
    "allow-equals-in-cookie-value" => false,
    "always-set-keep-alive" => true,
```

```

        "buffer-pipelined-data" => false,
        "buffer-pool" => "default",
        "certificate-forwarding" => false,
        "decode-url" => true,
        "disallowed-methods" => [],
        ...
    }
}

```

Para obter mais informações sobre o comportamento padrão dos métodos HTTP, consulte [Comportamento Padrão de Métodos HTTP](#) no JBoss EAP *Guia de configuração*.

4.5.8. Mudanças no Comportamento do Módulo da Web Padrão no JBoss EAP 7.1 [Esta é uma tradução automática]

No JBoss EAP 7.0, o contexto raiz de uma aplicação web foi desabilitado por padrão em `mod_cluster`.

Este não é mais o caso no JBoss EAP 7.1. Isso pode ter consequências inesperadas se você estiver esperando que o contexto raiz seja desativado. Por exemplo, as solicitações podem ser desviadas para nós indesejados ou um aplicativo particular que não deve ser exposto pode ser acessado inadvertidamente por meio de um proxy público. As localizações inferiores também são registradas com o balanceador de carga `mod_cluster` automaticamente, a menos que sejam explicitamente excluídas.

Use o seguinte comando CLI de gerenciamento para excluir ROOT do **modcluster** configuração do subsistema.

```

/subsystem=modcluster/mod-cluster-config=configuration:write-
attribute(name=excluded-contexts,value=ROOT)

```

Use o seguinte comando da CLI de gerenciamento para desativar o aplicativo da web de boas-vindas padrão.

```

/subsystem=undertow/server=default-server/host=default-
host/location=\/:remove
/subsystem=undertow/configuration=handler/file=welcome-content:remove
reload

```

Para obter mais informações sobre como configurar o aplicativo da Web de boas-vindas padrão, consulte [Configurar o aplicativo da Web de boas-vindas padrão](#) no *Guia de desenvolvimento* para o JBoss EAP.

4.6. ALTERAÇÕES DE CONFIGURAÇÕES DO SERVIDOR JGROUPS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

4.6.1. JGroups é pre-definido para uma interface de rede privada [Esta é uma tradução automática]

Na configuração padrão do JBoss EAP 6, os JGroups usaram o **public** interface definida no **<interfaces>** seção do arquivo de configuração do servidor.

Como é uma prática recomendada usar uma interface de rede dedicada, o JGroups agora usa o novo padrão **private** interface que é definida no **<interfaces>** seção do arquivo de configuração do servidor no JBoss EAP 7.

4.6.2. Alterações de Canais JGroups [Esta é uma tradução automática]

O JGroups fornece suporte de comunicação de grupo para serviços de alta disponibilidade na forma de canais do JGroups. O JBoss EAP 7 apresenta `<channel>` elementos para o `jgroups` subsistema no arquivo de configuração do servidor. Você pode adicionar, remover ou alterar a configuração de canais do JGroups usando a CLI de gerenciamento.

Para obter mais informações sobre como configurar JGroups, consulte [Comunicação de Cluster com JGroups](#) no JBoss EAP *Guia de configuração*.

4.7. ALTERAÇÕES DE CONFIGURAÇÃO DE SERVIDOR INFINISPAN [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

4.7.1. Alterações de configurações de caches por padrão de Infinispan [Esta é uma tradução automática]

No JBoss EAP 6, os caches clusterizados padrão para replicação de sessão da web e replicação EJB foram replicados **ASYNC** caches. Isso mudou no JBoss EAP 7. Os caches clusterizados padrão agora são distribuídos **ASYNC** caches. Os caches replicados não são mais configurados por padrão. Veja [Configurar o modo de cache](#) no JBoss EAP *Guia de configuração* para obter informações sobre como adicionar um cache replicado e torná-lo o padrão.

Isso só afeta você quando você usa a nova configuração padrão do JBoss EAP 7. Se você migrar a configuração do JBoss EAP 6, a configuração do **infinispan** subsistema será preservado.

4.7.2. Alterações de estratégia de cachê Infinispan [Esta é uma tradução automática]

O comportamento de **ASYNC** A estratégia de cache mudou no JBoss EAP 7.

No JBoss EAP 6, **ASYNC** Leituras de cache eram livres de bloqueio. Embora eles nunca fossem bloqueados, eles eram propensos a leituras sujas de dados obsoletos, por exemplo, em failover. Isso ocorre porque permitiria solicitações subsequentes para o mesmo usuário iniciar antes que a solicitação anterior fosse concluída. Essa permissividade não é aceitável ao usar o modo distribuído, já que as alterações da topologia de cluster podem afetar a afinidade da sessão e facilmente resultar em dados obsoletos.

No JBoss EAP 7, **ASYNC** leituras de cache exigem bloqueios. Como agora bloqueiam novas solicitações do mesmo usuário até que a replicação anterior termine, leituras sujas são evitadas.

4.7.3. Configurando o Cache do Bean de Sessão de Estado Customizado para Passivação [Esta é uma tradução automática]

Esteja ciente das seguintes restrições ao configurar um cache SFSB personalizado para passivação no JBoss EAP 7.1.

- o **idle-timeout** atributo, que é configurado no **infinispan passivation-store** do **ejb3** subsistema, está obsoleto no JBoss EAP 7.1. O JBoss EAP 6.4 suportava passivação rápida, passivando de acordo com o **idle-timeout** valor. O JBoss EAP 7.1 suporta passivação preguiçosa, passivando quando o **max-size** limiar é alcançado.
- No JBoss EAP 7.1, o nome do cluster usado pelo cliente EJB é determinado pelo nome real do cluster do canal, conforme configurado no **jgroups** subsistema.

- O JBoss EAP 7.1 ainda permite que você defina **max-size** atributo para controlar o limiar de passivação.
- Você não deve configurar o despejo ou a expiração em sua configuração de cache do EJB.
 - Você deve configurar o despejo usando o **max-size** atributo do **passivation-store** no **ejb3** subsistema.
 - Você deve configurar a expiração usando o **@StatefulTimeout** anotação no código fonte do SFSB Java ou especificando um **stateful-timeout** valor no **ejb-jar.xml** Arquivo.

4.7.4. Alterações no Transporte do Contêiner de Cache Infinispan [Esta é uma tradução automática]

Uma mudança de comportamento entre o JBoss EAP 7.0 e o JBoss EAP 7.1 requer que quaisquer atualizações no protocolo de transporte do contêiner de cache sejam feitas no modo batch ou usando um cabeçalho especial. Essa mudança de comportamento também afeta quaisquer ferramentas usadas para gerenciar o servidor do JBoss EAP.

A seguir, um exemplo dos comandos CLI de gerenciamento usados para configurar o protocolo de transporte do contêiner de cache no JBoss EAP 7.0.

```
/subsystem=infinispan/cache-container=my:add()
/subsystem=infinispan/cache-container=my/transport=jgroups:add()
/subsystem=infinispan/cache-container=my/invalidation-
cache=mycache:add(mode=SYNC)
```

A seguir, um exemplo dos comandos CLI de gerenciamento necessários para executar a mesma configuração no JBoss EAP 7.1. Note que os comandos são executados em modo batch.

```
batch
/subsystem=infinispan/cache-container=my:add()
/subsystem=infinispan/cache-container=my/transport=jgroups:add()
/subsystem=infinispan/cache-container=my/invalidation-
cache=mycache:add(mode=SYNC)
run-batch
```

Se preferir não usar o modo batch, você pode especificar o cabeçalho da operação **allow-resource-service-restart=true** ao definir o transporte. Esteja ciente de que isso reinicia o serviço para que as operações entrem em vigor, e alguns serviços podem parar de funcionar até que o serviço seja reiniciado.

Se você usar scripts para atualizar o protocolo de transporte do contêiner de cache, não deixe de revisá-los e adicionar o modo em lote.

4.8. ALTERAÇÕES DE CONFIGURAÇÕES DE SERVIDOR EJB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Não há **migrate** operação para o **ejb3** subsistema, por isso, se você usar o CLI de gerenciamento **migrate** operações para atualizar suas outras configurações existentes do JBoss EAP 6.4, esteja ciente de que **ejb3** a configuração do subsistema não é migrada. Porque a configuração do **ejb3** subsistema é um pouco diferente no JBoss EAP 7.1 do que no JBoss EAP 6.4, você pode ver exceções no log do servidor quando você implementa seus aplicativos EJB.



IMPORTANTE

Se você usar a Ferramenta de Migração do JBoss Server para atualizar a configuração do servidor, **ejb3** O subsistema deve ser configurado corretamente e você não deve ver nenhum problema ao implementar seus aplicativos EJB. Para obter informações sobre como configurar e executar a ferramenta, consulte [Usando a Ferramenta de Migração do JBoss Server](#).

DuplicateServiceException no log do servidor [Esta é uma tradução automática]

Os seguintes **DuplicateServiceException** é causado por alterações de cache no JBoss EAP 7.

DuplicateServiceException no log do servidor [Esta é uma tradução automática]

```
ERROR [org.jboss.msc.service.fail] (MSC service thread 1-3) MSC000001:
Failed to start service jboss.deployment.unit."mdb-1.0-
SNAPSHOT.jar".cache-dependencies-installer:
org.jboss.msc.service.StartException in service
jboss.deployment.unit."mdb-1.0-SNAPSHOT.jar".cache-dependencies-installer:
Failed to start service
...
Caused by: org.jboss.msc.service.DuplicateServiceException: Service
jboss.infinispan.ejb."mdb-1.0-SNAPSHOT.jar".config is already registered
```

Você deve reconfigurar o cache para resolver este erro.

1. Siga as instruções para [Inicie o servidor e a CLI de gerenciamento](#).
2. Emita os seguintes comandos para reconfigurar o armazenamento em cache no **ejb3** subsistema.

```
/subsystem=ejb3/file-passivation-store=file:remove
/subsystem=ejb3/cluster-passivation-store=infinispan:remove
/subsystem=ejb3/passivation-store=infinispan:add(cache-
container=ejb, max-size=10000)

/subsystem=ejb3/cache=passivating:remove
/subsystem=ejb3/cache=clustered:remove
/subsystem=ejb3/cache=distributable:add(passivation-
store=infinispan, aliases=[passivating, clustered])
```

4.9. ALTERAÇÕES DE CONFIGURAÇÃO DO SERVIDOR DE MENSAGENS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

No JBoss EAP 7, Active MQ Artemis substitui HornetQ como o fornecedor de suporte JMS. Esta seção descreve como migrar a configuração e dados de mensagens relacionados.

4.9.1. Alterações de configuração do servidor de subsistema de mensagens [Esta é uma tradução automática]

o **org.jboss.as.messaging** extensão de módulo, localizada em **EAP_HOME/modules/system/layers/base/**, foi substituído pelo **org.wildfly.extension.messaging-activemq** módulo de extensão.

o `urn:jboss:domain:messaging:3.0` o namespace de configuração do subsistema foi substituído pelo `urn:jboss:domain:messaging-activemq:2.0` namespace.

Mudanças no gerenciamento de JMX [Esta é uma tradução automática]

Na maioria dos casos, houve um empenho para manter o quanto mais possível a similaridade dos nomes de elementos e de atributos daqueles utilizados nas versões prévias. A seguinte tabela lista algumas das alterações.

Tabela 4.3. Mapeamento dos atributos de mensagens [Esta é uma tradução automática]

Nome HornetQ	Nome ActiveMQ
hornetq-server	servidor
Tipo de hornetq-server	serverType
conectores	conector
discovery-group-name	discovery-group

As operações de gerenciamento invocadas no novo `messaging-activemq` subsistema mudou de `/subsystem=messaging/hornetq-server=` para `/subsystem=messaging-activemq/server=`.

Você pode migrar um JBoss EAP 6 existente `messaging` configuração do subsistema para o `messaging-activemq` subsistema em um servidor JBoss EAP 7 invocando sua `migrate` Operação.

```
/subsystem=messaging:migrate
```

Antes de executar o `migrate` operação, você pode invocar o `describe-migration` operação para revisar a lista de operações de gerenciamento que serão executadas para migrar do JBoss EAP 6 existente `messaging` configuração do subsistema para o `messaging-activemq` subsistema no servidor do JBoss EAP 7.

```
/subsystem=messaging:describe-migration
```

o `migrate` e `describe-migration` operações também exibem uma lista de `migration-warnings` para recursos ou atributos que não podem ser migrados automaticamente.

Avisos para a operação de migração de subsistema de mensagem [Esta é uma tradução automática]

o `describe-migration` e `migrate` operações para o `messaging` subsistema fornece um argumento de configuração adicional. Se você deseja configurar o sistema de mensagens para permitir que clientes legados do JBoss EAP 6 se conectem ao servidor do JBoss EAP 7, você pode adicionar o sistema de mensagens booleano. `add-legacy-entries` argumento para o `describe-migration` ou `migrate` operação da seguinte forma.

```
/subsystem=messaging:describe-migration(add-legacy-entries=true)
/subsystem=messaging:migrate(add-legacy-entries=true)
```

Se o argumento booleano **add-legacy-entries** está configurado para **true**, a **messaging-activemq** subsistema cria o **legacy-connection-factory** recurso e adiciona **legacy-entries** ao **jms-queue** e **jms-topic** Recursos.

Se o argumento booleano **add-legacy-entries** está configurado para **false**, nenhum recurso legado é criado no **messaging-activemq** subsistemas e clientes JMS legados não poderão se comunicar com os servidores do JBoss EAP 7. Este é o valor padrão.

Para obter mais informações sobre compatibilidade com versões anteriores e futuras, consulte [Compatibilidade para trás e para frente](#) dentro *Configurando o Messaging* para o JBoss EAP.

Para mais informações sobre o CLI de gerenciamento **migrate** e **describe-migration** operações, consulte [Operação de Migração do CLI de Gerenciamento](#).

Alteração de comportamento de atributo **forward-when-no-consumers**

O comportamento do **forward-when-no-consumers** atributo foi alterado no JBoss EAP 7.

No JBoss EAP 6, quando **forward-when-no-consumers** foi definido para **false** e não havia consumidores em um cluster, as mensagens foram redistribuídas para todos os nós em um cluster.

Este comportamento mudou no JBoss EAP 7. Quando **forward-when-no-consumers** está configurado para **false** e não há consumidores em um cluster, as mensagens não são redistribuídas. Em vez disso, eles são mantidos no nó original para o qual foram enviados.

Alteração na diretiva de balanceamento de carga de cluster padrão

A política de balanceamento de carga de cluster padrão foi alterada no JBoss EAP 7.

No JBoss EAP 6, a política de balanceamento de carga de cluster padrão era similar a **STRICT**, que é como definir o legado **forward-when-no-consumers** parâmetro para **true**. No JBoss EAP 7, o padrão é agora **ON_DEMAND**, que é como definir o legado **forward-when-no-consumers** parâmetro para **false**. Para mais informações sobre essas configurações, consulte [Atributos de Conexão de Cluster](#) dentro *Configurando o Messaging* para o JBoss EAP.

Alterações de configuração do servidor de subsistema de mensagens [Esta é uma tradução automática]

A configuração XML mudou significativamente com o novo **messaging-activemq** subsistema, e agora fornece um esquema XML mais consistente com outros subsistemas do JBoss EAP.

É altamente recomendado que você não tente modificar o JBoss EAP **messaging** configuração XML do subsistema para se adequar ao novo **messaging-activemq** subsistema. Em vez disso, invoque o subsistema herdado **migrate** Operação. Esta operação irá escrever a configuração XML do novo **messaging-activemq** subsistema como parte de sua execução.

4.9.2. Migrar dados de mensagem [Esta é uma tradução automática]

Você pode usar uma das seguintes abordagens para migrar dados do sistema de mensagens de um release anterior para o release atual do JBoss EAP.

- Para sistemas de mensagens baseados em arquivos, você pode migrar dados de mensagens do JBoss EAP 6.4 ou do JBoss EAP 7.0 para o JBoss EAP 7.1 [usando o método de exportação e importação](#). Com esse método, você exporta os dados do sistema de mensagens do release anterior e os importa usando a CLI de gerenciamento **import-journal** Operação. Esteja ciente de que você pode usar essa abordagem apenas para sistemas de mensagens baseados em arquivos.

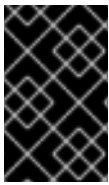
- Você pode migrar dados de mensagens do JBoss EAP 6.4 para o JBoss EAP 7.1 [configurando uma ponte JMS](#). Você pode usar essa abordagem para sistemas de mensagens baseados em arquivos e JDBC.

Devido à mudança do HornetQ para o ActiveMQ Artemis como provedor de suporte JMS, tanto o formato quanto a localização dos dados do sistema de mensagens foram alterados no JBoss EAP 7.0 e posterior. Veja [Mapeando nomes de pastas de mensagens](#) para obter detalhes sobre as alterações nos nomes e locais da pasta de dados do sistema de mensagens entre as versões 6.4 e 7.x.

4.9.2.1. Migrar dados de mensagens usando exportação e importação [Esta é uma tradução automática]

Usando essa abordagem, você exporta os dados do sistema de mensagens de um release anterior para um arquivo XML e, em seguida, importa esse arquivo usando o arquivo. **import-journal** Operação.

1. Exporte os dados do sistema de mensagens para um arquivo XML.
 - [Exportar dados de mensagens do JBoss EAP 6.4](#).
 - [Exportar dados de mensagens do JBoss EAP 7.0](#).
2. [Importe os dados de mensagens formatados em XML](#).



IMPORTANTE

Não é possível usar o método de exportação e importação para mover dados do sistema de mensagens entre sistemas que usam um diário baseado em JDBC para armazenamento.

Exemplo: Access Log Format no JBoss EAP 6.4 [Esta é uma tradução automática]

Devido à mudança do HornetQ para o ActiveMQ Artemis como provedor de suporte JMS, tanto o formato quanto a localização dos dados do sistema de mensagens foram alterados no JBoss EAP 7.0 e posterior.

Para exportar dados de mensagens do JBoss EAP 6.4, você deve usar o HornetQ **exporter** utilidade. O HornetQ **exporter** utilitário gera e exporta os dados do sistema de mensagens do JBoss EAP 6.4 para um arquivo XML. Este comando requer que você especifique os caminhos para os JARs do HornetQ que são enviados com o JBoss EAP 6.4, passe os caminhos para **messagingbindings/**, **messagingjournal/**, **messagingpaging/**, e **messaginglargemessages/** pastas do release anterior como argumentos e especifique um arquivo de saída no qual gravar os dados XML exportados.

A seguir, a sintaxe exigida pelo HornetQ **exporter** utilidade.

```
$ java -jar -mp MODULE_PATH org.hornetq.exporter
MESSAGING_BINDINGS_DIRECTORY MESSAGING_JOURNAL_DIRECTORY
MESSAGING_PAGING_DIRECTORY MESSAGING_LARGE_MESSAGES_DIRECTORY >
OUTPUT_DATA.xml
```

Crie um módulo personalizado para garantir que as versões corretas dos JARs do HornetQ, incluindo quaisquer JARs instalados com correções ou atualizações, sejam carregados e disponibilizados para o **exporter** utilidade. Usando seu editor favorito, crie um novo **module.xml** arquivo no **EAP6_HOME/modules/org/hornetq/exporter/main/** diretório e copie o seguinte conteúdo:

```
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="urn:jboss:module:1.1" name="org.hornetq.exporter">
```

```

<main-class
name="org.hornetq.jms.persistence.impl.journal.XmlDataExporter"/>
<properties>
  <property name="jboss.api" value="deprecated"/>
</properties>
<dependencies>
  <module name="org.hornetq"/>
</dependencies>
</module>

```



NOTA

O módulo personalizado é criado no **modules/** diretório, não o **modules/system/layers/base/** diretório.

Siga os passos abaixo para exportar os dados.

1. Exemplo: Nome do Driver do JBoss EAP 6 [Esta é uma tradução automática]
2. Crie um módulo personalizado conforme descrito acima.
3. Execute o seguinte comando para exportar os dados.

```

$ java -jar jboss-modules.jar -mp modules/ org.hornetq.exporter
standalone/data/messagingbindings/ standalone/data/messagingjournal/
standalone/data/messagingpaging
standalone/data/messaginglargemessages/ >
OUTPUT_DIRECTORY/OldMessagingData.xml

```

4. Certifique-se de que não há erros ou mensagens de aviso no registro na conclusão do comando.
5. Utilize as ferramentas disponíveis para seu sistema operacional para validar o XML no arquivo de saída gerado.

Mudanças de Configuração de Mensagens no JBoss EAP 7.1 [Esta é uma tradução automática]

Siga estas etapas para exportar dados de mensagens do JBoss EAP 7.0.

1. Abra um terminal, navegue até o diretório de instalação do JBoss EAP 7.0 e inicie o servidor em **admin-only** modo.

```

$ EAP_HOME/bin/standalone.sh -c standalone-full.xml --start-
mode=admin-only

```

2. Abra um novo terminal, navegue até o diretório de instalação do JBoss EAP 7.0 e conecte-se à CLI de gerenciamento.

```

$ EAP_HOME/bin/jboss-cli.sh --connect

```

3. Use o seguinte comando da CLI de gerenciamento para exportar os dados do diário de mensagens.

```

/subsystem=messaging-activemq/server=default:export-journal()

```

4. Certifique-se de que não há erros ou mensagens de aviso no registro na conclusão do comando.
5. Utilize as ferramentas disponíveis para seu sistema operacional para validar o XML no arquivo de saída gerado.

Migrar dados de mensagem [Esta é uma tradução automática]

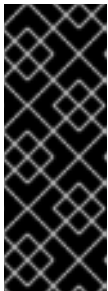
Você então importa o arquivo XML para o JBoss EAP 7.0 ou posterior usando o **import-journal** operação da seguinte forma.



IMPORTANTE

Se o servidor de destino já tiver executado algumas tarefas de mensagens, faça backup de suas pastas de mensagens antes de iniciar a tarefa. **import-journal** operação para evitar a perda de dados no caso de uma falha na importação. Veja [Fazendo backup dos dados da pasta de mensagens](#) Para maiores informações.

1. Se você está migrando seu servidor do JBoss EAP 6.4 para o 7.1, certifique-se de ter completado a migração da configuração do servidor antes de começar usando o [operação de migração da CLI de gerenciamento](#) ou executando a Ferramenta de Migração do JBoss Server. Para obter informações sobre como configurar e executar a ferramenta, consulte [Usando a Ferramenta de Migração do JBoss Server](#).
2. Inicie o servidor do JBoss EAP 7.x no modo normal com *não* Clientes JMS conectados.



IMPORTANTE

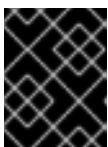
É importante que você inicie o servidor sem nenhum cliente JMS conectado. Isso é porque o **import-journal** operação se comporta como um produtor JMS. As mensagens estão imediatamente disponíveis quando a operação está em andamento. Se esta operação falhar no meio da importação e os clientes JMS estiverem conectados, não será possível recuperar porque os clientes JMS já podem ter consumido algumas das mensagens.

3. Abra um novo terminal, navegue até o diretório de instalação do JBoss EAP 7.xe conecte-se à CLI de gerenciamento.

```
$ EAP_HOME/bin/jboss-cli.sh --connect
```

4. Use o seguinte comando da CLI de gerenciamento para importar os dados do sistema de mensagens.

```
/subsystem=messaging-activemq/server=default:import-journal(file=OUTPUT_DIRECTORY/OldMessagingData.xml)
```



IMPORTANTE

Faz *não* execute este comando mais de uma vez, pois isso resultará em mensagens duplicadas!



ATENÇÃO

Se você estiver usando o JBoss EAP 7.0, você deve [Red Hat JBoss Enterprise Application Platform 7,0 atualização 05](#) ou um novo patch cumulativo para a sua instalação do JBoss EAP, a fim de evitar um problema conhecido ao ler mensagens grandes. Para mais informações, veja [JBEAP-4407 - O consumidor trava com IndexOutOfBoundsException ao ler mensagens grandes de um registro importado](#). Este problema não afeta o JBoss EAP 7.1 ou posterior.

Recuperando-se de uma falha de dados de mensagens de importação

Se o `import-journal` operação falhar, você pode tentar recuperar usando as etapas a seguir.

1. Encerre o servidor do JBoss EAP 7.x.
2. Exclua todas as pastas do diário de mensagens. Veja [Fazendo backup dos dados da pasta de mensagens](#) para os comandos da CLI de gerenciamento para determinar o local correto do diretório para as pastas do diário de mensagens.
3. Se você fez backup dos dados do sistema de mensagens do servidor de destino antes da importação, copie as pastas do sistema de mensagens do local de backup para o diretório do diário de mensagens determinado na etapa anterior.
4. Repita os passos para [importar os dados de mensagens formatados em XML](#).

4.9.2.2. Migrar dados de mensagem usando uma ponte JMS [Esta é uma tradução automática]

Usando essa abordagem, você configura e implementa uma ponte JMS no servidor do JBoss EAP 7.x. A ponte JMS move as mensagens da fila JBoss EAP 6.4 HornetQ para a fila do JBoss EAP 7.x ActiveMQ Artemis.

Uma ponte JMS consome mensagens a partir de uma fila JMS fonte ou tópico e as envia a uma fila JMS destino ou tópico, que está normalmente em um servidor diferente. Isto pode ser usado como ponte para as mensagens entre quaisquer servidores JMS, contanto que elas sejam compatíveis com o JMS 1.1. Os recursos JMS de destino e fonte são pesquisados usando o JNDI e as classes de cliente para a pesquisa JNDI devem ser empacotadas em um módulo. O nome do módulo é, então, declarado na configuração da ponte JMS.

Esta seção descreve como configurar os servidores e implantar uma ponte JMS para mover os dados do sistema de mensagens do JBoss EAP 6.4 para o JBoss EAP 7.x.

1. [Configure o servidor de origem do JBoss EAP 6.4.](#)
2. [Configure o servidor de destino do JBoss EAP 7.x.](#)
3. [Migre os dados do sistema de mensagens.](#)

Configurar o servidor do JBoss EAP 6.4

1. Exemplo: Nome do Driver do JBoss EAP 6 [Esta é uma tradução automática]

2. Fazendo o back up do diário HornetQ e arquivos de configuração.
 - Por padrão, o diário HornetQ está localizado no **EAP6_HOME/standalone/data/** diretório.
 - Veja [Mapeando nomes de pastas de mensagens](#) para locais de pastas de mensagens padrão para cada versão.
3. Certifique-se de que **InQueue** A fila JMS contendo as mensagens JMS é definida no servidor do JBoss EAP 6.4.
4. Certifique-se de que **messaging** configuração do subsistema contém uma entrada para o **RemoteConnectionFactory** semelhante ao seguinte.

```
<connection-factory name="RemoteConnectionFactory">
  <entries>
    <entry
name="java:jboss/exported/jms/RemoteConnectionFactory"/>
    </entries>
    ...
  </connection-factory>
```

Se não contiver a entrada, crie uma usando o seguinte comando da CLI de gerenciamento:

```
/subsystem=messaging/hornetq-server=default/connection-
factory=RemoteConnectionFactory:add(factory-type=XA_GENERIC,
connector=[netty], entries=
[java:jboss/exported/jms/RemoteConnectionFactory],ha=true,block-on-
acknowledge=true,retry-interval=1000,retry-interval-
multiplier=1.0,reconnect-attempts=-1)
```

Configurar o servidor do JBoss EAP 7.x de destino

1. A configuração da ponte JMS precisa do **org.hornetq** módulo para se conectar ao servidor HornetQ na versão anterior. Este módulo e suas dependências diretas não estão presentes no JBoss EAP 7.x, então você deve copiar os seguintes módulos da versão anterior.
 - Copie o **org.hornetq** módulo no JBoss EAP 7.x **EAP_HOME/modules/org/** diretório.
 - Se você não aplicou patches neste módulo, copie esta pasta do servidor do JBoss EAP 6.4: **EAP6_HOME/modules/system/layers/base/org/hornetq/**
 - Se você aplicou patches neste módulo, copie esta pasta do servidor do JBoss EAP 6.4: **EAP6_HOME/modules/system/layers/base/.overlays/layer-base-jboss-eap-6.4.x.CP/org/hornetq/**
 - Remova o **<resource-root>** para o HornetQ **lib** caminho do JBoss EAP 7.x **EAP_HOME/modules/org/hornetq/main/module.xml** Arquivo.
 - Se você não aplicou patches ao JBoss EAP 6.4 **org.hornetq** módulo, remova a seguinte linha do arquivo:

```
<resource-root path="lib"/>
```

- Se você aplicou patches ao JBoss EAP 6.4 **org.hornetq** module, remova as seguintes linhas do arquivo:

```
<resource-root path="lib"/>
<resource-root path="../../../../org/hornetq/main/lib"/>
```



ATENÇÃO

Falha ao remover o HornetQ **lib** caminho **resource-root** fará com que a ponte falhe com o seguinte erro no arquivo de log.

```
2016-07-15 09:32:25,660 ERROR
[org.jboss.as.controller.management-operation]
(management-handler-thread - 2) WFLYCTL0013:
Operation ("add") failed - address: ([
  ("subsystem" => "messaging-activemq"),
  ("jms-bridge" => "myBridge")
]) - descrição da falha: "WFLYMSGAMQ0086: Não
foi possível carregar módulo org.hornetq"
```

- Copie o **org.jboss.netty** módulo no JBoss EAP 7.x **EAP_HOME/modules/org/jboss/** diretório.
 - Se você não aplicou patches neste módulo, copie esta pasta do servidor do JBoss EAP 6.4: **EAP6_HOME/modules/system/layers/base/org/jboss/netty/**
 - Se você aplicou patches neste módulo, copie esta pasta do servidor do JBoss EAP 6.4: **EAP6_HOME/modules/system/layers/base/.overlays/layer-base-jboss-eap-6.4.x.CP/org/jboss/netty**
- Crie a fila do JMS para conter as mensagens recebidas do servidor do JBoss EAP 6.4. A seguir, um exemplo de um comando da CLI de gerenciamento que cria o **MigratedMessagesQueue** Fila JMS para receber a mensagem.

```
jms-queue add --queue-address=MigratedMessagesQueue --entries=
[jms/queue/MigratedMessagesQueue
java:jboss/exported/jms/queue/MigratedMessagesQueue]
```

Isso cria o seguinte **jms-queue** configuração para o servidor padrão no **messaging-activemq** subsistema do servidor do JBoss EAP 7.x.

```
<jms-queue name="MigratedMessagesQueue"
entries="jms/queue/MigratedMessagesQueue
java:jboss/exported/jms/queue/MigratedMessagesQueue"/>
```

- Certifique-se de que **messaging-activemq** subsistema **default** servidor contém uma configuração para o **InVmConnectionFactory connection-factory** semelhante ao seguinte:

—

```
<connection-factory name="InVmConnectionFactory" factory-
type="XA_GENERIC" entries="java:/ConnectionFactory" connectors="in-
vm"/>
```

Se não contiver a entrada, crie uma usando o seguinte comando da CLI de gerenciamento:

```
/subsystem=messaging-activemq/server=default/connection-
factory=InVmConnectionFactory:add(factory-type=XA_GENERIC,
connectors=[in-vm], entries=[java:/ConnectionFactory])
```

4. Crie e implemente uma ponte JMS que leia mensagens do **InQueue** Fila JMS configurada no servidor do JBoss EAP 6.4 e as transfere para o servidor **MigratedMessagesQueue** configurado no servidor do JBoss EAP 7.x.

```
/subsystem=messaging-activemq/jms-bridge=myBridge:add(add-messageID-
in-header=true,max-batch-time=100,max-batch-size=10,max-retries=-
1,failure-retry-interval=1000,quality-of-
service=AT_MOST_ONCE,module=org.hornetq,source-
destination=jms/queue/InQueue,source-connection-
factory=jms/RemoteConnectionFactory,source-context=
[("java.naming.factory.initial"=>"org.wildfly.naming.client.WildFlyI
nitialContextFactory"),
("java.naming.provider.url"=>"remote://127.0.0.1:4447")],target-
destination=jms/queue/MigratedMessagesQueue,target-connection-
factory=java:/ConnectionFactory)
```

Isso cria o seguinte **jms-bridge** configuração no **messaging-activemq** subsistema do servidor do JBoss EAP 7.x.

```
<jms-bridge name="myBridge" add-messageID-in-header="true" max-
batch-time="100" max-batch-size="10" max-retries="-1" failure-retry-
interval="1000" quality-of-service="AT_MOST_ONCE"
module="org.hornetq">
  <source destination="jms/queue/InQueue" connection-
factory="jms/RemoteConnectionFactory">
    <source-context>
      <property name="java.naming.factory.initial"
value="org.wildfly.naming.client.WildFlyInitialContextFactory"/>
      <property name="java.naming.provider.url"
value="remote://127.0.0.1:4447"/>
    </source-context>
  </source>
  <target destination="jms/queue/MigratedMessagesQueue"
connection-factory="java:/ConnectionFactory"/>
</jms-bridge>
```

5. Se a segurança estiver configurada para o JBoss EAP 6.4, você também deve configurar a configuração da ponte JMS **<source>** elemento para incluir um **source-context** que especifica o nome de usuário e a senha corretos a serem usados para a consulta JNDI ao criar a conexão.

Migrar dados de mensagem [Esta é uma tradução automática]

1. Verifique se a informação que você forneceu para a seguinte configuração está correta.

- Qualquer fila e nomes de tópicos.
 - o `java.naming.provider.url` para pesquisa de JNDI.
2. Certifique-se de ter implementado o destino JMS de destino no servidor do JBoss EAP 7.x.
 3. Inicie os servidores do JBoss EAP 6.4 e do JBoss EAP 7.x.

4.9.2.3. Mapeamento dos nomes da pasta de mensagens [Esta é uma tradução automática]

A tabela a seguir mostra os nomes dos diretórios do sistema de mensagens usados no release anterior e os nomes correspondentes usados no release atual do JBoss EAP. Os diretórios são relativos ao `jboss.server.data.dir` diretório, cujo padrão é `EAP_HOME/standalone/data/` se não for especificado.

Exemplo: Nome do Driver do JBoss EAP 6 [Esta é uma tradução automática]	Exemplo: Nome do Driver do JBoss EAP 7 [Esta é uma tradução automática]
<code>messagingbindings/</code>	<code>activemq/bindings/</code>
<code>messagingjournal/</code>	<code>activemq/journal/</code>
<code>messaginglargemessages/</code>	<code>activemq/largemessages/</code>
<code>messagingpaging/</code>	<code>activemq/paging/</code>



NOTA

o `messaginglargemessages/` e `messagingpaging/` diretórios podem não estar presentes se não houver mensagens grandes ou se a paginação estiver desativada.

4.9.2.4. Fazendo backup dos dados da pasta de mensagens [Esta é uma tradução automática]

Se o servidor de destino já tiver processado mensagens, é recomendável fazer backup das pastas de mensagens de destino em um local de backup antes de começar. A localização padrão das pastas de mensagens é `EAP_HOME/standalone/data/activemq/`; no entanto, é configurável. Se você não tiver certeza da localização de seus dados de mensagens, poderá usar os seguintes comandos da CLI de gerenciamento para encontrar o local das pastas de mensagens.

```
/subsystem=messaging-activemq/server=default/path=journal-
directory:resolve-path
/subsystem=messaging-activemq/server=default/path=paging-
directory:resolve-path
/subsystem=messaging-activemq/server=default/path=bindings-
directory:resolve-path
/subsystem=messaging-activemq/server=default/path=large-messages-
directory:resolve-path
```

Depois de saber a localização das pastas, copie cada pasta para um local de backup seguro.

4.9.3. Migrar as destinações JMS [Esta é uma tradução automática]

Nas versões anteriores do JBoss EAP, as filas de destino JMS foram configuradas no `<jms-destinations>` elemento sob o `<hornetq-server>` elemento no `messaging` subsistema.

```
<hornetq-server>
...
<jms-destinations>
  <jms-queue name="testQueue">
    <entry name="queue/test"/>
    <entry name="java:jboss/exported/jms/queue/test"/>
  </jms-queue>
</jms-destinations>
...
</hornetq-server>
```

No JBoss EAP 7, a fila de destino do JMS é configurada no padrão `<server>` elemento do `messaging-activemq` subsistema.

```
<server name="default">
...
<jms-queue name="testQueue" entries="queue/test
java:jboss/exported/jms/queue/test"/>
...
</server>
```

4.9.4. Migrar interceptores de mensagem [Esta é uma tradução automática]

Interceptores de mensagem foram alterados significativamente no JBoss EAP 7 com a substituição do HornetQ pelo ActiveMQ Artemis como o provedor de mensagem JMS.

O HornetQ `messaging` Um subsistema incluído no lançamento anterior do JBoss EAP requeria que você instalasse os interceptores HornetQ adicionando-os a um JAR e então modificando o HornetQ `module.xml` Arquivo.

o `messaging-activemq` subsistema incluído no JBoss EAP 7 não requer modificação de um `module.xml` Arquivo. Classes de interceptadores do usuário, que agora implementam o Apache ActiveMQ Artemis `Interceptor` interface, agora pode ser carregado de qualquer módulo do servidor. Você especifica o módulo do qual o interceptador deve ser carregado no `messaging-activemq` subsistema do arquivo de configuração do servidor.

Exemplo: Configuração do Interceptor [Esta é uma tradução automática]

```
<subsystem xmlns="urn:jboss:domain:messaging-activemq:2.0">
  <server name="default">
    ...
    <incoming-interceptors>
      <class name="com.mycompany.incoming.myInterceptor"
module="com.mycompany" />
      <class name="com.othercompany.incoming.myOtherInterceptor"
module="com.othercompany" />
    </incoming-interceptors>
    <outgoing-interceptors>
      <class name="com.mycompany.outgoing.myInterceptor"
```

```

module="com.mycompany" />
    <class name="com.othercompany.outgoing.myOtherInterceptor"
module="com.othercompany" />
    </outgoing-interceptors>
</server>
</subsystem>

```

4.9.5. Substituir as configurações Netty Servlet [Esta é uma tradução automática]

No JBoss EAP 6, você poderia configurar um mecanismo de servlet para trabalhar com o transporte Netty Servlet. Como o ActiveMQ Artemis substitui o HornetQ como o provedor de mensagens integrado no JBoss EAP 7, essa configuração não está mais disponível. Você deve substituir a configuração de servlet para usar os novos conectores HTTP de mensagens e os aceitadores HTTP.

4.9.6. Configurando um Adaptador de Recursos JMS Genérico [Esta é uma tradução automática]

A maneira como você configura um adaptador de recursos JMS genérico para uso com um provedor JMS de terceiros foi alterada no JBoss EAP 7. Para obter mais informações, consulte [Implantando um Adaptador de Recursos JMS Genérico](#) dentro *Configurando o Messaging* para o JBoss EAP.

4.9.7. Mudanças de Configuração de Mensagens no JBoss EAP 7.1 [Esta é uma tradução automática]

No JBoss EAP 7.0, se você configurou o **replication-master** política sem especificar o **check-for-live-server** atributo, seu valor padrão era **false**. Isso mudou no JBoss EAP 7.1. O valor padrão para o **check-for-live-server** atributo é agora **true**.

A seguir, um exemplo de um comando da CLI de gerenciamento que configura **replication-master** política sem especificar o **check-for-live-server** atributo.

```

/subsystem=messaging-activemq/server=default/ha-policy=replication-
master:add(cluster-name=my-cluster,group-name=group1)

```

Quando você lê o recurso usando a CLI de gerenciamento, observe que **check-for-live-server** o valor do atributo está definido como **true**.

```

/subsystem=messaging-activemq/server=default/ha-policy=replication-
master:read-resource(recursive=true)
{
  "outcome" => "success",
  "result" => {
    "check-for-live-server" => true,
    "cluster-name" => "my-cluster",
    "group-name" => "group1",
    "initial-replication-sync-timeout" => 30000L
  },
  "response-headers" => {"process-state" => "reload-required"}
}

```

4.9.8. Alterações no Comportamento de Serialização do JMS entre Liberações [Esta é uma tradução automática]

o **serialVersionUID** do `javax.jms.JMSEException` mudou entre o JMS 1.1 e o JMS 2.0.0. Isso significa que, se uma instância de um **JMSEException**, ou qualquer uma de suas subclasses, é serializado usando o JMS 1.1, não pode ser desserializado usando o JMS 2.0.0. O contrário também é verdade. Se uma instância de **JMSEException** é serializado usando o JMS 2.0.0, ele não pode ser desserializado usando o JMS 1.1. Em ambos os casos, lança uma exceção semelhante à seguinte:

```
javax.jms.JMSEException: javax.jms.JMSEException; local class incompatible:
stream classdesc serialVersionUID = 8951994251593378324, local class
serialVersionUID = 2368476267211489441
```

Esse problema foi corrigido na versão de manutenção do JMS 2.0.1.

Implementação do JMS para cada versão do JBoss EAP [Esta é uma tradução automática]

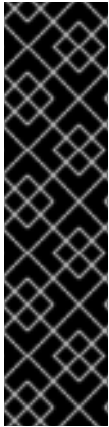
Tabela 4.4. Implementação do JMS para cada versão do JBoss EAP [Esta é uma tradução automática]

JBoss EAP Version	Implementação JMS	Versão JMS
6.4	HornetQ	JMS 1.1
7.0	Apache ActiveMQ Artemis	JMS 2.0.0
7.1	Apache ActiveMQ Artemis	JMS 2.0.1

Esteja ciente de que o **serialVersionUID** incompatibilidade pode resultar em um problema de migração nas seguintes situações:

- Se você enviar uma mensagem que contenha **JMSEException** usando um cliente do JBoss EAP 6.4, migre seus dados do sistema de mensagens para o JBoss EAP 7.0 e tente desserializar a mensagem usando um cliente do JBoss EAP 7.0, a desserialização falhará e lançará uma exceção. Isso é porque o **serialVersionUID** no JMS 1.1 é *não* compatível com o do JMS 2.0.0.
- Se você enviar uma mensagem que contenha **JMSEException** Usando um cliente do JBoss EAP 7.0, migre seus dados do sistema de mensagens para o JBoss EAP 7.1 e tente desserializar essa mensagem usando um cliente do JBoss EAP 7.1, a desserialização falhará e lançará uma exceção. Isso é porque o **serialVersionUID** no JMS 2.0.0 é *não* compatível com o do JMS 2.0.1.

Note que se você enviar uma mensagem que contenha **JMSEException** Usando um cliente do JBoss EAP 6.4, migre seus dados do sistema de mensagens para o JBoss EAP 7.1 e tente desserializar essa mensagem usando um cliente do JBoss EAP 7.1, a desserialização será bem sucedida porque o **serialVersionUID** no JMS 1.1 é compatível com o do JMS 2.0.1.



IMPORTANTE

A Red Hat recomenda que você faça o seguinte antes de migrar seus dados de mensagens:

- Certifique-se de consumir todas as mensagens do JMS 1.1 que contenham `JMSExceptions` antes de migrar os dados do sistema de mensagens do JBoss EAP 6.4 para o JBoss EAP 7.0.
- Certifique-se de consumir todas as mensagens do JMS 2.0.0 que contenham `JMSExceptions` antes de migrar os dados do sistema de mensagens do JBoss EAP 7.0 para o JBoss EAP 7.1.

4.10. MUDANÇAS NO GERENCIAMENTO DE JMX [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

O componente HornetQ no JBoss EAP 6 forneceu seu próprio gerenciamento JMX; no entanto, não foi recomendado e agora está obsoleto e não é mais suportado. Se você se baseou neste recurso no JBoss EAP 6, você deve migrar suas ferramentas de gerenciamento para usar a CLI de gerenciamento do JBoss EAP ou o gerenciamento JMX fornecido com o JBoss EAP 7.

Você também deve atualizar suas bibliotecas de clientes para usar o `jboss-client.jar` que vem com o JBoss EAP 7.

A seguir, um exemplo do código de gerenciamento do HornetQ JMX que foi usado no JBoss EAP 6.

```
JMXConnector connector = null;
try {
    HashMap environment = new HashMap();
    String[] credentials = new String[]{"admin", "Password123!"};
    environment.put(JMXConnector.CREDENTIALS, credentials);

    // HornetQ used the protocol "remoting-jmx" and port "9999"
    JMXServiceURL beanServerUrl = new JMXServiceURL("service:jmx:remoting-
jmx://127.0.0.1:9999");

    connector = JMXConnectorFactory.connect(beanServerUrl, environment);
    MBeanServerConnection mbeanServer =
connector.getMBeanServerConnection();

    // The JMX object name pointed to the HornetQ JMX management
    ObjectName objectName = new
ObjectName("org.hornetq:type=Server,module=JMS");

    // The invoked method name was "listConnectionIDs"
    String[] connections = (String[]) mbeanServer.invoke(objectName,
"listConnectionIDs", new Object[]{}, new String[]{});
    for (String connection : connections) {
        System.out.println(connection);
    }
} finally {
    if (connector != null) {
        connector.close();
    }
}
```


A seguir, um exemplo do código equivalente necessário para o ActiveMQ Artemis no JBoss EAP 7.

```
JMXConnector connector = null;
try {
    HashMap environment = new HashMap();
    String[] credentials = new String[]{"admin", "Password123!"};
    environment.put(JMXConnector.CREDENTIALS, credentials);

    // ActiveMQ Artemis uses the protocol "remote+http" and port "9990"
    JMXServiceURL beanServerUrl = new
JMXServiceURL("service:jmx:remote+http://127.0.0.1:9990");

    connector = JMXConnectorFactory.connect(beanServerUrl, environment);
    MBeanServerConnection mbeanServer =
connector.getMBeanServerConnection();

    // The JMX object name points to the new JMX management in the
`messaging-activemq` subsystem
    ObjectName objectName = new ObjectName("jboss.as:subsystem=messaging-
activemq,server=default");

    // The invoked method name is now "listConnectionIds"
    String[] connections = (String[]) mbeanServer.invoke(objectName,
"listConnectionIds", new Object[]{}, new String[]{});
    for (String connection : connections) {
        System.out.println(connection);
    }
} finally {
    if (connector != null) {
        connector.close();
    }
}
```

Observe que os nomes e parâmetros do método foram alterados na nova implementação. Você pode encontrar os novos nomes de métodos no JConsole seguindo estas etapas.

1. Conecte-se ao JConsole usando o seguinte comando.

```
$ EAP_HOME/bin/jconsole.sh
```

2. Conecte-se ao processo local do JBoss EAP. Note que deve começar com "jboss-modules.jar".
3. No **MBeans** aba, escolha **jboss.as** → **messaging-activemq** → **padrão** → **Operações** para exibir a lista de nomes e atributos de métodos.

4.11. ALTERAÇÕES DE CONFIGURAÇÕES DE SERVIDOR ORB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

A implementação do JacORB foi substituída com uma ramificação downstream do OpenJDK ORB no JBoss EAP 7.

o **org.jboss.as.jacorb** módulo de extensão, localizado em **EAP_HOME/modules/system/layers/base/**, foi substituído pelo **org.wildfly.iiop-openjdk** módulo de extensão.

o **urn:jboss:domain:jacorb:1.4** o namespace de configuração do subsistema no arquivo de configuração do servidor foi substituído pelo **urn:jboss:domain:iiop-openjdk:1.0** namespace.

O seguinte é um exemplo do padrão **jacorb** configuração do sistema no JBoss EAP 6.

```
<subsystem xmlns="urn:jboss:domain:jacorb:1.4">
  <orb socket-binding="jacorb" ssl-socket-binding="jacorb-ssl">
    <initializers security="identity" transactions="spec"/>
  </orb>
</subsystem>
```

O seguinte é um exemplo do padrão **iiop-openjdk** configuração do subsistema no JBoss EAP 7.

```
<subsystem xmlns="urn:jboss:domain:iiop-openjdk:1.0">
  <orb socket-binding="jacorb" ssl-socket-binding="jacorb-ssl" />
  <initializers security="identity" transactions="spec" />
</subsystem>
```

O novo **iiop-openjdk** A configuração do subsistema aceita apenas um subconjunto dos elementos e atributos legados. O seguinte é um exemplo de um **jacorb** configuração do subsistema na versão anterior do JBoss EAP que contém todos os elementos e atributos válidos:

```
<subsystem xmlns="urn:jboss:domain:jacorb:1.4">
  <orb name="JBoss" print-version="off" use-imr="off" use-bom="off"
cache-typecodes="off"
  cache-poa-names="off" giop-minor-version="2" socket-
binding="jacorb" ssl-socket-binding="jacorb-ssl">
    <connection retries="5" retry-interval="500" client-timeout="0"
server-timeout="0"
      max-server-connections="500" max-managed-buf-size="24" outbuf-
size="2048"
      outbuf-cache-timeout="-1"/>
    <initializers security="off" transactions="spec"/>
  </orb>
  <poa monitoring="off" queue-wait="on" queue-min="10" queue-max="100">
    <request-processors pool-size="10" max-threads="32"/>
  </poa>
  <naming root-context="JBoss/Naming/root" export-corbaloc="on"/>
  <interop sun="on" comet="off" iona="off" chunk-custom-rmi-
valuetypes="on"
    lax-boolean-encoding="off" indirection-encoding-disable="off"
strict-check-on-tc-creation="off"/>
  <security support-ssl="off" add-component-via-interceptor="on" client-
supports="MutualAuth"
    client-requires="None" server-supports="MutualAuth" server-
requires="None"/>
  <properties>
    <property name="some_property" value="some_value"/>
  </properties>
</subsystem>
```

Os seguintes atributos de elementos não possuem mais suporte e devem ser removidos.

Tabela 4.5. Atributos para remover [Esta é uma tradução automática]

Elemento	Atributos sem suporte
<orb>	<ul style="list-style-type: none"> • client-timeout • max-managed-buf-size • max-server-connections • outbuf-cache-timeout • outbuf-size • connection retries • retry-interval • name • server-timeout
<poa>	<ul style="list-style-type: none"> • queue-min • queue-max • pool-size • max-threads

Os seguintes **on/off** os atributos não são mais suportados e não serão migrados quando você executar a CLI de gerenciamento **migrate** Operação. Se eles estão definidos para **on**, você receberá um aviso de migração. De outros **on/off** atributos que não são mencionados nesta tabela, por exemplo **<security support-ssl="on|off">**, ainda são suportados e serão migrados com sucesso. A única diferença é que seus valores serão alterados de **on/off** para **true/false**.

Tabela 4.6. Atributos para desativar ou remover [Esta é uma tradução automática]

Elemento	Atributos para remover [Esta é uma tradução automática]
<orb>	<ul style="list-style-type: none"> • cache-poa-names • cache-typecodes • print-version • use-bom • use-imr

Elemento	Atributos para remover [Esta é uma tradução automática]
<interop>	<p>(tudo exceto sun)</p> <ul style="list-style-type: none"> • comet • iona • chunk-custom-rmi-valuetypes • indirection-encoding-disable • lax-boolean-encoding • strict-check-on-tc-creation
<poa>	<ul style="list-style-type: none"> • monitoring • queue-wait

4.12. MIGRAR A CONFIGURAÇÃO DE SUBSISTEMA DE THREADS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

A configuração do servidor do JBoss EAP 6 incluiu o **threads** subsistema usado para gerenciar conjuntos de encadeamentos nos diversos subsistemas no servidor.

o **threads** subsistema não está mais disponível no JBoss EAP 7. Em vez disso, cada subsistema é responsável por gerenciar seus próprios conjuntos de encadeamentos.

Para obter informações sobre como configurar pools de threads para o **infinispan** subsistema, consulte [Configurar pools de threads do Infinispan](#) no JBoss EAP *Guia de configuração*.

Para obter informações sobre como configurar pools de threads para o **jgroups** subsistema, consulte [Configurar pools de threads do JGroups](#) no JBoss EAP *Guia de configuração*.

No JBoss EAP 6, você configurou pools de threads para conectores e listeners para o **web** subsistema referenciando um **executor** que foi definido no **threads** subsistema. No JBoss EAP 7, você agora configura pools de threads para o **undertow** subsistema referenciando um **worker** que é definido no **io** subsistema. Para mais informações, veja [Configurando o subsistema de E / S](#) no JBoss EAP *Guia de configuração*.

Para obter informações sobre as alterações na configuração do pool de threads no **remoting** subsistema, consulte [Migrar a configuração do subsistema remoto](#) neste guia, e [Configurando o Endpoint](#) no JBoss EAP *Guia de configuração*.

4.13. MIGRAR A CONFIGURAÇÃO DE SUBSISTEMA REMOTO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

No JBoss EAP 6, você configurou o pool de threads para o **remoting** subsistema, definindo vários **worker-*** atributos. O pool de threads de trabalho não está mais configurado no **remoting** subsistema no JBoss EAP 7 e se você tentar modificar a configuração existente, você verá a seguinte

mensagem.

```
WFLYRMT0022: Worker configuration is no longer used, please use endpoint
worker configuration
```

No JBoss EAP 7, o pool de threads de trabalho é substituído por uma configuração de terminal que referencia **worker** definido no **io** subsistema.

Para obter informações sobre como configurar o nó de extremidade, consulte [Configurando o Endpoint](#) no JBoss EAP *Guia de configuração*.

4.14. ALTERAÇÕES DE CONFIGURAÇÕES DE SERVIDOR WEBSOCKET [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Para usar WebSockets no JBoss EAP 6, você tinha que habilitar o protocolo do conector NIO2 sem bloqueio para o **http** conector no **web** subsistema do arquivo de configuração do servidor JBoss EAP usando um comando similar ao seguinte.

```
/subsystem=web/connector=http/:write-
attribute(name=protocol,value=org.apache.coyote.http11.Http11NioProtocol)
```

Para usar WebSockets em um aplicativo, você também tinha que criar um **<enable-websockets>** elemento na aplicação **WEB-INF/jboss-web.xml** arquivo e configurá-lo para **true**.

No JBoss EAP 7, você não precisa mais configurar o servidor para suporte padrão a WebSocket ou configurar o aplicativo para usá-lo. WebSockets são um requisito da especificação Java EE 7 e os protocolos necessários são configurados por padrão. Uma configuração WebSocket mais complexa é feita no **servlet-container** do **undertow** subsistema do arquivo de configuração do servidor JBoss EAP. Você pode visualizar as configurações disponíveis usando o seguinte comando.

```
/subsystem=undertow/servlet-container=default/setting=websockets:read-
resource(recursive=true)
{
  "outcome" => "success",
  "result" => {
    "buffer-pool" => "default",
    "dispatch-to-worker" => true,
    "worker" => "default"
  }
}
```

Para obter mais informações sobre o desenvolvimento do WebSocket, consulte [Criando Aplicativos WebSocket](#) no JBoss EAP *Guia de desenvolvimento*.

Exemplos de códigos WebSocket também podem ser encontrados no início rápido (quickstarts) que é fornecido com JBoss EAP.

4.15. ALTERAÇÕES DO SERVIDOR SINGLE SIGN-ON [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

o **infinispan** O subsistema ainda fornece suporte a caching distribuído para serviços de HA na forma de caches Infinispan no JBoss EAP 7; no entanto, o armazenamento em cache e a distribuição de informações de autenticação são tratados de maneira diferente das versões anteriores.

No JBoss EAP 6, se o single sign-on (SSO) não foi fornecido um cache Infinispan, o cache não era distribuído.

No JBoss EAP 7, SSO é distribuído automaticamente quando você seleciona o perfil HA. Quando executar com o perfil HA, cada host tem seu próprio cache Infinispan, que é baseado no cache padrão do contêiner de cache da web. Este cache armazena sessões relevantes e informações de cookie SSO para o host. JBoss EAP trata a propagação de informação de cache individual para todos os hosts. Não existe nenhuma maneira para atribuir especificamente um cache Infinispan para um SSO no JBoss EAP 7.

No JBoss EAP 7, o SSO é configurado no **undertow** subsistema do arquivo de configuração do servidor.

Não há exigência de alteração de código de aplicativos para SSO quando migrando para JBoss EAP 7.

4.16. ALTERAÇÕES NA CONFIGURAÇÃO DA FONTE DE DADOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

4.16.1. Nome do Driver da Origem de Dados JDBC [Esta é uma tradução automática]

Quando você configurou uma fonte de dados no release anterior do JBoss EAP, o valor especificado para o nome do driver dependia do número de classes listadas no **META-INF/services/java.sql.Driver** arquivo contido no JAR do driver JDBC.

Driver contendo uma classe única

Se o **META-INF/services/java.sql.Driver** arquivo especificado apenas uma classe, o valor do nome do driver era simplesmente o nome do JAR do driver JDBC. Isto não mudou no JBoss EAP 7.

Driver contendo classes múltiplas

Em JBoss EAP 6, se houvesse mais de uma classe listada no **META-INF/services/java.sql.Driver** arquivo, você especificou qual classe era a classe do driver, anexando seu nome ao nome do JAR, juntamente com a versão principal e secundária, no seguinte formato.

```
JAR_NAME + DRIVER_CLASS_NAME + "_" + MAJOR_VERSION + "_" + MINOR_VERSION
```

No JBoss EAP 7, isto mudou. Você agora especifica o nome do driver utilizando o seguinte formato.

```
JAR_NAME + "_" + DRIVER_CLASS_NAME + "_" + MAJOR_VERSION + "_" + MINOR_VERSION
```



NOTA

Um sublinhado foi adicionado entre o *JAR_NAME* e a *DRIVER_CLASS_NAME*.

O driver JDBC do MySQL 5.1.31 é um exemplo de um driver que contém duas classes. O nome da classe do driver é **com.mysql.jdbc.Driver**. Os exemplos a seguir demonstram as diferenças entre como você especifica o nome do driver na versão anterior e atual do JBoss EAP.

Exemplo: Nome do Driver do JBoss EAP 6 [Esta é uma tradução automática]

```
mysql-connector-java-5.1.31-bin.jarcom.mysql.jdbc.Driver_5_1
```

Exemplo: Nome do Driver do JBoss EAP 7 [Esta é uma tradução automática]

```
mysql-connector-java-5.1.31-bin.jar_com.mysql.jdbc.Driver_5_1
```

4.17. ALTERAÇÕES DE CONFIGURAÇÕES DE SERVIDOR DE SEGURANÇA [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Se você migrar para o JBoss EAP 7 e planejar executar com o Java Security Manager ativado, deve estar ciente de que as alterações foram feitas na maneira como as políticas são definidas e que mudanças adicionais na configuração podem ser necessárias. Também esteja ciente de que os gerenciadores de segurança customizados não são suportados no JBoss EAP 7.

Para obter informações sobre as alterações de configuração do servidor Java Security Manager, consulte [Considerações que se movem de versões anteriores](#) dentro *Como configurar a segurança do servidor* para o JBoss EAP.

4.17.1. Mudanças no Comportamento de Segurança Legado entre o JBoss EAP 7.0 e o JBoss EAP 7.1 [Esta é uma tradução automática]

4.17.1.1. Alteração de status HTTP para domínios LDAP inacessíveis [Esta é uma tradução automática]

Se nenhum domínio LDAP fosse acessível pelo servidor no JBoss EAP 7.0, o **security** subsistema retornou um código de status HTTP de "401 Não Autorizado". Isso mudou no JBoss EAP 7.1. O legado **security** O subsistema no JBoss EAP 7.1 retorna um código de status HTTP de "500 Internal Error" para descrever com mais precisão que ocorreu uma condição inesperada que impedia o servidor de processar a solicitação com êxito.

4.17.1.2. Habilitando o domínio de segurança do LDAP para analisar funções de um DN [Esta é uma tradução automática]

No JBoss EAP 7.0, o

org.jboss.as.domain.management.security.parseGroupNameFromLdapDN A propriedade **system** foi usada para permitir que a região de segurança LDAP analise as funções de um DN. Quando esta propriedade foi definida como **true**, papéis foram analisados a partir de um DN. Caso contrário, uma pesquisa LDAP normal foi usada para procurar por funções.

No JBoss EAP 7.1, esta propriedade do sistema agora está obsoleta. Em vez disso, você configura essa opção definindo o recém-introduzido **parse-group-name-from-dn** atribuir a **true** no caminho do serviço principal usando o seguinte comando da CLI de gerenciamento:

```
/core-service=management/security-  
realm=REALM_NAME/authorization=ldap/group-search=principal-to-  
group:add(parse-group-name-from-dn=true)
```

4.17.1.3. Mudanças no envio do certificado SSL do JBoss EAP para um servidor LDAP [Esta é uma tradução automática]

No JBoss EAP 7.0, quando a interface de gerenciamento é configurada para usar o **ldapSSL** região de

segurança, a autenticação mútua entre o servidor e o LDAP pode falhar, resultando em uma falha de autenticação na interface de gerenciamento. Isso ocorre porque duas conexões LDAP diferentes são feitas, cada uma por um encadeamento diferente, e elas não compartilham as sessões SSL.

O JBoss EAP 7.1 apresenta um novo booleano **always-send-client-cert** atributo de gerenciamento no LDAP **outbound-connection**. Esta opção permite a configuração de conexões LDAP de saída para suportar servidores LDAP configurados para sempre exigir um certificado de cliente.

A autenticação LDAP acontece em duas etapas:

1. Ele procura pela conta.
2. Verifica as credenciais.

Por padrão, o **always-send-client-cert** atributo está definido como **false**, o que significa que o certificado SSL do cliente é enviado apenas com a primeira solicitação de conta de pesquisa. Quando este atributo está configurado para **true**, o cliente LDAP do JBoss EAP envia o certificado do cliente para o servidor LDAP com as solicitações de pesquisa e verificação.

Você pode definir esse atributo para **true** usando o seguinte comando CLI de gerenciamento.

```
/core-service=management/ldap-connection=my-ldap-connection:write-attribute(name=always-send-client-cert,value=true)
```

Isso resulta na seguinte conexão de saída LDAP no arquivo de configuração do servidor.

```
<management>
  ....
  <outbound-connections>
    <ldap name="my-ldap-connection" url="ldap://127.0.0.1:389" search-
dn="cn=search,dc=myCompany,dc=com" search-credential="myPass" always-send-
client-cert="true"/>
  </outbound-connections>
  ....
</management>
```

4.17.2. Mudanças no modo FIPS [Esta é uma tradução automática]

Mudanças no Comportamento de Segurança Legado entre o JBoss EAP 7.0 e o JBoss EAP 7.1 [Esta é uma tradução automática]

Ao usar regiões de segurança legadas, o JBoss EAP 7.1 fornece a geração automática de um certificado autoassinado para fins de desenvolvimento. Este recurso, que não estava disponível no JBoss EAP 7.0, é ativado por padrão. Isso significa que, se você estiver executando no modo FIPS, deverá configurar o servidor para desativar a criação automática de certificado autoassinado. Caso contrário, você poderá ver o seguinte erro ao iniciar o servidor.

```
ERROR [org.xnio.listener] (default I/O-6) XNI0001007: A channel event
listener threw an exception: java.lang.RuntimeException: WFLYDM0114:
Failed to lazily initialize SSL context
...
Caused by: java.lang.RuntimeException: WFLYDM0112: Failed to generate self
signed certificate
```


...

Caused by: java.security.KeyStoreException: Cannot get key bytes, not PKCS#8 encoded

Para obter informações sobre criação automática de certificado autoassinado, consulte [Criação automática de certificado autoassinado para aplicativos](#) dentro *Como configurar a segurança do servidor* para o JBoss EAP.

4.18. ALTERAÇÕES DO SUBSISTEMA DE TRANSAÇÕES [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Alguns atributos de configuração do Gerenciador de Transações que estavam disponíveis no **transactions** O subsistema no JBoss EAP 6 foi alterado no JBoss EAP 7.

Alterações do subsistema de transações [Esta é uma tradução automática]

A tabela a seguir lista os atributos do JBoss EAP 6 que foram removidos do **transactions** subsistema no JBoss EAP 7 e os atributos de substituição equivalentes.

Atributo no JBoss EAP 6	Mudanças no Cliente EJB no JBoss EAP 7 [Esta é uma tradução automática]
path	object-store-path
relative-to	object-store-relative-to

Alterações do subsistema de transações [Esta é uma tradução automática]

Os seguintes atributos que estavam disponíveis no **transactions** O subsistema no JBoss EAP 6 está obsoleto no JBoss EAP 7. Os atributos obsoletos podem ser removidos em uma versão futura do produto. A tabela a seguir lista os atributos de substituição equivalentes.

Atributo no JBoss EAP 6	Mudanças no Cliente EJB no JBoss EAP 7 [Esta é uma tradução automática]
use-hornetq-store	use-journal-store
hornetq-store-enable-async-io-to	journal-store-enable-async-io
enable-statistics	statistics-enabled

4.19. CHANGES TO MOD_CLUSTER CONFIGURATION [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

A configuração para listas de proxy estáticas em mod_cluster foi alterada no JBoss EAP 7.

No JBoss EAP 6, você configurou o **proxy-list** atributo, que era uma lista separada por vírgulas de endereços de proxy httpd especificados no formato de **hostname:port**.

o **proxy-list** atributo está obsoleto no JBoss EAP 7. Ele foi substituído pelo **proxies** attribute, que é uma lista de nomes de ligação de soquete de saída.

Essa alteração afeta o modo como você define uma lista de proxy estático, por exemplo, ao desativar a publicidade de `mod_cluster`. Para obter informações sobre como desativar a publicidade para o `mod_cluster`, consulte [Desativar publicidade para mod_cluster](#) no JBoss EAP *Guia de configuração*.

Para obter mais informações sobre os atributos do `mod_cluster`, consulte [Atributos do Subsistema ModCluster](#) no JBoss EAP *Guia de configuração*.

4.20. EXIBINDO ALTERAÇÕES NA CONFIGURAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

O JBoss EAP 7 fornece a capacidade de rastrear as alterações de configuração feitas no servidor em execução. Isso permite que os administradores visualizem um histórico de alterações de configuração feitas por usuários autorizados.

No JBoss EAP 7.0, você deve usar o **core-service** comando CLI de gerenciamento para configurar opções e listar alterações de configuração recentes.

Exemplo: Listar Mudanças de Configuração no JBoss EAP 7.0 [Esta é uma tradução automática]

```
/core-service=management/service=configuration-changes:add(max-history=10)
/core-service=management/service=configuration-changes:list-changes
```

O JBoss EAP 7.1 introduziu um novo **core-management** subsistema que pode ser configurado para rastrear as alterações de configuração feitas no servidor em execução. Este é o método preferido de configuração e visualização de mudanças de configuração no JBoss EAP 7.1.

Exemplo: Listar Mudanças de Configuração no JBoss EAP 7.1 [Esta é uma tradução automática]

```
/subsystem=core-management/service=configuration-changes:add(max-history=20)
/subsystem=core-management/service=configuration-changes:list-changes
```

Para mais informações sobre como usar o novo **core-management** subsistema introduzido no JBoss EAP 7.1, veja [Visualizar alterações na configuração](#) no JBoss EAP *Guia de configuração*.

CAPÍTULO 5. ALTERAÇÕES NA MIGRAÇÃO DOS APLICATIVOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

5.1. ALTERAÇÕES NOS APLICATIVOS DE SERVIÇOS WEB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

O JBossWS 5 traz novos recursos e melhorias de desempenho aos serviços Web do JBoss EAP 7, principalmente através de atualizações [Apache CXF](#), [Apache WSS4J](#), e [Apache Santuario](#) componentes.

5.1.1. Alterações de suporte JAX-RPC [Esta é uma tradução automática]

A API Java para RPC (JAX-RPC) baseada em XML foi preterida no Java EE 6 e é opcional no Java EE 7. Ela não está mais disponível ou é suportada no JBoss EAP 7. Os aplicativos que usam JAX-RPC devem ser migrados para uso [JAX-WS](#), que é a atual estrutura de serviços da Web padrão do Java EE.

A utilização dos serviços web JAX-RPC pode ser identificada em qualquer uma das seguintes maneiras:

- A presença de um arquivo de mapeamento JAX-RPC, que é um arquivo XML com o elemento raiz **<java-wsdl-mapping>**.
- A presença de um **webservices.xml** Arquivo descritor XML que contém um **<web-service-description>** elemento, que inclui um **<jaxrpc-mapping-file>** elemento filho. O seguinte é um exemplo de **webservices.xml** arquivo descritor que define um serviço da Web JAX-RPC.

```
<webservices xmlns="http://java.sun.com/xml/ns/j2ee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://www.ibm.com/webservices/xsd/j2ee_web_services_1_1.xsd"
  version="1.1">
  <web-service-description>
    <web-service-description-name>HelloService</web-service-
description-name>
    <wsdl-file>WEB-INF/wsdl/HelloService.wsdl</wsdl-file>
    <jaxrpc-mapping-file>WEB-INF/mapping.xml</jaxrpc-mapping-file>
    <port-component>
      <port-component-name>Hello</port-component-name>
      <wsdl-port>HelloPort</wsdl-port>
      <service-endpoint-
interface>org.jboss.chap12.hello.Hello</service-endpoint-interface>
      <service-impl-bean>
        <servlet-link>HelloWorldServlet</servlet-link>
      </service-impl-bean>
    </port-component>
  </web-service-description>
</webservices>
```

- A presença de um **ejb-jar.xml** arquivo, que contém um **<service-ref>** que faz referência a um arquivo de mapeamento JAX-RPC.

5.1.2. Alterações dos serviços web Apache CXF Spring [Esta é uma tradução automática]

Nas versões anteriores do JBoss EAP, você poderia personalizar a integração do JBossWS e do Apache CXF incluindo uma **jbossws-cxf.xml** arquivo de configuração com o arquivo de implantação do nó de extremidade. Um caso de uso para isso era configurar cadeias de interceptor para terminais de cliente e servidor de serviço da Web no barramento Apache CXF. Essa integração exigiu que o Spring fosse implantado no servidor do JBoss EAP.

A integração Spring não é mais suportada no JBoss EAP 7. Qualquer aplicativo que contenha **jbossws-cxf.xml** O arquivo de configuração do descritor deve ser modificado para substituir a configuração personalizada definida nesse arquivo. Embora ainda seja possível acessar diretamente a API do Apache CXF, esteja ciente de que o aplicativo não será portátil.

A abordagem sugerida é para substituir configurações personalizadas de Spring com as novas opções de configuração de descritor JBossWS quando possível. A abordagem baseada em descritor JBossWS fornece funcionalidade similar sem exigir modificação do código de ponto de extremidade do cliente. Em alguns casos, você pode substituir Spring por Context Dependency Injection (CDI).

Migrar interceptores de mensagem [Esta é uma tradução automática]

O descritor JBossWS fornece novas opções de configuração que permitem declarar os interceptores sem modificar o código do terminal do cliente. Em vez disso, você declara interceptores dentro de configurações predefinidas de cliente e nó de extremidade especificando uma lista de nomes de classe de interceptor para o **cxf.interceptors.in** e **cxf.interceptors.out** propriedades.

O seguinte é um exemplo de um **jaxws-endpoint-config.xml** arquivo que declara interceptores usando essas propriedades.

```
<?xml version="1.0" encoding="UTF-8"?>
<jaxws-config xmlns="urn:jboss:jbossws-jaxws-config:4.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:javaee="http://java.sun.com/xml/ns/javaee"
  xsi:schemaLocation="urn:jboss:jbossws-jaxws-config:4.0 schema/jbossws-
jaxws-config_4_0.xsd">
  <endpoint-config>
    <config-
name>org.jboss.test.ws.jaxws.cxf.interceptors.EndpointImpl</config-name>
    <property>
      <property-name>cxf.interceptors.in</property-name>
      <property-
value>org.jboss.test.ws.jaxws.cxf.interceptors.EndpointInterceptor,org.jbo
ss.test.ws.jaxws.cxf.interceptors.FooInterceptor</property-value>
      </property>
    </property>
      <property-name>cxf.interceptors.out</property-name>
      <property-
value>org.jboss.test.ws.jaxws.cxf.interceptors.EndpointCounterInterceptor<
/property-value>
      </property>
    </endpoints-config>
  </jaxws-config>
```

Recursos Apache CXF

O descritor JBossWS permite que você declare recursos dentro de configurações predefinidas de cliente e terminal, especificando uma lista de nomes de classes de **cxf.features** propriedade.

O seguinte é um exemplo de um **jaxws-endpoint-config.xml** arquivo que declara um recurso usando essa propriedade.

```
<?xml version="1.0" encoding="UTF-8"?>
<jaxws-config xmlns="urn:jboss:jbossws-jaxws-config:4.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:javaee="http://java.sun.com/xml/ns/javaee"
  xsi:schemaLocation="urn:jboss:jbossws-jaxws-config:4.0 schema/jbossws-
jaxws-config_4_0.xsd">
  <endpoint-config>
    <config-name>Custom FI Config</config-name>
    <property>
      <property-name>cxf.features</property-name>
      <property-value>org.apache.cxf.feature.FastInfosetFeature</property-
value>
    </property>
  </endpoint-config>
</jaxws-config>
```

Apache CXF HTTP Transport

No Apache CXF, a configuração de transporte HTTP é obtida especificando **org.apache.cxf.transport.http.HTTPConduit** opções. A integração do JBossWS permite que as condutas sejam modificadas programaticamente usando a API Apache CXF da seguinte forma.

```
import org.apache.cxf.frontend.ClientProxy;
import org.apache.cxf.transport.http.HTTPConduit;
import org.apache.cxf.transports.http.configuration.HTTPClientPolicy;

// Set chunking threshold before using a JAX-WS port client
...
HTTPConduit conduit =
(HTTPConduit)ClientProxy.getClient(port).getConduit();
HTTPClientPolicy client = conduit.getClient();

client.setChunkingThreshold(8192);
...
```

Você também pode controlar e substituir o Apache CXF **HTTPConduit** valores padrão, definindo as propriedades do sistema.

Propriedade	Tipo	Descrição
cxf.client.allowChunking	Boolean	Especifica se envia uma solicitações usando agrupamento
cxf.client.chunkingThreshold	Integer	Define o limite no qual é feita a alteração do modo sem agrupamento para o modo de agrupamento.
cxf.client.connectionTimeout	Long	Define o número de milissegundos de tempo limite da conexão.
cxf.client.receiveTimeout	Long	Define o número de milissegundos para expirar a recepção.

Propriedade	Tipo	Descrição
<code>cxfrclient.connection</code>	String	Especifica se deve usar o Keep-Alive ou close Tipo de conexão.
<code>cxfrtls-client.disableCNCheck</code>	Boolean	Especifica a desativação da verificação de nome de host CN.

5.1.3. Alterações WS-Security

- Se o seu aplicativo contiver um manipulador de retorno de chamada personalizado que acessa o **org.apache.ws.security.WSPasswordCallback** classe, esteja ciente de que esta classe foi movida para o pacote **org.apache.wss4j.common.ext**.
- A maioria dos objetos de bean SAML do **org.apache.ws.security.saml.ext** pacote foram movidos para o **org.apache.wss4j.common.saml** package.
- O uso de transporte de chave RSA v1.5 e todos dos algoritmos relacionados não são permitidos por padrão.
- O Security Token Service (STS) anteriormente validado apenas **onBehalfOf** fichas. Agora também valida **ActAs** fichas. Como consequência, um nome de usuário e senha válidos devem ser especificados no **UsernameToken** que é fornecido para o **ActAs** símbolo.
- Agora, os tokens SAML Bearer precisam ter uma assinatura interna. o **org.apache.wss4j.dom.validate.SamlAssertionValidator** classe agora tem um **setRequireBearerSignature()** método para ativar ou desativar a verificação de assinatura.

5.1.4. Alterações de estruturas de módulos JBoss [Esta é uma tradução automática]

o **cxfrapi** e **cxfrt-core** JARs foram fundidos em um **cxfrcore** JAR. Como consequência, **org.apache.cxf** módulo no JBoss EAP agora contém o **cxfrcore** JAR e expõe mais classes do que na versão anterior.

5.1.5. Alterações e requisitos de Bouncy Castle [Esta é uma tradução automática]

Se você quiser usar criptografia AES com Galois/Counter Mode (GCM) para criptografia simétrica em XML/WS-Security, você precisa do provedor de segurança BouncyCastle.

O JBoss EAP 7 vem com o **org.bouncycastle** módulo e JBossWS agora é capaz de confiar em seu carregador de classe para obter e usar o provedor de segurança BouncyCastle. Portanto, não é mais necessário instalar estaticamente o BouncyCastle na JVM atual. Para aplicativos em execução fora do contêiner, o provedor de segurança pode ser disponibilizado para o JBossWS adicionando uma biblioteca BouncyCastle ao caminho da classe.

Você pode desativar esse comportamento definindo **org.jboss.ws.cxf.noLocalBC** valor da propriedade para **true** no **jaxws-endpoint-config.xml** arquivo descritor de implementação para o servidor ou o **jaxws-client-config.xml** arquivo descritor para clientes.

Se você quiser uma versão diferente daquela que é enviada com o JBoss EAP, você ainda pode instalar estaticamente o BouncyCastle para a JVM. Neste caso, o provedor de segurança BouncyCastle é

escolhido entre o provedor presente no caminho da classe. Para evitar quaisquer problemas, você deve usar BouncyCastle 1.49, 1.51 ou versão superior.

5.1.6. Estratégia de seleção de barramento Apache CXF [Esta é uma tradução automática]

A estratégia de seleção de barramento padrão para clientes em execução no container foi alterada de **THREAD_BUS** para **TCCL_BUS**. Para clientes que estão ficando sem contêiner, a estratégia padrão ainda é **THREAD_BUS**. Você pode restaurar o comportamento para o release anterior usando um dos seguintes métodos.

- Inicialize o servidor do JBoss EAP com a propriedade do sistema **org.jboss.ws.cxf.jaxws-client.bus.strategy** valor definido como **THREAD_BUS**.
- Defina explicitamente a estratégia de seleção no código do cliente.

5.1.7. Requisitos JAX-WS 2.2 para WebServiceRef [Esta é uma tradução automática]

Os contêineres devem usar os construtores de estilo JAX-WS 2.2, que incluem o [WebServiceFeature](#) class como um argumento no construtor, para construir clientes que são injetados em referências de serviço da web. O JBoss EAP 6.4, que acompanha o JBossWS 4, oculta esse requisito. O JBoss EAP 7 é fornecido com o JBossWS 5, que não oculta mais esse requisito. Isso significa que as classes de serviço fornecidas pelo usuário devem implementar o JAX-WS 2.2 ou posterior, atualizando o código existente para usar o [javax.xml.ws.Service](#) construtor que inclui um ou mais **WebServiceFeature** argumentos.

```
protected Service(URL wsdlDocumentLocation,
                  QName serviceName,
                  WebServiceFeature... features)
```

5.1.8. Alterações de verificação CN IgnoreHttpsHost [Esta é uma tradução automática]

Em versões anteriores, você poderia desabilitar a verificação de nome de host de URL HTTPS em relação ao Nome Comum (CN) de um serviço fornecido em seu certificado, definindo a propriedade do sistema **org.jboss.security.ignoreHttpsHost** para **true**. Este nome de propriedade do sistema foi substituído por **cxf.tls-client.disableCNCheck**.

5.1.9. Configuração lado cliente e carregamento de classe [Esta é uma tradução automática]

Como consequência da ativação de injeções em manipuladores de clientes de serviço e de terminal, não é mais possível carregar automaticamente classes do manipulador **org.jboss.as.webservices.server.integration** Módulo JBoss. Se seu aplicativo depender de uma determinada configuração predefinida, talvez seja necessário definir explicitamente novas dependências do módulo para sua implementação. Para mais informações, veja [Migrar Dependências de Módulo Explícito](#)

5.1.10. O mecanismo de substituição de standards endossados Java está descontinuado. [Esta é uma tradução automática]

o [Mecanismo de substituição de padrões endossados por Java](#) foi preterido no JDK 1.8_40 com a

intenção de removê-lo no JDK 9. Esse mecanismo permitiu que os desenvolvedores disponibilizassem as bibliotecas para todos os aplicativos implementados, colocando JARs em um diretório endossado dentro do JRE.

Se seu aplicativo utiliza implementações JBossWS de Apache CXF, o JBoss EAP 7 assegura que as dependências necessárias sejam adicionadas na ordem correta e esta alteração não deve causar impacto para você. Se seu aplicativo acessa Apache CXF diretamente, você deve fornecer agora as dependências Apache CXF depois das dependências JBossWS como parte da implantação de seu aplicativo.

5.1.11. Especificação de descritor no arquivo EAR [Esta é uma tradução automática]

Nas versões anteriores do JBoss EAP, você poderia configurar o **jboss-webservices.xml** arquivo descritor de implementação para implantações de serviços da Web EJB no **META-INF/** diretório de arquivos JAR ou no **WEB-INF/** diretório para implementações de serviço da web POJO e terminais de serviço da Web EJB agrupados em arquivos WAR.

No JBoss EAP 7, agora você pode configurar o **jboss-webservices.xml** arquivo descritor de implementação no **META-INF/** diretório de um arquivo EAR. Se um **jboss-webservices.xml** arquivo é encontrado no arquivo EAR e no arquivo JAR ou WAR, os dados de configuração no arquivo **jboss-webservices.xml** O arquivo para o JAR ou WAR substitui os dados correspondentes no arquivo do descritor EAR.

5.2. ATUALIZE A PORTA E CONECTOR URL REMOTO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

No JBoss EAP 7, o conector padrão foi alterado de **remote** para **http-remoting** e a porta de conexão remota padrão foi alterada de **4447** para **8080**. A URL do provedor JNDI para a configuração padrão foi alterada de **remote://localhost:4447** para **http-remoting://localhost:8080**.

Se você usa o JBoss EAP 7 **migrate** operação para atualizar sua configuração, você não precisa modificar o conector remoto, a porta remota ou as URLs do provedor JNDI porque a operação de migração preserva o conector de comunicação remota do JBoss EAP 6 e **4447** configurações de porta na configuração do subsistema. Para mais informações sobre o **migrate** operação, consulte [Operação de Migração do CLI de Gerenciamento](#).

Se você não usar o **migrate** operação e, em vez disso, executar com a nova configuração padrão do JBoss EAP 7, você deve alterar o conector remoto, porta remota e URL do provedor de JNDI para usar as novas configurações conforme descrito acima.

5.3. ALTERAÇÕES DE APLICATIVOS DE MENSAGENS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

5.3.1. Substituir ou atualizar descritores de implantação JMS [Esta é uma tradução automática]

Os arquivos proprietários do descritor de implementação de recursos do HornetQ JMS identificados pelo padrão de nomenclatura -**jms.xml** não funciona mais no JBoss EAP 7. A seguir, um exemplo de um arquivo descritor de implantação de recurso JMS no JBoss EAP 6.

```
<?xml version="1.0" encoding="UTF-8"?>
```



```

<messaging-deployment xmlns="urn:jboss:messaging-deployment:1.0">
  <hornetq-server>
    <jms-destinations>
      <jms-queue name="testQueue">
        <entry name="queue/test"/>
        <entry name="java:jboss/exported/jms/queue/test"/>
      </jms-queue>
      <jms-topic name="testTopic">
        <entry name="topic/test"/>
        <entry name="java:jboss/exported/jms/topic/test"/>
      </jms-topic>
    </jms-destinations>
  </hornetq-server>
</messaging-deployment>

```

Se você usou `-jms.xml` Descritores de implementação JMS em seu aplicativo no release anterior, você pode converter seu aplicativo para usar o descritor de implementação padrão do Java EE 7, conforme especificado na seção EE.5.18 do [Especificação Java EE 7](#) ou você pode atualizar o descritor de implantação para usar o **messaging-activemq-deployment** esquema em vez disso.

Se você escolher atualizar o descritor, você precisa fazer as seguintes modificações.

- Altere o espaço de nomes de "urn:jboss:messaging-deployment:1.0" para "urn:jboss:messaging-activemq-deployment:1.0".
- Mudar o **<hornetq-server>** nome do elemento para **<server>**.

O arquivo modificado deve parecer com o exemplo que segue.

```

<?xml version="1.0" encoding="UTF-8"?>
<messaging-deployment xmlns="urn:jboss:messaging-activemq-deployment:1.0">
  <server>
    <jms-destinations>
      <jms-queue name="testQueue">
        <entry name="queue/test"/>
        <entry name="java:jboss/exported/jms/queue/test"/>
      </jms-queue>
      <jms-topic name="testTopic">
        <entry name="topic/test"/>
        <entry name="java:jboss/exported/jms/topic/test"/>
      </jms-topic>
    </jms-destinations>
  </server>
</messaging-deployment>

```

Para obter informações sobre alterações de configuração do servidor relacionadas a mensagens, consulte [Alterações na configuração do servidor de mensagens](#).

5.3.2. Atualizar clientes externo JMS [Esta é uma tradução automática]

JBoss EAP 7 ainda suporta API JMS 1.1, assim você não precisa modificar seu código.

O conector e a porta remotos padrão foram alterados no JBoss EAP 7. Para obter detalhes sobre essa alteração, consulte [Atualizar o conector e porta de URL remoto](#).

Se você migrar a configuração do servidor usando o **migrate** operação, as configurações antigas são

preservadas e você não precisa atualizar **PROVIDER_URL**. No entanto, se você executar com a nova configuração padrão do JBoss EAP 7, você deve alterar **PROVIDER_URL** no código do cliente para usar o novo **http-remoting://localhost:8080** configuração. Para mais informações, veja [Migrar o código do cliente de nomeação remota](#).

Se você planeja migrar seu código para usar a API do JMS 2.0, consulte **helloworld-jms** início rápido para um exemplo de trabalho.

5.3.3. Substitua a API HornetQ [Esta é uma tradução automática]

O JBoss EAP 6 incluiu o **org.hornetq** módulo, o que lhe permitiu usar o [API do HornetQ](#) no código-fonte da sua aplicação.

Apache ActiveMQ Artemis substituiu o HornetQ no JBoss EAP 7, então você deve migrar qualquer código que use a API do HornetQ para usar o [Apache ActiveMQ Artemis API](#). As bibliotecas para esta API estão incluídas no **org.apache.activemq.artemis** módulo.

ActiveMQ Artemis é uma evolução do HornetQ, por isso muitos conceitos ainda são válidos.

5.4. ALTERAÇÕES DE APLICATIVOS JAX-RS E RESTEasy [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

O JBoss EAP 6 incluiu o RESTEasy 2, que foi uma implementação do JAX-RS 1.x.

O JBoss EAP 7.0 inclui o RESTEasy 3, que é uma implementação do JAX-RS 2.0, conforme definido pelo [JSR 339: JAX-RS 2.0: A API Java para serviços da Web RESTful](#) especificação. Para obter mais informações sobre a API Java para serviços Web RESTful, consulte o [Especificação da API JAX-RS 2.0](#).

A versão do Jackson incluída no JBoss EAP mudou. O JBoss EAP 6 incluiu o Jackson 1.9.9. O JBoss EAP 7 e posterior agora inclui o Jackson 2.6.3 ou superior.

Esta seção descreve como essas alterações podem afetar aplicativos que usam RESTEasy ou JAX-RS.

5.4.1. Classes preteridas de RESTEasy [Esta é uma tradução automática]

Classes de corpo de mensagem e interceptores

[JSR 311: JAX-RS: A API Java™ para serviços da Web RESTful](#) não incluiu um framework interceptador, então o RESTEasy 2 forneceu um. [JSR 339: JAX-RS 2.0: A API Java para serviços da Web RESTful](#) introduziu uma estrutura oficial de interceptador e filtro, de modo que a estrutura do interceptador incluída no RESTEasy 2 está obsoleta e é substituída pela instalação do interceptor compatível com JAX-RS 2.0 no RESTEasy 3.x. As interfaces relevantes são definidas no **javax.ws.rs.ext** pacote do **jaxrs-api** módulo.

- As interfaces de interceptores seguintes são preteridas no RESTEasy 3.x.
 - [org.jboss.resteasy.spi.interception.PreProcessInterceptor](#)
 - [org.jboss.resteasy.spi.interception.PostProcessInterceptor](#)
 - [org.jboss.resteasy.spi.interception.ClientExecutionInterceptor](#)
 - [org.jboss.resteasy.spi.interception.ClientExecutionContext](#)
 - [org.jboss.resteasy.spi.interception.AcceptedByMethod](#)

- o `org.jboss.resteasy.spi.interception.PreProcessInterceptor` interface é substituída pela `javax.ws.rs.container.ContainerRequestFilter` interface no RESTEasy 3.x.
- As seguintes interfaces e classes também foram preteridas no RESTEasy 3.x.
 - o `org.jboss.resteasy.spi.interception.MessageBodyReaderInterceptor`
 - o `org.jboss.resteasy.spi.interception.MessageBodyWriterInterceptor`
 - o `org.jboss.resteasy.spi.interception.MessageBodyWriterContext`
 - o `org.jboss.resteasy.spi.interception.MessageBodyReaderContext`
 - o `org.jboss.resteasy.core.interception.InterceptorRegistry`
 - o `org.jboss.resteasy.core.interception.InterceptorRegistryListener`
 - o `org.jboss.resteasy.core.interception.ClientExecutionContextImpl`
- o `org.jboss.resteasy.spi.interception.MessageBodyWriterInterceptor` interface é substituída pela `javax.ws.rs.ext.WriterInterceptor` interface.
- Além disso, algumas mudanças no `javax.ws.rs.ext.MessageBodyWriter` interface pode não ser compatível com o JAX-RS 1.x. Se seu aplicativo usou o JAX-RS 1.x, revise o código do aplicativo para certificar-se de definir `@Produces` ou `@Consumes` para seus endpoints. Não fazer isso pode resultar em um erro semelhante ao seguinte.

```
org.jboss.resteasy.core.NoMessageBodyWriterFoundFailure: Could not
find MessageBodyWriter for response object of type: <OBJECT> of
media type:
```

Segue um exemplo de um ponto de extremidade REST que pode causar este erro.

```
@Path("dates")
public class DateService {

    @GET
    @Path("daysuntil/{targetdate}")
    public long showDaysUntil(@PathParam("targetdate") String
targetDate) {
        DateLogger.LOGGER.logDaysUntilRequest(targetDate);
        final long days;

        try {
            final LocalDate date = LocalDate.parse(targetDate,
DateTimeFormatter.ISO_DATE);
            days = ChronoUnit.DAYS.between(LocalDate.now(), date);
        } catch (DateTimeParseException ex) {
            // ** DISCLAIMER **. This example is contrived.
            throw new
WebApplicationException(Response.status(400).entity(ex.getLocalizedMessage())
.type(MediaType.TEXT_PLAIN)
                .build());
        }
    }
}
```

```

        return days;
    }
}

```

Para corrigir o problema, adicione a importação para `javax.ws.rs.Produces` e a `@Produces` anotação da seguinte forma.

```

...
import javax.ws.rs.Produces;
...

@Path("/dates")
public class DateService {

    @GET
    @Path("/daysuntil/{targetdate}")
    @Produces(MediaType.TEXT_PLAIN)
    public long showDaysUntil(@PathParam("targetdate") String
targetDate) {
        DateLogger.LOGGER.logDaysUntilRequest(targetDate);
        final long days;

        try {
            final LocalDate date = LocalDate.parse(targetDate,
DateTimeFormatter.ISO_DATE);
            days = ChronoUnit.DAYS.between(LocalDate.now(), date);
        } catch (DateTimeParseException ex) {
            // ** DISCLAIMER **. This example is contrived.
            throw new
WebApplicationException(Response.status(400).entity(ex.getLocalizedMessage())
.type(MediaType.TEXT_PLAIN)
                .build());
        }
        return days;
    }
}

```



NOTA

Todos interceptores de lançamentos prévios do RESTEasy podem executar em paralelo com o novo filtro JAX-RS 2.0 e interfaces de interceptor.

Para obter mais informações sobre interceptores, consulte [Interceptores RESTEasy](#) dentro *Desenvolvendo Aplicativos de Serviços da Web* para o JBoss EAP.

Para mais informações sobre a nova API de substituição, consulte o [RESTEasy JAX-RS 3.0.13.Final API](#) ou mais tarde.

API Cliente

A estrutura do cliente RESTEasy em **resteasy-jaxrs** foi substituído pelo padrão JAX-RS 2.0 **resteasy-client** módulo. Como resultado, algumas classes e métodos da API do cliente RESTEasy foram descontinuados.

- As seguintes classes são preteridas.

```

... javax.ws.rs.client.ClientRequest

```

- [org.jboss.resteasy.client.ClientRequest](#)
- [org.jboss.resteasy.client.ClientRequestFactory](#)
- [org.jboss.resteasy.client.ClientResponse](#)
- [org.jboss.resteasy.client.ProxyBuilder](#)
- [org.jboss.resteasy.client.ProxyConfig](#)
- [org.jboss.resteasy.client.ProxyFactory](#)
- o [org.jboss.resteasy.client.ClientResponseFailure](#) exceção e o [org.jboss.resteasy.client.ClientExecutor](#) e [org.jboss.resteasy.client.EntityTypeFactory](#) interfaces também são descontinuadas.
- Você deve substituir o [org.jboss.resteasy.client.ClientRequest](#) e [org.jboss.resteasy.client.ClientResponse](#) aulas com [org.jboss.resteasy.client.jaxrs.ResteasyClient](#) e [javax.ws.rs.core.Response](#) respectivamente.
Segue um exemplo de como enviar um link header com o cliente RESTEasy em RESTEasy 2.3.x.

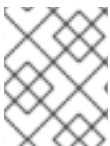
```
ClientRequest request = new
ClientRequest(generateURL("/linkheader/str"));
request.addLink("previous chapter", "previous",
"http://example.com/TheBook/chapter2", null);
ClientResponse response = request.post();
LinkHeader header = response.getLinkHeader();
```

O seguinte é um exemplo de como executar a mesma tarefa com cliente RESTEasy em RESTEasy 3.

```
ResteasyClient client = new ResteasyClientBuilder().build();
Response response =
client.target(generateURL("/linkheader/str")).request()
.header("Link", "<http://example.com/TheBook/chapter2>";
rel="previous";
title="previous chapter\"").post(Entity.text(new String()));
javax.ws.rs.core.Link link = response.getLink("previous");
```

Veja o **resteasy-jaxrs-client** início rápido para um exemplo de um cliente externo JAX-RS RESTEasy que interage com um serviço da Web JAX-RS.

- As classes e interfaces no [org.jboss.resteasy.client.cache](#) pacote também são descontinuados. Eles são substituídos por classes e interfaces equivalentes no [org.jboss.resteasy.client.jaxrs.cache](#) pacote.



NOTA

Para mais informações sobre o [org.jboss.resteasy.client.jaxrs](#) Classes de API, consulte o [RESTEasy JAX-RS JavaDoc](#).

StringConverter

o [org.jboss.resteasy.spi.StringConverter](#) A classe está obsoleta em RESTEasy 3.x. Esta funcionalidade pode ser substituída usando o JAX-RS 2.0 [jax.ws.rs.ext.ParamConverterProvider](#) classe.

5.4.2. Classes de RESTEasy removidas ou protegidas [Esta é uma tradução automática]

Métodos de adição de ResteasyProviderFactory

A maioria dos [org.jboss.resteasy.spi.ResteasyProviderFactory](#) `add()` métodos foram removidos ou protegidos no RESTEasy 3.0. Por exemplo, o `addBuiltInMessageBodyReader()` e `addBuiltInMessageBodyWriter()` métodos foram removidos e os `addMessageBodyReader()` e `addMessageBodyWriter()` métodos foram protegidos.

Você deve agora usar o `registerProvider()` e `registerProviderInstance()` métodos.

Classes suplementares removidas do RESTEasy 3

o `@org.jboss.resteasy.annotations.cache.ServerCached` anotação, que especificou a resposta para o método JAX-RS deve ser armazenada em cache no servidor, foi removida do RESTEasy 3 e deve ser removida do código do aplicativo.

5.4.3. Outras alterações de RESTEasy [Esta é uma tradução automática]

SignedInput e SignedOutput

- **SignedInput** e **SignedOutput** para **resteasy-crypto** deve ter o **Content-Type** definido como **multipart/signed** tanto no **Request** ou **Response** objeto, ou usando o **@Consumes** ou **@Produces** anotação.
- **SignedOutput** e **SignedInput** pode ser usado para retornar **application/pkcs7-signature** Formato do tipo MIME na forma binária, definindo esse tipo na **@Produces** ou **@Consumes** anotações.
- Se o **@Produces** ou **@Consumes** é **text/plain** Tipo MIME, **SignedOutput** será codificado em base64 e enviado como uma String.

Alterações WS-Security

Os filtros de segurança para **@RolesAllowed**, **@PermitAll**, e **@DenyAll** agora retorne "403 Proibido" em vez de "401 Não Autorizado".

Filtros do lado do cliente

Os novos filtros JAX-RS 2.0 lado cliente não serão limitados e executarão quando você estiver usando a API de cliente RESTEasy a partir de um lançamento prévio.

Suporte HTTP assíncrono

Como a especificação JAX-RS 2.0 adiciona suporte HTTP assíncrono usando o **@Suspended** anotação e o **AsynResponse** interface, a API proprietária do RESTEasy para HTTP assíncrono foi preterida e pode ser removida em uma versão futura do RESTEasy. Os módulos assíncronos Tomcat e JBoss Web assíncronos também foram removidos da instalação do servidor. Se você não estiver usando o contêiner Servlet 3.0 ou superior, o processamento do lado do servidor HTTP assíncrono será simulado e executado de forma síncrona no mesmo encadeamento de solicitação.

Cache do lado do servidor

A configuração do cache do lado do servidor foi alterada. por favor veja o [Documentação RESTEasy](#) Para maiores informações.

Alterações do provedor Jackson [Esta é uma tradução automática]

Nas versões anteriores do JBoss EAP, a configuração do provedor RESTEasy YAML era ativada por padrão. Isso mudou no JBoss EAP 7. O provedor YAML agora está desabilitado por padrão. Seu uso não é suportado devido a um problema de segurança no **SnakeYAML** biblioteca usada pelo RESTEasy para unmarshalling e deve ser ativada explicitamente no aplicativo. Para obter informações sobre como habilitar o provedor YAML em seu aplicativo e adicionar as dependências do Maven, consulte [Provedor YAML](#) dentro *Desenvolvendo Aplicativos de Serviços da Web* para o JBoss EAP.

Charset padrão UTF-8 no cabeçalho Content-Type

No JBoss EAP 7.1, o `resteasy.add.charset` parâmetro está definido como `true` por padrão. Você pode definir o `resteasy.add.charset` parâmetro para `false` se você não quiser que o RESTEasy adicione `charset=UTF-8` para o cabeçalho de tipo de conteúdo retornado quando o método de recurso retorna um `text/*` ou `application/xml*` tipo de mídia sem um conjunto de caracteres explícito.

Para obter mais informações sobre tipos de mídia de texto e conjuntos de caracteres, consulte [Tipos de mídia de texto e conjuntos de caracteres](#) dentro *Desenvolvendo Aplicativos de Serviços da Web* para o JBoss EAP.

SerializableProvider

A desserialização de objetos Java de fontes não confiáveis não é segura. Por esta razão, no JBoss EAP 7, o `org.jboss.resteasy.plugins.providers.SerializableProvider` A classe está desabilitada por padrão e não é recomendável usar esse provedor.

Solicitações correspondentes aos métodos de recursos

No RESTEasy 3, melhorias e correções foram feitas na implementação de regras de correspondência, conforme definido na especificação JAX-RS 2.0. Em particular, foi feita uma alteração na forma como um URI ambíguo em um método de sub-recurso e um localizador de sub-recurso é tratado.

No RESTEasy 2, era possível que um localizador de sub-recursos fosse executado com sucesso mesmo quando havia outro sub-recurso com o mesmo URI. Esse comportamento estava incorreto de acordo com a especificação.

No RESTEasy 3, quando houver um URI ambíguo para um sub-recurso e um localizador de sub-recurso, chamar o sub-recurso será bem-sucedido; no entanto, chamar o localizador de sub-recurso resultará em um status HTTP **405 Method Not Allowed** erro.

O exemplo a seguir contém um erro ambíguo `@Path` anotação em um método de sub-recurso e um localizador de sub-recurso. Observe que o URI para ambos os endpoints, `anotherResource` e `anotherResourceLocator`, é o mesmo. A diferença entre os dois pontos finais é que o `anotherResource` método está associado com o verbo REST, **POST**. o `anotherResourceLocator` O método não está associado a nenhum verbo REST. De acordo com a especificação, o terminal com o verbo REST, neste caso, o `anotherResource` método, sempre será selecionado.

```
@Path("myResource")
public class ExampleSubResources {
    @POST
    @Path("items")
    @Produces("text/plain")
    public Response anotherResource(String text) {
        return Response.ok("ok").build();
    }

    @Path("items")
    @Produces("text/plain")
    public SubResource anotherResourceLocator() {
```

```

    return new SubResource();
}
}

```

5.4.4. Alterações SPI RESTEasy [Esta é uma tradução automática]

Exceções SPI

Todas exceções de falhas SPI foram preteridas e não são mais utilizadas internamente. Elas foram substituídas com a exceção JAX-RS 2.0 correspondente.

Exceção preterida	Exceção de substituição em <code>jaxrs-api</code> módulo
<code>org.jboss.resteasy.spi.ForbiddenException</code>	<code>javax.ws.rs.ForbiddenException</code>
<code>org.jboss.resteasy.spi.MethodNotAllowedException</code>	<code>javax.ws.rs.NotAllowedException</code>
<code>org.jboss.resteasy.spi.NotAcceptableException</code>	<code>javax.ws.rs.NotAcceptableException</code>
<code>org.jboss.resteasy.spi.NotFoundException</code>	<code>javax.ws.rs.NotFoundException</code>
<code>org.jboss.resteasy.spi.UnauthorizedException</code>	<code>javax.ws.rs.NotAuthorizedException</code>
<code>org.jboss.resteasy.spi.UnsupportedMediaTypeException</code>	<code>javax.ws.rs.NotSupportedException</code>

InjectorFactory e Registro

o **InjectorFactory** e **Registry** SPIs mudaram. Isso não deve ser um problema se você usar o RESTEasy conforme documentado e suportado.

5.4.5. Alterações do provedor Jackson [Esta é uma tradução automática]

A versão do Jackson incluída no JBoss EAP mudou. A versão anterior do JBoss EAP incluía o Jackson 1.9.9. O JBoss EAP 7 agora inclui o Jackson 2.6.3 ou superior. Como resultado, o provedor de Jackson mudou de **resteasy-jackson-provider** para **resteasy-jackson2-provider**.

A atualização para o **resteasy-jackson2-provider** requer algumas alterações no pacote. Por exemplo, o pacote de anotações de Jackson mudou de **org.codehaus.jackson.annotate** para **com.fasterxml.jackson.annotation**.

Para mudar seu aplicativo para usar o provedor padrão que foi incluído na versão anterior do JBoss EAP, veja [Mudando o provedor Jackson padrão](#) dentro *Desenvolvendo Aplicativos de Serviços da Web* para o JBoss EAP.

5.4.6. Alterações à integração de Spring RESTEasy [Esta é uma tradução automática]

A framework de Spring 4.0 introduziu suporte para Java 8. Se você planeja usar integração RESTEasy 3.x com Spring, certifique-se que especificou 4.2.x como a versão mínima para Spring em sua implantação pois esta é a versão estável mais recente suportada pelo JBoss EAP 7.

5.4.7. Alterações ao fornecedor RESTEasy Jettison JSON [Esta é uma tradução automática]

O provedor JSON do RESTEasy Jettison está obsoleto no JBoss EAP 7 e não é mais adicionado às implementações por padrão. Você é encorajado a mudar para o fornecedor recomendado do RESTEasy Jackson. Se preferir continuar a usar o provedor Jettison, você deve definir uma dependência explícita para ele no `jboss-deployment-descriptor.xml` arquivo conforme demonstrado no exemplo a seguir.

```
<?xml version="1.0" encoding="UTF-8"?>
<jboss-deployment-structure>
  <deployment>
    <exclusions>
      <module name="org.jboss.resteasy.resteasy-jackson2-provider"/>
      <module name="org.jboss.resteasy.resteasy-jackson-provider"/>
    </exclusions>
    <dependencies>
      <module name="org.jboss.resteasy.resteasy-jettison-provider"
services="import"/>
    </dependencies>
  </deployment>
</jboss-deployment-structure>
```

Para obter mais informações sobre como definir dependências explícitas, consulte [Adicionar uma dependência de módulo explícita a uma implantação](#) no JBoss EAP *Guia de desenvolvimento*.

5.5. ALTERAÇÕES AOS APLICATIVOS CDI 1.2 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

O JBoss EAP 7 inclui suporte para o CDI 1.2. Como resultado, os aplicativos escritos usando o CDI 1.0 podem ver algumas mudanças no comportamento quando migrados para o JBoss EAP 7. Esta seção resume apenas algumas dessas mudanças.

Você pode encontrar mais informações sobre Weld e CDI 1.2 nas seguintes referências:

- [Contextos e Injeção de Dependência para a plataforma Java EE](#)
- [CDI 1.2 Javadoc](#)
- [Weld 2.3.3.Final - Implementação da Referência do CDI](#)

Arquivos Bean

As classes Bean de beans ativos devem ser implantadas nos arquivos bean para certificarmos de que estão escaneados pelo CDI para achar e processar as classes bean.

No CDI 1.0, um arquivo foi definido como um *explícito* arquivo de feijão se continha um `beans.xml` arquivo no `META-INF/` diretório para um aplicativo cliente, EJB ou biblioteca JAR, ou se continha `beans.xml` arquivo no `WEB-INF/` diretório para um WAR.

CDI 1.1 introduziu *implícito* arquivos de bean, que são arquivos que contêm uma ou mais classes de bean com uma anotação de definição de bean ou um ou mais beans de sessão. Os arquivos de beans implícitos são verificados pelo CDI e, durante a descoberta de tipos, somente classes com anotações de definição de bean são descobertas. Para mais informações, veja [Tipo e Descoberta de Feijão](#) dentro *Contextos e Injeção de Dependência para a plataforma Java EE*.

Um arquivo de beans possui um modo de descoberta de **all**, **annotated** ou **none**. Um arquivo de bean que contém um **beans.xml** arquivo sem versão tem um modo de descoberta de bean padrão **all**. Um arquivo de bean que contém um **beans.xml** arquivo com versão **1.1** ou mais tarde deve especificar o **bean-discovery-mode** atributo. O valor padrão para o atributo é **annotated**.

Um arquivo não é um arquivo bean nos seguintes casos:

- Ele contém um **beans.xml** arquivo com um **bean-discovery-mode** do **none**.
- Ele contém uma extensão CDI sem **beans.xml** Arquivo.

Um arquivo é um *explícito* arquivo de feijão nos seguintes casos:

- O arquivo contém um **beans.xml** arquivo com um número de versão de 1.1 ou posterior e um **bean-discovery-mode** do **all**.
- O arquivo contém um **beans.xml** arquivo sem número de versão.
- Exemplo: Bandeira de Anotação no **jboss-deployment-structure.xml** Arquivo [Esta é uma tradução automática]

Um arquivo é um *implícito* arquivo de feijão nos seguintes casos:

- O arquivo contém uma ou mais classes de bean com uma anotação de definição de bean ou um ou mais beans de sessão, mesmo que ele não contenha **beans.xml** Arquivo.
- O arquivo contém um **beans.xml** arquivo com um **bean-discovery-mode** do **annotated**.

CDI 1.2 limitado [Anotações de definição de bean](#) para o seguinte:

- **@ApplicationScoped**, **@SessionScoped**, **@ConversationScoped**, e **@RequestScoped** anotações
- Todos os outros tipos de escopos normais.
- Exemplo: **@DateBridge** e **@CalendarBridge** Anotação [Esta é uma tradução automática]
- Todas as anotações de estereótipo, que são anotações anotadas com **@Stereotype**
- Exemplo: **@IndexedEmbedded** Anotação [Esta é uma tradução automática]

Para obter mais informações sobre arquivos de bean, consulte [Arquivos de Feijão](#) dentro *Contextos e Injeção de Dependência para a plataforma Java EE*.

Esclarecimento de resolução de conversação

O ciclo de vida do contexto de conversação foi alterado para evitar conflitos com a especificação Servlet, conforme descrito em [Edição de Especificação CDI CDI-411](#). O escopo de conversação está ativo durante todas as solicitações de servlet e não deve impedir que outros servlets ou filtros de servlet definam o corpo do pedido ou a codificação de caracteres.

Resolução por observação

A resolução de evento foi parcialmente reescrita em CDI 1.2. Em CDI 1.0, um evento é entregue a um método de observação se o método de observação possui todos os qualificadores de evento. Em CDI 1.2, um evento é entregue a um método de observação se o método de observação não possui qualificadores de evento ou possui um subconjunto dos qualificadores de evento.

5.6. MIGRAR AS DEPENDÊNCIAS DE MÓDULO EXPLÍCITO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

A introdução do sistema modular de carregamento de classes e dos JBoss Modules na versão anterior do JBoss EAP permitiu um controle refinado das classes disponíveis para os aplicativos. Esse recurso permitia que você configurasse dependências explícitas do módulo usando o aplicativo **MANIFEST.MF** arquivo ou o **jboss-deployment-structure.xml** arquivo descritor de implantação.

Se você definiu dependências de módulo explícito em seu aplicativo, você deve estar ciente das seguintes alterações no JBoss EAP 7.

Verifique dependências para disponibilidade

Os módulos que estão incluídos no JBoss EAP foram alterados. Quando você migrar seu aplicativo para o JBoss EAP 7, revise seu **MANIFEST.MF** e **jboss-deployment-structure.xml** entradas de arquivo para certificar-se de que eles não se referem a nenhum módulo que foi removido deste release do produto.

Dependências que requerem scan de anotação

No release anterior do JBoss EAP, se sua dependência continha anotações que precisavam ser processadas durante a verificação de anotação, como ao declarar EJB Interceptors, era necessário gerar e incluir um índice Jandex em um novo arquivo JAR e, em seguida, definir um sinalizador a **MANIFEST.MF** ou **jboss-deployment-structure.xml** arquivo descritor de implantação.

O JBoss EAP 7 agora fornece geração automática de índices de anotação em tempo de execução para módulos estáticos, assim você não precisa mais gerá-los manualmente. No entanto, você ainda precisa adicionar **annotations** bandeira para o aplicativo **MANIFEST.MF** arquivo ou o **jboss-deployment-structure.xml** arquivo descritor de implantação, conforme demonstrado abaixo.

Exemplo: Bandeira de Anotação no MANIFEST.MF Arquivo [Esta é uma tradução automática]

```
Dependências: com.company.my-ejb annotations, com.company.other
```

Exemplo: Bandeira de Anotação no jboss-deployment-structure.xml Arquivo [Esta é uma tradução automática]

```
<jboss-deployment-structure>
  <deployment>
    <dependencies>
      <module name="com.company.my-ejb" annotations="true"/>
      <module name="com.company.other"/>
    </dependencies>
  </deployment>
</jboss-deployment-structure>
```

5.7. ALTERAÇÕES DE MIGRAÇÃO JPA E HIBERNATE [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

5.7.1. Hibernate ORM 3.0 [Esta é uma tradução automática]

As classes de integração que facilitaram a utilização do Hibernate ORM 3 em lançamentos prévios foram removidas do JBoss EAP 7. Se seu aplicativo ainda utiliza bibliotecas do Hibernate ORM 3, é

altamente recomendável que você migre seu aplicativo para utilizar Hibernate ORM 5 pois Hibernate ORM 3 não funcionará mais em JBoss EAP facilmente. Se você não pode migrar para Hibernate ORM 5, você deve definir um módulo JBoss personalizado para os JARs de Hibernate ORM 3 e excluir as classes de Hibernate ORM 5 de seu aplicativo.

5.7.2. Hibernate ORM 4.0 - 4.3 [Esta é uma tradução automática]

Caso seu aplicativo necessite que o cache de segundo nível seja ativado, você deve migrar para Hibernate ORM 5.0, que é integrado com Infinispan 8.x.

As aplicações escritas com o Hibernate ORM 4.x ainda podem usar o Hibernate ORM 4.x. Você deve definir um módulo JBoss personalizado para os JARs ORM 4.x do Hibernate e excluir as classes Hibernate ORM 5 do seu aplicativo. No entanto, é altamente recomendável que você reescreva o código do aplicativo para usar o Hibernate ORM 5. Para obter informações sobre como migrar para o Hibernate ORM 5, consulte [Migrando para o Hibernate ORM 5](#).

5.7.3. Migrando para Hibernate ORM 5 [Esta é uma tradução automática]

Esta seção destaca as alterações que você precisa fazer ao migrar do Hibernate ORM versão 4.3 para a versão 5. Para obter mais informações sobre as mudanças implementadas entre o Hibernate ORM 4 e o Hibernate ORM 5, consulte [Guia de migração do Hibernate ORM 5.0](#).

Classes preteridas de RESTEasy [Esta é uma tradução automática]

As seguintes classes reprovadas foram removidas do Hibernate ORM 5:

- [org.hibernate.cfg.AnnotationConfiguration](#)
- [org.hibernate.id.TableGenerator](#)
- [org.hibernate.id.TableHiLoGenerator](#)
- [org.hibernate.id.SequenceGenerator](#)

Outras alterações às classes e pacotes

- o [org.hibernate.integrator.spi.Integrator](#) Interface alterada para dar conta de redesenho bootstrap.
- Um novo pacote [org.hibernate.engine.jdbc.env.spi](#) foi criado. Ele contém o [org.hibernate.engine.jdbc.env.spi.JdbcEnvironment](#) interface, que foi extraída do [org.hibernate.engine.jdbc.spi.JdbcServices](#) interface.
- Um novo [org.hibernate.boot.model.relational.ExportableProducer](#) interface foi introduzida que afetará [org.hibernate.id.PersistentIdentifierGenerator](#) implementações.
- A assinatura de [org.hibernate.id.Configurable](#) foi alterado para aceitar [org.hibernate.service.ServiceRegistry](#) em vez de apenas [org.hibernate.dialect.Dialect](#).
- o [org.hibernate.metamodel.spi.TypeContributor](#) interface migrou para [org.hibernate.boot.model.TypeContributor](#).
- o [org.hibernate.metamodel.spi.TypeContributions](#) interface migrou para [org.hibernate.boot.model.TypeContributions](#).

Type Handling

- Construídas em `org.hibernate.type.descriptor.sql.SqlTypeDescriptor` implementações não mais se registram automaticamente `org.hibernate.type.descriptor.sql.SqlTypeDescriptorRegistry`. Aplicativos usando personalizado `SqlTypeDescriptor` implementações que estendem as implementações internas e dependem desse comportamento devem ser atualizadas para chamar `SqlTypeDescriptorRegistry.addDescriptor()` si mesmos.
- Para IDs definidos como UUIDs gerados, alguns bancos de dados exigem que você defina explicitamente `@Column(length=16)` para gerar `BINARY(16)` para que as comparações funcionem corretamente.
- Para `EnumType` mapeamentos definidos no `hbm.xml`, onde você quer `javax.persistence.EnumType.STRING` `name-mapping`, esta configuração deve ser declarada explicitamente usando o `useNamed(true)` configuração ou especificando um `VARCHAR` valor de `12`.

Alterações do subsistema de transações [Esta é uma tradução automática]

- A transação SPI passou por uma grande reformulação no Hibernate ORM 5. No Hibernate ORM 4.3, você usou o `org.hibernate.Transaction` API para acessar diretamente diferentes estratégias de transação de back-end. O Hibernate ORM 5 introduziu um nível de indireção. No back-end, o `org.hibernate.Transaction` implementação agora fala com um `org.hibernate.resource.transaction.TransactionCoordinator`, que representa o contexto transacional para uma determinada sessão de acordo com a estratégia de backend. Embora isso não tenha um impacto direto nos desenvolvedores, isso pode afetar a configuração do bootstrap. Anteriormente, os aplicativos especificariam `hibernate.transaction.factory_class` propriedade, que agora está obsoleta, e se refere a um `org.hibernate.engine.transaction.spi.TransactionFactory` FQN (nome totalmente qualificado). Com o Hibernate ORM 5, você especifica o `hibernate.transaction.coordinator_class` configuração e consulte um `org.hibernate.resource.transaction.TransactionCoordinatorBuilder`. Veja `org.hibernate.cfg.AvailableSettings.TRANSACTION_COORDINATOR_STRATEGY` para detalhes adicionais.
- Os nomes abreviados a seguir agora são reconhecidos:
 - **jdbc**: Gerenciar transações usando o JDBC `java.sql.Connection`. Este é o padrão para transações não-JPA.
 - **jta**: Gerenciar transações usando o JTA.



IMPORTANTE

Se um aplicativo JPA não fornecer uma configuração para o **hibernate.transaction.coordinator_class** propriedade, o Hibernate irá construir automaticamente o coordenador de transação apropriado com base no tipo de transação para a unidade de persistência.

Se um aplicativo não-JPA não fornecer uma configuração para o **hibernate.transaction.coordinator_class** propriedade, o Hibernate será o padrão **jdbc** para gerenciar as transações. Esse padrão causará problemas se o aplicativo realmente usar transações baseadas em JTA. Um aplicativo não-JPA que usa transações baseadas em JTA deve definir explicitamente **hibernate.transaction.coordinator_class** valor da propriedade para **jta** ou fornecer um costume **org.hibernate.resource.transaction.TransactionCoordinatorBuilder** que constrói um **org.hibernate.resource.transaction.TransactionCoordinator** que coordena corretamente com transações baseadas em JTA.

Alterações no Hibernate Search [Esta é uma tradução automática]

- o **cfg.xml** os arquivos são novamente totalmente analisados e integrados a eventos, segurança e outras funções.
- As propriedades carregadas do **cfg.xml** usando o **EntityManagerFactory** não prefixou anteriormente nomes com **hibernate**. Isso agora se tornou consistente.
- A configuração não é serializável.
- o **org.hibernate.dialect.Dialect.getQuerySequencesString()** O método agora recupera valores de catálogo, esquema e incremento.
- o **AuditConfiguration** modificador foi removido de **org.hibernate.envers.boot.internal.EnversService**.
- o **AuditStrategy** parâmetros do método foram alterados para remover o obsoleto **AuditConfiguration** e usar o novo **EnversService**.
- Várias classes e interfaces no **org.hibernate.hql.spi** pacote e subpacotes foram movidos para o novo **org.hibernate.hql.spi.id** pacote. Isso inclui o **MultiTableBulkIdStrategy** classe e o **AbstractTableBasedBulkIdHandler**, **TableBasedDeleteHandlerImpl**, e **TableBasedUpdateHandlerImpl** interfaces e suas subclasses.
- Houve um redesenho completo de contratos de acesso a propriedade.
- Válido **hibernate.cache.default_cache_concurrency_strategy** valores de configuração são agora definidos usando o **org.hibernate.cache.spi.access.AccessType.getExternalName()** método em vez do **org.hibernate.cache.spi.access.AccessType** constantes de enum. Isso é mais consistente com outras configurações do Hibernate.

5.7.4. Migrando do Hibernate ORM 5.0 para o Hibernate ORM 5.1 [Esta é uma tradução automática]

Enquanto o JBoss EAP 7.0 incluía o Hibernate ORM 5.0, o JBoss EAP 7.1 agora inclui o Hibernate ORM 5.1. Esta seção destaca as diferenças e as mudanças necessárias ao migrar do Hibernate ORM versão 5.0 para a versão 5.1.

Hibernate ORM 3.0 [Esta é uma tradução automática]

Esta versão inclui muitas melhorias de desempenho e correções de bugs, que são detalhadas em [Recursos do Hibernate ORM 5.1](#) no JBoss EAP *Notas de versão 7.1.0*. Para informações adicionais sobre as mudanças implementadas entre o Hibernate ORM 5.0 eo Hibernate ORM 5.1, veja o [Hibernate ORM 5.1 Guia de Migração](#).

Mudanças no gerenciamento de JMX [Esta é uma tradução automática]

Mudanças no Cliente EJB no JBoss EAP 7 [Esta é uma tradução automática]

As mudanças de ferramentas de gerenciamento de esquema no Hibernate ORM 5.1 são principalmente focadas nas seguintes áreas:

- Unificando o manuseio de **hbm2ddl.auto** e o JPA do Hibernate **schema-generation** Apoio, suporte.
- Removendo as preocupações do JDBC do SPI para facilitar a substituição real do Hibernate OGM, um mecanismo de persistência que fornece suporte ao Java Persistence (JPA) para armazenamentos de dados NoSQL.

As alterações de ferramentas de gerenciamento de esquema devem ser apenas uma preocupação de migração para aplicativos que usam diretamente qualquer uma das seguintes classes:

- **org.hibernate.tool.hbm2ddl.SchemaExport**
- **org.hibernate.tool.hbm2ddl.SchemaUpdate**
- **org.hibernate.tool.hbm2ddl.SchemaValidator**
- **org.hibernate.tool.schema.spi.SchemaManagementTool** ou qualquer um dos seus delegados

Mudanças de Configuração de Mensagens no JBoss EAP 7.1 [Esta é uma tradução automática]

O Hibernate ORM 5.1.10, incluído no JBoss EAP 7.1, introduziu uma nova estratégia para recuperar tabelas de banco de dados que melhoram **SchemaMigrator** e **SchemaValidator** desempenho. Esta estratégia executa uma única **java.sql.DatabaseMetaData#getTables(String, String, String, String[])** ligue para determinar se cada **javax.persistence.Entity** tem uma tabela de banco de dados mapeada. Esta é a estratégia padrão, e usa o **hibernate.hbm2ddl.jdbc_metadata_extraction_strategy=grouped** configuração de propriedade. Essa estratégia pode exigir **hibernate.default_schema** e / ou **hibernate.default_catalog** ser provido.

Para usar a estratégia antiga, que executa uma **java.sql.DatabaseMetaData#getTables(String, String, String, String[])** chamar para **each javax.persistence.Entity**, use o **hibernate.hbm2ddl.jdbc_metadata_extraction_strategy=individually** configuração de propriedade.

5.8. ALTERAÇÕES NO HIBERNATE SEARCH [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

A versão do Hibernate Search fornecida com JBoss EAP 7 foi alterada. O lançamento prévio do JBoss EAP fornecia Hibernate Search 4.6.x. O JBoss EAP 7 fornece Hibernate Search 5.5.x.

O Hibernate Search 5.5 é construído sobre o Apache Lucene 5.3.1. Se você usa APIs nativas do Lucene, lembre-se de alinhar com essa versão. o [Hibernate Search 5.5 API](#) envolve e oculta a complexidade de muitas das alterações da API do Lucene feitas entre a versão 3 e a versão 5; No entanto, algumas classes agora são reprovadas, renomeadas ou reempacotadas. Esta seção descreve como essas alterações podem afetar o código do seu aplicativo.

Alterações no Hibernate Search [Esta é uma tradução automática]

Indexing of id Fields of Embedded Relations

Ao usar um `@IndexedEmbedded` anotação para incluir campos de uma entidade relacionada, o `id` da entidade relacionada não está mais incluída. Você pode ativar a inclusão do `id` usando o `includeEmbeddedObjectId` atributo do `@IndexedEmbedded` anotação.

Exemplo: @IndexedEmbedded Anotação [Esta é uma tradução automática]

```
@IndexedEmbedded(includeEmbeddedObjectId=true)
```

Alterações de migração JPA e Hibernate [Esta é uma tradução automática]

Números e datas agora são indexados como campos numéricos por padrão. Propriedades do tipo `int`, `long`, `float`, `double`, e suas classes de wrapper correspondentes não são mais indexadas como strings. Em vez disso, eles agora são indexados usando a codificação numérica apropriada do Lucene. o `id` campos são uma exceção a essa regra. Mesmo quando eles são representados por um tipo numérico, eles ainda são indexados como uma palavra-chave de string por padrão. O uso de `@NumericField` agora está obsoleto, a menos que você queira especificar uma precisão personalizada para a codificação numérica. Você pode manter o antigo formato de índice baseado em sequência especificando explicitamente uma ponte de campo de codificação de cadeia de caracteres. No caso de inteiros, esta é a [org.hibernate.search.bridge.builtin.IntegerBridge](#). Verifica a [org.hibernate.search.bridge.builtin](#) pacote para outras pontes de campo disponíveis publicamente.

`Date` e `Calendar` não são mais indexados como strings. Em vez disso, as instâncias são codificadas como valores longos representando o número de milissegundos desde 1º de janeiro de 1970, 00:00:00 GMT. Você pode alternar o formato de indexação usando o novo [EncodingType](#) enum. Por exemplo:

Exemplo: @DateBridge e @CalendarBridge Anotação [Esta é uma tradução automática]

```
@DateBridge(encoding=EncodingType.STRING)
@CalendarBridge(encoding=EncodingType.STRING)
```

A alteração de codificação de números e datas é importante e pode ter um grande impacto no comportamento do aplicativo. Se você tiver uma consulta que segmente um campo que foi anteriormente codificado por string, mas agora está codificado numericamente, será necessário atualizar a consulta. Os campos numéricos devem ser pesquisados com um `NumericRangeQuery`. Você também deve se certificar de que todos os campos segmentados por facetação sejam codificados por string. Se você usar a consulta de pesquisa DSL, a consulta correta deverá ser criada automaticamente para você.

Alterações no Hibernate Search [Esta é uma tradução automática]

- As opções de classificação foram aprimoradas e a codificação de campo especificada incorretamente para opções de classificação agora resulta em exceções de tempo de execução. O Lucene também oferece uma classificação mais eficiente se os campos usados no tipo forem conhecidos de antemão. O Hibernate Search 5.5 fornece o novo `@SortableField` anotação e seu companheiro multi-valorizado `@SortableFields`. Veja o [Guia de Migração do Hibernate Search 5.4 to 5.5](#) Para maiores informações.

- O Lucene **SortField** API requer a seguinte alteração no código do aplicativo. No lançamento prévio do JBoss EAP, você definia o tipo do campo de classificação na consulta conforme segue.

```
fulltextQuery.setSort(new Sort(new SortField("title",
SortField.STRING)));
```

O seguinte é um exemplo de como definir no JBoss EAP 7.

```
fulltextQuery.setSort(new Sort(new SortField("title",
SortField.Type.STRING)));
```

- Desde a **SearchFactory** só deve ser usado pela integração ORM, foi movido do **hibernate-search-engine** módulo para o **hibernate-search-orm** módulo. Outros integradores devem depender exclusivamente **SearchIntegrator**, que substitui o obsoleto **SearchFactoryIntegrator**.
- O valor enum **SpatialMode.GRID** foi renomeado para **SpatialMode.HASH**.
- **FullTextIndexEventListener** agora é uma aula final. Se você atualmente estende essa classe, você deve encontrar uma solução alternativa para obter a mesma funcionalidade.
- o **hibernate-search-analyzers** módulo foi removido. A abordagem recomendada é usar diretamente o artefato apropriado do Lucene, por exemplo **org.apache.lucene:lucene-analyzers-common**.
- A API do controlador JMS foi alterada. A dependência de backend do JMS no Hibernate ORM foi removida para que pudesse ser usada em outros ambientes não-ORM. Uma consequência é que os implementadores de **org.hibernate.search.backend.impl.jms.AbstractJMSHibernateSearchController** deve ajustar-se à nova assinatura. Essa classe é uma classe interna e é recomendável usá-la como exemplo, em vez de estendê-la.
- o **org.hibernate.search.spi.ServiceProvider** O SPI foi refatorado. Se você estava integrando com o contrato de serviço antigo, consulte o [Hibernate Search 5.5 Javadoc](#) do **ServiceManager**, **Service**, **Startable** e **Stoppable** para detalhes sobre o novo contrato.
- Se você manteve os índices gerados pelo Lucene 3.xe não os reconstruiu com o Hibernate Search 5.0 ou posterior, você terá um **IndexFormatTooOldException**. Recomenda-se reconstruir os índices com o indexador de massa. Se você não for capaz de fazer isso, tente usar o Lucene **IndexUpgrader**. Você deve atualizar cuidadosamente os mapeamentos do Hibernate Search, caso o comportamento padrão tenha mudado. Para mais informações, consulte o [Guia de migração do Apache Lucene](#).
- O Apache Lucene foi atualizado de 3.6 para 5.3 no JBoss EAP 7. Se o seu código importar o código do Lucene diretamente, veja o [Guia de migração do Apache Lucene](#) para detalhes das alterações. Informações adicionais também podem ser encontradas no [Lucene Change Log](#).
- Ao usar **@Field(indexNullAs=)** Para codificar um valor de marcador nulo no índice, o tipo do marcador deve ser compatível com todos os outros valores indexados nesse mesmo campo. Por exemplo, anteriormente era possível codificar um valor nulo para campos numéricos usando uma string "null". Isso não é mais permitido. Em vez disso, você deve escolher um número para representar o **null** valor, como **-1**.

- Melhorias significativas foram feitas no motor de lapidação. A maioria das alterações não afeta a API. A única exceção notável é que você deve agora anotar todos os campos que você pretende usar para lapidação com o `@Facet` ou `@Facets` anotação.

Alterações no Hibernate Search [Esta é uma tradução automática]

A lista que segue é uma lista de classes de Hibernate Search que foram reempacotadas ou renomeadas.

Pacote e anterior e classe	Novo pacote e classe
org.hibernate.search.Environment	org.hibernate.search.cfg.Environment
org.hibernate.search.FullTextFilter	org.hibernate.search.filter.FullTextFilter
org.hibernate.search.ProjectionConstants	org.hibernate.search.engine.ProjectionConstants
org.hibernate.search.SearchException	org.hibernate.search.exception.SearchException
org.hibernate.search.Version	org.hibernate.search.engine.Version

Lucene - Classes renomeadas e reempacotadas

Os analisadores de consulta foram movidos para um novo módulo, resultando em uma alteração de `org.apache.lucene.queryParser.QueryParser` para `org.apache.lucene.queryparser.classic.QueryParser`.

Muitos dos analisadores Lucene foram refatorados, resultando em mudanças de embalagem. Veja o [Documentação do Apache Lucene](#) para encontrar os pacotes de substituição.

Algumas classes de utilitário do Apache Solr, por exemplo `TokenizerFactory` ou `TokenFilterFactory`, foram movidos para o Apache Lucene. Se o seu aplicativo usar esses utilitários ou analisadores personalizados, você deverá localizar o novo nome do pacote no Apache Lucene.

Veja o [Guia de migração do Apache Lucene](#) Para maiores informações.

Alterações no Hibernate Search [Esta é uma tradução automática]

Para obter a lista completa de interfaces, classes, enumerações, tipos de anotações, métodos, construtores e constantes enum do Hibernate Search, consulte [Hibernate Search Deprecated API](#) documento.

Alterações no Hibernate Search [Esta é uma tradução automática]

Interface	Descrição
org.hibernate.search.store.IndexShardingStrategy	Depreciado a partir do Hibernate Search 4.4. Pode ser removido na pesquisa 5. Use ShardIdentifierProvider em vez de.
org.hibernate.search.store.Workspace	Essa interface será movida e deve ser considerada uma API não pública. Para mais informações, veja HSEARCH-1915 .

Alterações no Hibernate Search [Esta é uma tradução automática]

Classe	Descrição
org.hibernate.search.filter.FilterKey	As chaves de filtragem personalizadas são preteridas e estão programadas para serem removidas em Hibernate Search 6. A partir de Hibernate Search 5.1, as chaves para caching os filtros Lucene são calculadas automaticamente baseando-se nos parâmetros de filtros fornecidos.
org.hibernate.search.filter.StandardFilterKey	As chaves de filtragem personalizadas são preteridas e estão programadas para serem removidas em Hibernate Search 6. A partir de Hibernate Search 5.1, as chaves para caching os filtros Lucene são calculadas automaticamente baseando-se nos parâmetros de filtros fornecidos.

Alterações no Hibernate Search [Esta é uma tradução automática]

Enum	Descrição
org.hibernate.search.annotations.FieldCacheType	Remova o CacheFromIndex anotação como é obsoleto. Veja Hibernate Search Annotations Deprecated .

Alterações no Hibernate Search [Esta é uma tradução automática]

Anotação	Descrição
org.hibernate.search.annotations.CacheFromIndex	Remover anotação. Não é necessário um substituto alternativo.
org.hibernate.search.annotations.Key	As chaves de cache de filtro personalizado são uma funcionalidade preterida e estão programadas para serem removidas em Hibernate Search 6. A partir de Hibernate Search 5.1, as chaves de cache de filtro são definidas automaticamente baseadas nos parâmetros de filtros, portanto não é mais necessário fornecer um objeto chave.

Alterações no Hibernate Search [Esta é uma tradução automática]

Método	Descrição
org.hibernate.search.FullTextSharedSessionBuilder.utoClose()	Sem substituição
org.hibernate.search.FullTextSharedSessionBuilder.utoClose(boolean)	Sem substituição

Método	Descrição
<code>org.hibernate.search.cfg.IndexedMapping.cacheFromIndex(FieldCacheType...)</code>	Isto será removido sem substituição.
<code>org.hibernate.search.cfg.EntityDescriptor.getCacheInMemory()</code>	Isto será removido sem substituição.
<code>org.hibernate.search.cfg.ContainedInMapping.numericField()</code>	Invocar <code>field().numericField()</code> em vez de.
<code>org.hibernate.search.cfg.EntityDescriptor.setCacheInMemory(Map<String, Object>)</code>	Isto será removido sem substituição.
<code>org.hibernate.search.MassIndexer.threadsForSubsequentFetching(int)</code>	Este método será removido.
<code>org.hibernate.search.query.dsl.FuzzyContext.withThreshold(float)</code>	Use <code>FuzzyContext.withEditDistanceUpTo(int)</code> .

Alterações no Hibernate Search [Esta é uma tradução automática]

Construtor	Descrição
<code>org.hibernate.search.cfg.NumericFieldMapping(PropertyDescriptor, EntityDescriptor, SearchMapping)</code>	Usar <code>NumericFieldMapping.NumericFieldMapping(String, PropertyDescriptor, EntityDescriptor, SearchMapping)</code> em vez de.

Alterações que impactam integradores avançados

Esta seção descreve alterações que não fazem parte da API pública. Isto não deve impactar o desenvolvedor comum pois estes artefatos devem ser acessados somente por integradores que estendem a framework Hibernate Search.

- o **`IndexWriterSetting.MAX_THREAD_STATES`** e **`IndexWriterSetting.TERM_INDEX_INTERVAL`** As constantes de enum são obsoletas. Eles afetam quais propriedades são lidas da configuração, portanto, o fato de estarem ausentes significa que as propriedades de configuração, como **`hibernate.search.Animals.2.indexwriter.term_index_interval = default`** agora são ignorados. O único efeito colateral é que a propriedade não é aplicada.
- o **`SearchFactoryIntegrator`** A interface agora está obsoleta. Você deve migrar imediatamente todo o código para usar **`SearchIntegrator`**.
- o **`SearchFactoryBuilder`** A classe agora está obsoleta. Usar **`SearchIntegrationBuilder`** em vez de.

- o `HSQuery.getExtendedSearchIntegrator()` o método foi descontinuado. Pode ser possível usar `SearchIntegrator`, mas é preferível removê-lo completamente.
- o `DocumentBuilderIndexedEntity.getFieldCacheOption()` o método foi descontinuado. Não há substituto.
- o `BuildContext.getIndexingStrategy()` método é obsoleto. Usar `BuildContext.getIndexingMode()` em vez de.
- o `DirectoryHelper.getVerifiedIndexDir(String, Properties, boolean)` método é obsoleto. Usar `DirectoryHelper.getVerifiedIndexPath(java.lang.String, java.util.Properties, boolean)` em vez de.
- A lista que segue é uma lista de classes de Hibernate Search que foram reempacotadas ou renomeadas.

Pacote e anterior e classe	Novo pacote e classe
org.hibernate.search.engine.impl.SearchMappingBuilder	org.hibernate.search.engine.spi.SearchMappingHelper
org.hibernate.search.indexes.impl.DirectoryBasedIndexManager	org.hibernate.search.indexes.spi.DirectoryBasedIndexManager
org.hibernate.search.spi.MassIndexerFactory	org.hibernate.search.batchindexing.spi.MassIndexerFactory
org.hibernate.search.spi.SearchFactoryBuilder	org.hibernate.search.spi.SearchIntegratorBuilder
org.hibernate.search.spi.SearchFactoryIntegrator	org.hibernate.search.spi.SearchIntegrator

5.9. MIGRAR DE BEANS DE ENTIDADE PARA JPA [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Suporte para beans de entidade EJB é facultativo em Java EE 7 e não são suportados em JBoss EAP 7. Isto significa que beans de entidade de persistência gerenciada por contêiner (CMP) e persistência gerenciada pelo bean (BMP) devem ser reescritos para usar entidades Java Persistence API (JPA) quando você migrar para JBoss EAP 7.

Nas versões anteriores do JBoss EAP, os beans de entidade eram criados no código-fonte do aplicativo, estendendo `javax.ejb.EntityBean` classe e implementar os métodos necessários. Eles foram então configurados no `ejb-jar.xml` Arquivo. Um bean de entidade CMP foi especificado usando um `<entity>` elemento que continha um `<persistence-type>` elemento filho com um valor de **Recipiente**. Um bean de entidade BMP foi especificado usando um `<entity>` elemento que continha um `<persistence-type>` elemento filho com um valor de **Feijão**.

No JBoss EAP 7, você deve substituir quaisquer beans de entidade CMP e BMP em seu código por entidades JPA (Java Persistence API). Entidades JPA são criadas usando o `javax.persistence.*` classes e são definidos no `persistence.xml` Arquivo.

O que segue é um exemplo de uma classe de entidade JPA.

–

```

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
// User is a keyword in some SQL dialects!
@Table(name = "MyUsers")
public class MyUser {
    @Id
    @GeneratedValue
    private Long id;

    @Column(unique = true)
    private String username;
    private String firstName;
    private String lastName;

    public Long getId() {
        return id;
    }
    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    public String getLastName() {
        return lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
}

```

O seguinte é um exemplo de um **persistence.xml** Arquivo.

```

<persistence version="2.1"
    xmlns="http://xmlns.jcp.org/xml/ns/persistence"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="
        http://xmlns.jcp.org/xml/ns/persistence
        http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
    <persistence-unit name="my-unique-persistence-unit-name">
        <properties>
            // properties...
        </properties>
    </persistence-unit>
</persistence>

```

Para exemplos de trabalho de entidades JPA, consulte o **bmt**, **cmt**, e **hibernate5** quickstarts que acompanham o JBoss EAP 7.

5.10. ALTERAÇÕES DE PROPRIEDADES DE PERSISTÊNCIA JPA [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Uma nova propriedade de persistência **jboss.as.jpa.deferdetach**, foi adicionado para fornecer compatibilidade com o comportamento de persistência em releases anteriores do JBoss EAP.

o **jboss.as.jpa.deferdetach** property controla se o contexto de persistência com escopo de transação usado em um encadeamento de transação não JTA desanexa as entidades carregadas após cada **EntityManager** invocação ou se aguarda até que o contexto de persistência seja fechado, por exemplo, quando a invocação do bean de sessão terminar. O valor da propriedade é padronizado para **false**, ou seja, as entidades são desanexadas ou limpas após cada **EntityManager** invocação. Esse é o comportamento padrão correto, conforme definido no [Especificação JPA](#). Se o valor da propriedade estiver definido como **true**, as entidades não são desanexadas até que o contexto de persistência seja fechado.

No JBoss EAP 5, a persistência se comportou como se **jboss.as.jpa.deferdetach** propriedade foi definida para **true**. Para obter esse mesmo comportamento ao migrar seu aplicativo do JBoss EAP 5 para o JBoss EAP 7, você deve definir **jboss.as.jpa.deferdetach** valor da propriedade para **true** na tua **persistence.xml** conforme mostrado no exemplo a seguir.

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence xmlns="http://java.sun.com/xml/ns/persistence" version="1.0">
  <persistence-unit name="EAP5_COMPAT_PU">
    <jta-data-source>java:jboss/datasources/ExampleDS</jta-data-source>
    <properties>
      <property name="jboss.as.jpa.deferdetach" value="true" />
    </properties>
  </persistence-unit>
</persistence>
```

No JBoss EAP 6, a persistência se comportou como se **jboss.as.jpa.deferdetach** propriedade foi definida para **false**. Este é o mesmo comportamento visto no JBoss EAP 7, portanto, nenhuma mudança é necessária quando você migra seu aplicativo.

5.10.1. Mudanças de propriedade de persistência do JPA no JBoss EAP 7.1 [Esta é uma tradução automática]

No JBoss EAP 7.0, a verificação de erro de contexto de persistência não sincronizada não era tão rigorosa quanto deveria nas áreas a seguir.

- Um contexto de persistência gerenciado por contêiner sincronizado tinha permissão para usar um contexto de persistência estendida não sincronizada que estava associado a uma transação JTA. Em vez disso, deveria ter lançado um **IllegalStateException** para impedir que o contexto de persistência não sincronizado seja usado.
- Um contexto de persistência não sincronizado especificado em um descritor de implantação foi tratado como sincronizado.

Além do que, além do mais, **PersistenceProperty** sugestões no **@PersistenceContext** foram erroneamente ignorados no JBoss EAP 7.0.

Esses problemas foram abordados e corrigidos no JBoss EAP 7.1. Como essas atualizações podem resultar em uma mudança indesejada no comportamento do aplicativo, duas novas propriedades de unidade de persistência foram introduzidas no JBoss EAP 7.1 para fornecer compatibilidade com versões anteriores e preservar o comportamento anterior.

Propriedade	Descrição
<code>wildfly.jpa.skipmixedsynccheckpointing</code>	Esta propriedade desativa a verificação de erros. Ele deve ser usado apenas como uma medida temporária para compatibilidade com versões anteriores em situações onde os aplicativos funcionavam no JBoss EAP 7.0 e falhavam no JBoss EAP 7.1. Como essa propriedade pode ser reprovada em uma versão futura, recomenda-se que você corrija o código do aplicativo assim que puder.
<code>wildfly.jpa.allowjoinedunsync</code>	Esta propriedade é uma alternativa para <code>wildfly.jpa.skipmixedsynccheckpointing</code> . Ele permite que o aplicativo trate contextos de persistência não sincronizados associados a uma transação JTA como se fossem contextos de persistência sincronizados.

5.11. MIGRAR O CÓDIGO DE CLIENTE EJB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

5.11.1. Mudanças no Cliente EJB no JBoss EAP 7 [Esta é uma tradução automática]

O conector e a porta remotos padrão foram alterados no JBoss EAP 7. Para obter detalhes sobre essa alteração, consulte [Atualizar o conector e porta de URL remoto](#).

Se você usou o **migrate** operação para migrar a configuração do servidor, as configurações antigas são preservadas e você não precisa fazer as alterações detalhadas abaixo. No entanto, se você executar com a nova configuração padrão do JBoss EAP 7, deverá fazer as seguintes alterações.

5.11.1.1. Atualize a porta de conexão remota padrão [Esta é uma tradução automática]

Altere o valor da porta de conexão remota de **4447** para **8080** no `jboss-ejb-client.properties` Arquivo.

Exemplo: `jboss-ejb-client.properties` Arquivo de propriedades [Esta é uma tradução automática]

Exemplo: JBoss EAP 6 `jboss-ejb-client.properties` Arquivo [Esta é uma tradução automática]

```
remote.connectionprovider.create.options.org.xnio.Options.SSL_ENABLED=false
remote.connections=default
remote.connection.default.host=localhost
```



```
remote.connection.default.port=4447
remote.connection.default.connect.options.org.xnio.Options.SASL_POLICY_NOA
NONYMOUS=false
```

Exemplo: JBoss EAP 7 `jboss-ejb-client.properties` Arquivo [Esta é uma tradução automática]

```
remote.connectionprovider.create.options.org.xnio.Options.SSL_ENABLED=false
remote.connections=default
remote.connection.default.host=localhost
remote.connection.default.port=8080
remote.connection.default.connect.options.org.xnio.Options.SASL_POLICY_NOA
NONYMOUS=false
```

5.11.1.2. Atualizar o conector padrão [Esta é uma tradução automática]

Se você estiver executando com a nova configuração do JBoss EAP 7, o conector padrão foi alterado de **remote** para **http-remoting**. Essa mudança impacta os clientes que usam bibliotecas de uma versão do JBoss EAP e se conectam ao servidor em uma versão diferente.

- Se um aplicativo cliente usa a biblioteca de cliente EJB do JBoss EAP 6 e deseja se conectar ao servidor do JBoss EAP 7, o servidor deve ser configurado para expor um **remote** conector em uma porta diferente **8080**. O cliente deve então se conectar usando esse conector recém-configurado.
- Um aplicativo cliente que usa a biblioteca de cliente EJB do JBoss EAP 7 e deseja se conectar ao servidor do JBoss EAP 6 deve estar ciente de que a instância do servidor não usa o **http-remoting** conector e, em vez disso, usa um **remote** conector. Isso é obtido definindo uma nova propriedade de conexão do lado do cliente.

Exemplo: `remote` Propriedade de Conexão [Esta é uma tradução automática]

```
remote.connection.default.protocol=remote
```

5.11.2. Migrar código de cliente de nomeação remota [Esta é uma tradução automática]

Se você está executando com a nova configuração padrão JBoss EAP 7, você deve modificar seu código cliente para usar a nova porta remota padrão e conector.

O exemplo seguinte é de como propriedades de nomeação remota foram especificadas no código cliente no JBoss EAP 6.

```
java.naming.factory.initial=org.jboss.naming.remote.client.InitialContextFactory
java.naming.provider.url=remote://localhost:4447
```

O exemplo seguinte é de como especificar as propriedades de nomeação remota no código cliente no JBoss EAP 7.

```
java.naming.factory.initial=org.wildfly.naming.client.WildFlyInitialContextFactory
java.naming.provider.url=http-remoting://localhost:8080
```

5.11.3. Alterações Adicionais do Cliente EJB Introduzidas no JBoss EAP 7.1 [Esta é uma tradução automática]

Enquanto o JBoss EAP 7.0 é fornecido com o JBoss EJB Client 2.1.4, o JBoss EAP 7.1 é fornecido com o JBoss EJB Client 4.0.x, que inclui várias mudanças na API.

- o `org.ejb.client.EJBClientInvocationContext` class adicionou os seguintes novos métodos.

Método	Tipo	Descrição
<code>isBlockingCaller()</code>	booleano	Determine se esta invocação está atualmente bloqueando o encadeamento de chamada.
<code>isClientAsync()</code>	booleano	Determine se o método está marcado como assíncrono do cliente, o que significa que a chamada deve ser assíncrona, independentemente de o método do lado do servidor ser assíncrono.
<code>isIdempotent()</code>	booleano	Determine se o método está marcado como idempotente, o que significa que o método pode ser chamado mais de uma vez sem efeito adicional.
<code>setBlockingCaller(boolean)</code>	vazio	Estabeleça se esta invocação está atualmente bloqueando o encadeamento de chamada.
<code>setLocator(EJBLocator<T>)</code>	<code><T> void</code>	Definir o localizador para o destino de invocação.

- o `org.ejb.client.EJBLocator` class adicionou os seguintes novos métodos.

Método	Tipo	Descrição
<code>asStateful()</code>	<code>StatefulEJBLocator<T></code>	Devolva este localizador como um localizador com estado, se for um.
<code>asStateless()</code>	<code>StatelessEJBLocator<T></code>	Devolva este localizador como um localizador sem estado, se for um.
<code>isEntity()</code>	booleano	Determine se este é um localizador de entidades.
<code>isHome()</code>	booleano	Determine se esse é um localizador de casas.

Método	Tipo	Descrição
isStateful()	booleano	Determine se este é um localizador com estado.
isStateless()	booleano	Determine se esse é um localizador sem estado.
withNewAffinity(Affinity)	abstract EJBLocator<T>	Crie uma cópia deste localizador, mas com a nova afinidade fornecida.

- Um novo **org.ejb.client.EJBClientPermission** classe, que é uma subclasse de **java.security.Permission**, foi introduzido para controlar o acesso a operações EJB privilegiadas.

- Ele fornece os seguintes construtores.

- **EJBClientPermission(String name)**
- **EJBClientPermission(String name, String actions)**

- Ele fornece os seguintes métodos.

Método	Tipo	Descrição
equals(EJBClientPermission obj)	booleano	Verifica dois EJBClientPermission objetos para igualdade.
equals(Object obj)	booleano	Verifica dois Permission objetos para igualdade.
equals(Permission obj)	booleano	Verifica dois Permission objetos para igualdade.
getActions()	String	Retorna as ações como uma string.
hashCode()	int	Retorna o valor do código hash para este Permission objeto.
implies(EJBClientPermission permission)	booleano	Verifica se as ações da permissão especificada são <i>implicado por</i> esta EJBClientPermission ações do objeto.
implies(Permission permission)	booleano	Verifica se as ações da permissão especificada são <i>implicado por</i> esta Permission ações do objeto.

- Um novo **org.ejb.client.EJBMethodLocator** Uma classe foi introduzida para localizar um método EJB específico.

- Ele fornece o construtor a seguir.

■ **EJBMethodLocator(String methodName, String... parameterTypeNames)**

- Ele fornece os seguintes métodos.

Método	Tipo	Descrição
equals(EJBMethodLocator other)	booleano	Determine se esse objeto é igual a outro.
equals(Object other)	booleano	Determine se esse objeto é igual a outro.
forMethod(Method method)	static EJBMethodLocator	Obtenha um localizador de método para o método de reflexão fornecido.
getMethodName()	String	Obtenha o nome do método.
getParameterCount()	int	Obtenha a contagem de parâmetros.
getParameterTypeName(int index)	String	Obtenha o nome do parâmetro no índice fornecido.
hashCode()	int	Obtenha o código hash.

- Um novo **org.jboss.ejb.client.EJBReceiverInvocationContext.ResultProducer.Failed** classe foi introduzida para casos de falha.

- Ele fornece o construtor a seguir.

■ **Failed(Exception cause)**

- Ele fornece os seguintes métodos.

Método	Tipo	Descrição
discardResult()	vazio	Descarte o resultado, indicando que ele não será usado.
getResult()	Object	Obter o resultado

- Um novo **org.jboss.ejb.client.EJBReceiverInvocationContext.ResultProducer.Immediate** classe foi introduzida para resultados imediatos.

- Ele fornece o construtor a seguir.

■ **Failed(Exception cause)**

- Ele fornece os seguintes métodos.

Método	Tipo	Descrição
discardResult()	vazio	Descarte o resultado, indicando que ele não será usado.
getResult()	Object	Obter o resultado

- Um novo **org.jboss.ejb.client.URIAffinity** classe, que é uma subclasse de **org.jboss.ejb.client.Affinity** foi introduzido para especificação de afinidade de URI.
 - É criado usando **Affinity.forUri(URI)**.
 - Ele fornece os seguintes métodos.

Método	Tipo	Descrição
equals(Affinity other)	booleano	Indica se outro objeto é igual a este.
equals(Object other)	booleano	Indica se outro objeto é igual a este.
equals(URIAffinity other)	booleano	Indica se outro objeto é igual a este.
getURI()	URI	Obtenha o URI associado.
hashCode()	int	Obtenha o código hash.
toString()	String	Retorna uma representação de string do objeto.

- o **org.jboss.ejb.client.EJBMetaDataImpl** classe depreciou os seguintes métodos.
 - **toAbstractEJBMetaData()**
 - **EJBMetaDataImpl(AbstractEJBMetaData<?, ?>)**

5.12. MIGRAR CLIENTES PARA USAR O ARQUIVO DE CONFIGURAÇÃO DO WILDFLY [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Antes do release 7.1, as bibliotecas do cliente do JBoss EAP, como EJB e nomeação, usavam diferentes estratégias de configuração. O JBoss EAP 7.1 introduz o **wildfly-config.xml** arquivo com o objetivo de unificar todas as configurações do cliente em um único arquivo de configuração, de maneira semelhante à maneira como a configuração do servidor é tratada.

Por exemplo, antes do JBoss EAP 7.1, você pode criar um novo **InitialContext** para um cliente Java EJB usando um **jboss-ejb-client.properties** arquivo ou definindo programaticamente as propriedades usando um **Properties** classe.

Exemplo: jboss-ejb-client.properties Arquivo de propriedades [Esta é uma tradução automática]

```
remote.connectionprovider.create.options.org.xnio.Options.SSL_ENABLED=false
remote.connections=one
remote.connection.one.port=8080
remote.connection.one.host=127.0.0.1
remote.connection.one.username=quickuser
remote.connection.one.password=quick-123
```

No JBoss EAP 7.1, você cria um **wildfly-config.xml** arquivo no **META-INF/** diretório do arquivo do cliente. Esta é a configuração equivalente usando um **wildfly-config.xml** Arquivo.

Exemplo: Configuração Equivalente Usando o wildfly-config.xml Arquivo [Esta é uma tradução automática]

```
<configuration>
  <authentication-client xmlns="urn:elytron:1.0.1">
    <authentication-rules>
      <rule use-configuration="ejb"/>
    </authentication-rules>
    <authentication-configurations>
      <configuration name="ejb">
        <set-user-name name="quickuser"/>
        <credentials>
          <clear-password password="quick-123"/>
        </credentials>
      </configuration>
    </authentication-configurations>
  </authentication-client>
  <jboss-ejb-client xmlns="urn:jboss:wildfly-client-ejb:3.0">
    <connections>
      <connection uri="remote+http://127.0.0.1:8080" />
    </connections>
  </jboss-ejb-client>
</configuration>
```

Para obter informações sobre como configurar a autenticação de cliente para o Elytron Client usando o **wildfly-config.xml** arquivo, consulte [Configurar autenticação de cliente com o cliente Elytron](#) dentro *Como configurar o gerenciamento de identidades* para o JBoss EAP.

Para obter mais informações sobre os tipos de configurações do cliente que podem ser feitas usando o **wildfly-config.xml** arquivo, consulte [Configuração do cliente usando o wildfly-config.xml Arquivo](#) no JBoss EAP *Guia de desenvolvimento*.

5.13. MIGRAR CONFIGURAÇÕES DE PLANOS DE IMPLEMENTAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

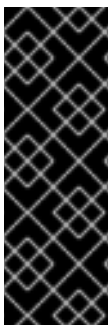
o [Especificação de implementação de aplicativo Java EE \(JSR-88\)](#) O objetivo era definir um contrato padrão para permitir que ferramentas de vários provedores configurassem e implantassem aplicativos em qualquer produto da plataforma Java EE. O contrato exigiu que os provedores de produtos Java EE implementassem o **DeploymentManager** e outro **javax.enterprise.deploy.spi** interfaces a serem acessadas pelos provedores de ferramentas. No caso do JBoss EAP 6, um plano de implementação é identificado por um descritor XML chamado **deployment-plan.xml** que é empacotado em um arquivo ZIP ou JAR.

Essa especificação teve pouca adoção porque a maioria dos produtos de servidor de aplicativos fornece suas próprias soluções de implantação mais "sofisticadas". Por esse motivo, o suporte a JSR-88 foi retirado do Java EE 7 e, por sua vez, do JBoss EAP 7.

Se você usou o JSR-88 para implantar seu aplicativo, deverá usar outro método para implantar o aplicativo. A CLI de gerenciamento do JBoss EAP **deploy** O comando fornece uma maneira padrão de implantar arquivos em servidores independentes ou em grupos de servidores em um domínio gerenciado. Para obter mais informações sobre a CLI de gerenciamento, consulte o [Guia do CLI de gerenciamento](#).

5.14. MIGRAR VÁLVULAS DE APLICATIVOS PERSONALIZADAS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Você deve migrar manualmente as válvulas personalizadas ou quaisquer válvulas definidas no **jboss-web.xml** Arquivo XML. Isto inclui válvulas criadas pela extensão do **org.apache.catalina.valves.ValveBase** classe e configurado no **<valve>** elemento do **jboss-web.xml** arquivo descritor.



IMPORTANTE

Válvulas e válvulas personalizadas definidas no **jboss-web.xml** arquivo deve ser reescrito ou substituído pelo manipulador interno Undertow correspondente. Para obter informações sobre como mapear válvulas para manipuladores Undertow, consulte [Migrar Válvulas Web JBoss](#).

Válvulas de autenticação devem ser substituídas manualmente usando mecanismos de autenticação internos Undertow.

Migrar configurações SSL [Esta é uma tradução automática]

No JBoss EAP 6, você pode definir válvulas personalizadas no nível do aplicativo, configurando-as no **jboss-web.xml** arquivo descritor de aplicativo da web. No JBoss EAP 7, é possível fazer isso com manipuladores Undertow também.

Segue-se um exemplo de uma válvula configurada no **jboss-web.xml** arquivo no JBoss EAP 6.

```
<jboss-web>
  <valve>
    <class-name>org.jboss.examples.MyValve</class-name>
    <param>
      <param-name>myParam</param-name>
      <param-value>foobar</param-value>
    </param>
  </valve>
</jboss-web>
```

Para obter mais informações sobre como criar e configurar manipuladores personalizados no JBoss EAP, consulte [Criando manipuladores personalizados](#) no JBoss EAP *Guia de desenvolvimento*.

Migrar válvulas do autenticador [Esta é uma tradução automática]

Para obter informações sobre como migrar as válvulas do autenticador, consulte [Migrar válvulas do autenticador](#) neste guia.

5.15. ALTERAÇÕES DE APLICATIVOS DE SEGURANÇA [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

A substituição de JBoss Web com Undertow requer alterações nas configurações de segurança no JBoss EAP 7.

5.15.1. Migrar válvulas do autenticador [Esta é uma tradução automática]

Se você criou uma válvula autenticadora personalizada que **AuthenticatorBase** no JBoss EAP 6.4, você deve substituí-lo manualmente por uma implementação de autenticação HTTP personalizada no JBoss EAP 7.1. O mecanismo de autenticação HTTP é criado no **elytron** subsistema e, em seguida, registrado com o **undertow** subsistema. Para obter informações sobre como implementar um mecanismo de autenticação HTTP personalizado, consulte [Implementando um Mecanismo HTTP Customizado](#) no *Guia de desenvolvimento* para o JBoss EAP.

5.15.2. Alterações de PicketLink [Esta é uma tradução automática]

Para obter informações sobre as alterações necessárias para o SSO com configuração do SAML v2, consulte [Mudanças das versões anteriores do JBoss EAP](#) dentro *Como configurar o SSO com o SAML v2* para o JBoss EAP.

5.15.3. Outras alterações de aplicativos de segurança [Esta é uma tradução automática]

Para obter informações sobre as diferenças na configuração de SSO com o Kerberos, consulte [Diferenças de Configurando Versões Anteriores do JBoss EAP](#) dentro *Como configurar o SSO com o Kerberos* para o JBoss EAP.

5.16. ALTERAÇÕES DE JBOSS LOGGING [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Se o seu aplicativo usa o JBoss Logging, esteja ciente de que as anotações no **org.jboss.logging** pacote agora estão obsoletos no JBoss EAP 7. Eles foram movidos para o **org.jboss.logging.annotations** pacote, portanto, você deve atualizar seu código-fonte para importar o novo pacote.

As anotações também foram movidas para um Maven separado **groupId:artifactId:version** (GAV) ID, então você precisa adicionar uma nova dependência de projeto para **org.jboss.logging:jboss-logging-annotations** no seu projecto **pom.xml** Arquivo.



NOTA

Apenas as anotações de registro foram movidas. o **org.jboss.logging.BasicLogger** e **org.jboss.logging.Logger** ainda existem no **org.jboss.logging** pacote.

A tabela seguinte lista as classes de anotações preteridas e substituições correspondentes.

Tabela 5.1. Substituições de anotações de registro em log preteridas [Esta é uma tradução automática]

Classes preteridas de RESTEasy [Esta é uma tradução automática]	Classes de substituição
org.jboss.logging.Cause	org.jboss.logging.annotations.Cause
org.jboss.logging.Field	org.jboss.logging.annotations.Field
org.jboss.logging.FormatWith	org.jboss.logging.annotations.FormatWith
org.jboss.logging.LoggingClass	org.jboss.logging.annotations.LoggingClass
org.jboss.logging.LogMessage	org.jboss.logging.annotations.LogMessage
org.jboss.logging.Message	org.jboss.logging.annotations.Message
org.jboss.logging.MessageBundle	org.jboss.logging.annotations.MessageBundle
org.jboss.logging.MessageLogger	org.jboss.logging.annotations.MessageLogger
org.jboss.logging.Param	org.jboss.logging.annotations.Param
org.jboss.logging.Property	org.jboss.logging.annotations.Property

5.17. ALTERAÇÕES AO CÓDIGO JAVASERVER FACES (JSF) [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Suporte para JSF 1.2 descontinuado

Exemplo: Bandeira de Anotação no `jboss-deployment-structure.xml` Arquivo [Esta é uma tradução automática]

JBoss EAP 7 inclui JSF 2.2 e não suporta mais a API JSF 1.2. Se seu aplicativo usa JSF 1.2, você deve regravá-lo para usar JSF 2.2.

Problema de compatibilidade entre JSF 2.1 e JSF 2.2

As APIs JSF 2.1 e JSF 2.2 não são totalmente compatíveis. o `FACELET_CONTEXT_KEY` valor constante alterado de `com.sun.faces.facelets.FACELET_CONTEXT` para `javax.faces.FACELET_CONTEXT` entre os lançamentos. Esse valor é embutido pelo compilador e o código compilado em um release não funcionará em relação ao outro.

Aplicativos que contêm código semelhante ao exemplo a seguir, e são compilados com a API do JSF 2.1, mas são executados no JBoss EAP 7, que usa a API JSF 2.2, resultando em um **NullPointerException**. Para corrigir o problema, você deve recompilar o aplicativo na API do JSF 2.2.

Exemplo: código Java que usa a API do JSF 2.1 [Esta é uma tradução automática]

■

```
Object obj =
FacesContext.getCurrentInstance().getAttributes().get(FaceletContext.FACELET_CONTEXT_KEY);
```

Veja [Impedir que a constante FaceletContext.FACELET_CONTEXT_KEY seja embutida pelo compilador](#) Para maiores informações.

5.18. ALTERAÇÕES AO CARREGAMENTO DE CLASSES DE MÓDULOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

No JBoss EAP 7, o comportamento do carregamento de classe foi alterado em casos onde múltiplos módulos contêm as mesmas classes ou pacotes.

Suponha que existem dois módulos, **MODULE_A** e **MODULE_B**, que dependem uns dos outros e contêm alguns dos mesmos pacotes. No JBoss EAP 6, as classes ou pacotes que foram carregados das dependências tiveram precedência sobre aqueles especificados no **resource-root** do **module.xml** Arquivo. Isso significava **MODULE_A** viu os pacotes para **MODULE_B** e **MODULE_B** viu os pacotes para **MODULE_A**. Esse comportamento foi confuso e poderia causar conflitos. Este comportamento mudou no JBoss EAP 7. Agora as classes ou pacotes especificados pelo **resource-root** no **module.xml** arquivo tem precedência sobre aqueles especificados pela dependência. Isso significa **MODULE_A** vê os pacotes para **MODULE_A** e **MODULE_B** vê os pacotes para **MODULE_B**. Isso evita conflitos e fornece um comportamento mais apropriado.

Se você tiver definido módulos personalizados que incluem **resource-root** bibliotecas ou pacotes que contêm classes duplicadas em suas dependências de módulo, você pode ver **ClassCastException**, **LinkageError**, erros de carregamento de classes ou outras mudanças de comportamento quando você migra para o JBoss EAP 7. Para resolver esses problemas, você deve **module.xml** arquivo para garantir que apenas uma versão de uma classe seja usada. Isso pode ser feito usando uma das seguintes abordagens.

- Você pode evitar especificar um **resource-root** que duplica as classes na dependência do módulo.
- Você pode usar o **include** e **exclude** sub-elementos do **imports** e **exports** elementos para controlar o carregamento da classe no **module.xml** Arquivo. A seguir, um elemento de exportação que exclui classes está no pacote especificado.

```
<exports>
  <exclude path="com/mycompany/duplicateclassspath"/>
</exports>
```

Se preferir preservar seu comportamento existente, você deve filtrar os pacotes de dependência do dependente **resource-root** no **module.xml** arquivo usando o **filter** elemento. Isso permite que você retenha o comportamento existente sem o loop impar que você veria no JBoss EAP 6. O seguinte é um exemplo de **root-resource** que filtra classes em um pacote especificado.

```
<resource-root path="mycompany.jar">
  <filter>
    <exclude path="com/mycompany/duplicateclassspath"/>
  </filter>
</resource-root>
```

Para obter mais informações sobre módulos e carregamento de classes, consulte [Carregamento de Classe e Módulos](#) no JBoss EAP *Guia de desenvolvimento*.

5.19. ALTERAÇÕES À CLUSTERIZAÇÃO DE APLICATIVOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

5.19.1. Visão geral de novas funcionalidades de clusterização [Esta é uma tradução automática]

A lista seguinte descreve algumas das novas funcionalidades de clusterização para considerar quando fizer a migração de seu aplicativo do JBoss EAP 6 para JBoss EAP 7.

- O JBoss EAP 7 introduz uma nova API pública para construir serviços singleton que simplifica significativamente o processo. Para obter informações sobre serviços singleton, consulte [Serviço HA Singleton](#) no JBoss EAP *Guia de desenvolvimento*
- Uma implementação de singleton pode ser configurada para implantar e iniciar somente um único nó no cluster de cada vez. Para mais informações, veja [Implantações de HA Singleton](#) no JBoss EAP *Guia de desenvolvimento*.
- Agora você pode definir MDBs singleton em cluster. Para mais informações, veja [MDBs de Singleton agrupados](#) dentro *Desenvolvendo Aplicativos EJB* para o JBoss EAP.
- O JBoss EAP 7 inclui a implementação Undertow mod_cluster. Isso oferece uma solução de balanceamento de carga Java pura que não requer um servidor da Web httpd. Para mais informações, veja [Configurando o JBoss EAP como um balanceador de carga front-end](#) no JBoss EAP *Guia de configuração*.

O restante desta seção descreve como as mudanças de cluster podem impactar a migração de seus aplicativos para o JBoss EAP 7.

5.19.2. Alterações nas Web Session Clustering [Esta é uma tradução automática]

O JBoss EAP 7 introduz uma nova implementação em web session clustering. Ela substitui a implementação anterior, que era acoplada rigidamente ao código-fonte do subsistema JBoss Web herdado.

A nova implementação de clustering de sessões da Web afeta o modo como o aplicativo é configurado no `jboss-web.xml` Arquivo de descritor XML do aplicativo da web proprietário do JBoss EAP. A seguir estão os únicos elementos de configuração de cluster que permanecem nesse arquivo.

```
<jboss-web>
...
<max-active-sessions>...</max-active-sessions>
...
<replication-config>
  <replication-granularity>...</replication-granularity>
  <cache-name>...</cache-name>
</replication-config>
...
</jboss-web>
```

A tabela a seguir descreve como obter um comportamento semelhante para elementos no `jboss-web.xml` arquivo que agora está obsoleto.

Mudanças de configuração de servidor [Esta é uma tradução automática]	Alterações de aplicativos de segurança [Esta é uma tradução automática]
<max-active-sessions/>	<p>Anteriormente, a criação da sessão falharia se fizesse com que o número de sessões ativas excedesse o valor especificado por <max-active-sessions/>.</p> <p>Na nova implementação, <max-active-sessions/> é usado para ativar a passivação de sessão. Se a criação da sessão fizer com que o número de sessões ativas exceda o <max-active-sessions/>, então a sessão mais antiga conhecida pelo gerente de sessão passará para dar lugar à nova sessão.</p>
<passivation-config/>	Este elemento de configuração e seus subelementos não são mais utilizados em JBoss EAP 7.
<use-session-passivation/>	<p>Anteriormente, a passivação era ativada utilizando este atributo.</p> <p>Na nova implementação, a passivação é ativada especificando um valor não negativo para <max-active-sessions/>.</p>
<passivation-min-idle-time/>	<p>Anteriormente, as sessões precisavam estar ativas por um período mínimo de tempo antes de tornarem-se candidatas à passivação. Isto poderia causar uma falha na criação de sessão, mesmo que a passivação estivesse habilitada.</p> <p>A nova implementação não suporta esta lógica e assim evita esta vulnerabilidade de recusa de serviço (Denial of Service - DoS) .</p>
<passivation-max-idle-time/>	<p>Anteriormente, uma sessão tornaria-se passiva após estar inativa por um período específico de tempo.</p> <p>A nova implementação só suporta passivação preguiçosa. Não suporta passivação ávida. As sessões só são passivadas quando necessário para cumprir <max-active-sessions/>.</p>
<replication-config/>	A nova implementação torna preterido um certo número de subelementos.
<replication-trigger/>	Anteriormente, esse elemento era usado para determinar quando a replicação da sessão era acionada. A nova implementação substitui essa opção de configuração por uma estratégia única e robusta. Para mais informações, veja Atributos de Sessão Imutáveis no JBoss EAP <i>Guia de desenvolvimento</i> .
<use-jk/>	<p>Anteriormente, o instance-id do nó que manipula uma determinada solicitação foi anexado ao jsessionId, para uso por balanceadores de carga como mod_jk, mod_proxy_balancer, mod_cluster, dependendo do valor especificado para <use-jk/>.</p> <p>Na nova implementação, o instance-id, se definido, é sempre anexado ao jsessionId.</p>

Mudanças de configuração de servidor [Esta é uma tradução automática]	Alterações de aplicativos de segurança [Esta é uma tradução automática]
<code><max-unreplicated-interval/></code>	<p>Anteriormente, esta opção de configuração era destinada a ser uma otimização para evitar a replicação de um carimbo de data/hora de sessão se não houvesse alteração de atributo de sessão. Isto parece bom, mas na prática não evita quaisquer RPCs, já que acesso à sessão necessita transação de cache RPCs independente de qualquer alteração de atributo de sessão.</p> <p>Na nova implementação, o carimbo de data/hora de uma sessão é replicado a cada solicitação. Isto previne metadados de sessões obsoletas após um failover.</p>
<code><snapshot-mode/></code>	Anteriormente, podia-se configurar <code><snapshot-mode/></code> Como INSTANT ou INTERVAL . A replicação assíncrona do Infinispan torna essa opção de configuração obsoleta.
<code><snapshot-interval/></code>	Isso só foi relevante para <code><snapshot-mode>INTERVAL</snapshot-mode></code> . Desde a <code><snapshot-mode/></code> é obsoleto, esta opção também está obsoleta.
<code><session-notification-policy/></code>	<p>Anteriormente, o valor especificado por este atributo definia uma política de acionamento de eventos de sessões.</p> <p>Na nova implementação este comportamento é acionado na especificação e não configurável.</p>

Esta nova implementação também suporta armazenamentos de cache de gravação, bem como armazenamentos de cache somente de passivação. Normalmente, armazenamentos de cache de gravação é usado em conjunto com um cache de invalidação. A implementação de cluster de sessões web no JBoss EAP 6 não funcionou corretamente quando usada com um cache de invalidação.

5.19.3. Alterações em Stateful Session EJB Clustering [Esta é uma tradução automática]

No JBoss EAP 6, era exigido que você habilitasse o comportamento do clustering para stateful session beans (SFSBs) em uma das seguintes maneiras.

- Você pode adicionar o `org.jboss.ejb3.annotation.Clustered` anotação no bean de sessão.

```
@Stateful
@Clustered
public class MyBean implements MySessionInt {

    public void myMethod() {
        //
    }
}
```

- Você pode adicionar o **<clustered>** elemento para o **jboss-ejb3.xml** Arquivo.

```
<c:clustering>
  <ejb-name>DDBasedClusteredSFSB</ejb-name>
  <c:clustered>true</c:clustered>
</c:clustering>
```

O JBoss EAP 7 não exige mais que você habilite o comportamento do clustering. Por padrão, se o servidor for iniciado utilizando um perfil HA, o estado de SFSBs será replicado automaticamente.

Você pode desabilitar este comportamento padrão com uma das seguintes maneiras.

- Você pode desabilitar o comportamento padrão de um único bean de sessão com **@Stateful(passivationCapable=false)**, que é novo na especificação EJB 3.2.
- Você pode desabilitar esse comportamento globalmente na configuração do **ejb3** subsistema na configuração do servidor.



NOTA

Se o **@Clustered** anotação não é removida do aplicativo, ela é simplesmente ignorada e não afeta a implementação do aplicativo.

5.19.4. Alterações nos serviços de clustering [Esta é uma tradução automática]

No JBoss EAP 6, as APIs para serviços de clustering estavam em módulos privados e não eram suportadas.

O JBoss EAP 7 introduz uma API de serviço de clustering público para ser utilizada pelos aplicativos. Os novos serviços são projetados para serem leves, facilmente injetáveis e não necessitar de dependências externas.

- O novo **org.wildfly.clustering.group.Group** A interface fornece acesso ao status atual do cluster e permite escutar alterações de associação de cluster.
- O novo **org.wildfly.clustering.dispatcher.CommandDispatcher** A interface permite executar código no cluster, em todos ou em um subconjunto selecionado de nós.

Esses serviços substituem APIs similares que estavam disponíveis em releases anteriores, **HAPartition** do JBoss EAP 5 e **GroupCommunicationService**, **GroupMembershipNotifier**, e **GroupRpcDispatcher** no JBoss EAP 6.

Para mais informações, veja [API pública para serviços de cluster](#) no JBoss EAP *Guia de desenvolvimento*.

5.19.5. Migrar Clustering HA Singleton [Esta é uma tradução automática]

No JBoss EAP 6, não havia nenhuma API pública disponível para o serviço de singleton de HA em todo o cluster. Se você usou o privado **org.jboss.as.clustering.singleton.*** classes, você deve alterar seu código para usar o novo **org.wildfly.clustering.singleton.*** pacotes quando você migra seu aplicativo para o JBoss EAP 7.

Para obter mais informações sobre serviços singleton de HA, consulte [Serviço HA Singleton](#) no *Guia de desenvolvimento* para o JBoss EAP. Para obter informações sobre implantações de singleton de alta disponibilidade, consulte [Implantações de HA Singleton](#) no *Guia de desenvolvimento* para o JBoss EAP.

CAPÍTULO 6. ALTERAÇÕES DIVERSAS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

6.1. ALTERAÇÕES NO MODO DE ENTREGA DO JBOSS EAP NATIVES E SERVIDOR HTTP APACHE [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

JBoss EAP 7 nativos são entregues de uma forma diferente neste lançamento. Agora alguns são enviados com o novo produto Red Hat JBoss Core Services, que é um grupo de softwares suplementares que é comum em muitos produtos middleware da Red Hat JBoss. O novo produto permite distribuição mais rápida de atualizações e uma experiência de atualização mais consistente. O produto JBoss Core Services está disponível para download em uma localidade diferente no Portal do Cliente Red Hat .

- A seguinte tabela lista as diferenças nos métodos de entrega entre os lançamentos.

Pacote	JBoss EAP 6	JBoss EAP 7
AIO Natives para mensagens	Entregue com o produto em um download "Native Utilities" separado	Incluído na distribuição de JBoss EAP. Não é necessário nenhum download adicional.
Servidor HTTP Apache	Entregue com o produto em um download "Servidor HTTP Apache" separado.	Entregue com o novo produto JBoss Core Services
mod_cluster, mod_jk, isapi, e conectores nsapi	Entregue com o produto em um download "Webserver Connector Natives" separado	Entregue com o novo produto JBoss Core Services
JSVC	Entregue com o produto em um download "Native Utilities" separado	Entregue com o novo produto JBoss Core Services
OpenSSL	Entregue com o produto em um download "Native Utilities" separado	Entregue com o novo produto JBoss Core Services
tcnatives	Entregue com o produto em um download "Native Components" separado	Revise O Que Há de Novo no JBoss EAP 7 [Esta é uma tradução automática]

- Você também deve estar ciente das seguintes alterações:
 - O suporte foi removido para os conectores mod_cluster e mod_jk usados com o Apache HTTP Server dos canais RPM do Red Hat Enterprise Linux. Se você executar o Apache HTTP Server a partir dos canais RPM do Red Hat Enterprise Linux e precisar configurar o balanceamento de carga para os servidores do JBoss EAP 7, você pode fazer o seguinte:
 - Use o servidor HTTP Apache fornecido pelo JBoss Core Services.
 - Você pode configurar o JBoss EAP 7 para atuar como um balanceador de carga front-end. Para mais informações, veja [Configurando o JBoss EAP como um balanceador de carga front-end](#) no JBoss EAP Guia de configuração.

- Você pode implantar o Apache HTTP Server em uma máquina que é suportada e certificada e, em seguida, executar o balanceador de carga nessa máquina. Para a lista de configurações suportadas, consulte [Visão Geral dos Conectores HTTP](#) no JBoss EAP 7 *Guia de configuração*.
- Foi eliminado suporte para conectores `mod_cluster` e `mod_jk` utilizados com o servidor HTTP Apache a partir do HP-UX Web Server Suites. Se você executar o servidor HTTP Apache a partir do HP-UX Web Server Suites e precisar configurar balanceamento de carga para servidores JBoss EAP 7, você pode fazer uma das seguintes opções:
 - Você pode configurar o JBoss EAP 7 para atuar como um balanceador de carga front-end. Para mais informações, veja [Configurando o JBoss EAP como um balanceador de carga front-end](#) no JBoss EAP *Guia de configuração*.
 - Você pode implantar o Apache HTTP Server em uma máquina que é suportada e certificada e, em seguida, executar o balanceador de carga nessa máquina. Para a lista de configurações suportadas, consulte [Visão Geral dos Conectores HTTP](#) no JBoss EAP *Guia de configuração*.
- Você pode encontrar mais informações sobre [JBoss Core Services](#) no *Guia de Instalação do Servidor HTTP Apache*.

6.2. ALTERAÇÕES ÀS IMPLEMENTAÇÕES NO AMAZON EC2 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Várias mudanças foram feitas na Amazon Machine Images (AMI) no JBoss EAP 7. Esta seção resume brevemente algumas dessas mudanças.

- O modo como você inicia instâncias JBoss EAP clusterizadas e não clusterizadas e domínios no Amazon EC2 foram alterados significativamente.
- O JBoss EAP 6 usou o **User Data**: campo para a configuração do JBoss EAP. Os scripts da AMI que analisaram a configuração no **User Data**: campo e iniciou os servidores automaticamente na inicialização da instância foram removidos do JBoss EAP 7.
- O agente Red Hat JBoss Operations Network era instalado em lançamentos prévios do JBoss EAP. No JBoss EAP 7, você deve instalá-lo separadamente.

Para detalhes sobre a implantação do JBoss EAP 7 no Amazon EC2, veja [Implantando Red Hat JBoss Enterprise Application Platform na Amazon EC2](#).

6.3. REMOÇÃO DE IMPLEMENTAÇÃO DE APLICATIVOS QUE INCLUEM MÓDULOS COMPARTILHADOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Mudanças no servidor do JBoss EAP 7.1 e no plug-in do Maven podem resultar na seguinte falha quando você tentar desimplantar seu aplicativo. Esse erro pode ocorrer se seu aplicativo contiver módulos que interajam ou dependam um do outro.

```
WFLYCTL0184:      New missing/unsatisfied dependencies
```

Por exemplo, suponha que você tenha um aplicativo que contenha dois módulos de projeto do Maven WAR, **application-A** e **application-B**, que compartilham dados gerenciados pelo **data-sharing** módulo.

Ao implantar esse aplicativo, você deve implantar o compartilhamento **data-sharing** primeiro módulo e, em seguida, implemente os módulos que dependem dele. A ordem de implantação é especificada no **<modules>** elemento do pai **pom.xml** Arquivo. Isso é verdade nas versões 6.4 a 7.1 do JBoss EAP.

Nas versões anteriores do JBoss EAP, você poderia desimplantar todos os arquivos para este aplicativo a partir da raiz do projeto pai usando o seguinte comando.

```
$ mvn wildfly:undeploy
```

No JBoss EAP 7.1, você deve primeiro desimplantar os arquivos que usam os módulos compartilhados e, em seguida, remover os módulos compartilhados. Como não há como especificar a ordem de desocupação no projeto **pom.xml** arquivo, você deve desimplantar os módulos manualmente. Você pode fazer isso executando os seguintes comandos a partir da raiz do diretório pai.

```
$ mvn wildfly:undeploy -pl application-A,application-B
$ mvn wildfly:undeploy -pl data-shared
```

Esse novo comportamento de remoção de implementação é mais correto e garante que você não acabe em um estado de implantação instável.

6.4. ALTERAÇÕES NOS SCRIPTS DO JBOSS EAP [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

o **add-user** O comportamento do script mudou no JBoss EAP 7 devido a uma mudança na política de senha. O JBoss EAP 6 tinha uma política de senha estrita. Como resultado, **add-user** O script rejeitou senhas fracas que não satisfaziam os requisitos mínimos. No JBoss EAP 7, senhas fracas são aceitas e um aviso é emitido. Para mais informações, veja [Configurando Restrições de Senha do Utilitário Add-User](#) no JBoss EAP *Guia de configuração*.

6.5. REMOÇÃO DE SUPORTE OSGI [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Quando o JBoss EAP 6.0 GA foi lançado pela primeira vez, o JBoss OSGi, uma implementação da especificação OSGi, foi incluído como um recurso de apresentação prévia de tecnologia. Com o lançamento do JBoss EAP 6.1.0, o JBoss OSGi foi rebaixado de apresentação técnica para não suportado.

No JBoss EAP 6.1.0, o **configadmin** e **osgi** módulos de extensão e configuração do subsistema para um servidor independente foram movidos para um **EAP_HOME/standalone/configuration/standalone-osgi.xml** arquivo de configuração. Como você não deve migrar esse arquivo de configuração não suportado, a remoção do suporte do JBoss OSGi não deve afetar a migração de uma configuração de servidor independente. Se você modificou qualquer um dos outros arquivos de configuração independentes para configurar **osgi** ou **configadmin**, essas configurações devem ser removidas.

Para um domínio gerenciado, o **osgi** extensão e configuração do subsistema foram removidos do **EAP_HOME/domain/configuration/domain.xml** arquivo na versão do JBoss EAP 6.1.0. No entanto, o **configadmin** a extensão do módulo e a configuração do subsistema permanecem no **EAP_HOME/domain/configuration/domain.xml** Arquivo. Esta configuração não é mais suportada no JBoss EAP 7 e deve ser removida.

CAPÍTULO 7. MIGRANDO PARA O ELYTRON NO JBOSS EAP

7.1 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

7.1. VISÃO GERAL DE ELYTRON [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

O JBoss EAP 7.1 apresenta o Elytron, que fornece uma única estrutura unificada que pode gerenciar e configurar o acesso para servidores independentes e domínios gerenciados. Também pode ser usado para configurar o acesso de segurança para aplicativos implantados nos servidores do JBoss EAP.

IMPORTANTE

As arquiteturas do Elytron e do subsistema de segurança legado que é baseado no PicketBox são muito diferentes. Com o Elytron, foi feita uma tentativa de criar uma solução que permite operar nos mesmos ambientes de segurança em que você opera atualmente; no entanto, isso faz *não* significa que cada opção de configuração do PicketBox tem uma opção de configuração equivalente no Elytron.

Se você não conseguir encontrar informações na documentação para ajudá-lo a obter uma funcionalidade semelhante usando o Elytron ao usar a implementação de segurança herdada, poderá encontrar ajuda de uma das maneiras a seguir.

- Se você tem um [Subscrição do Red Hat Development](#), você tem acesso a [Casos de suporte](#), [Soluções](#), e [Artigos de conhecimento](#) no Portal do Cliente Red Hat. Você também pode abrir um caso com [Suporte técnico](#) e [obter ajuda](#) da comunidade WildFly, conforme descrito abaixo.
- Se você não tem uma assinatura do Red Hat Development, você ainda pode acessar [Artigos de conhecimento](#) no Portal do Cliente Red Hat. Você também pode participar do [fóruns de usuários e chat ao vivo](#) para fazer perguntas da comunidade WildFly. As ofertas da comunidade WildFly são ativamente monitoradas pela equipe de engenharia da Elytron.

A configuração do seu servidor JBoss EAP 7.0 e as implementações que usam o legado **security** O subsistema, que é baseado no PicketBox, deve rodar sem alterações no JBoss EAP 7.1. O PicketBox continua a oferecer suporte a domínios de segurança, o que permite que os aplicativos continuem usando os módulos de login existentes. Os realms de segurança, que são usados pela camada de gerenciamento para segurança, também são transportados e emulados pelo Elytron. Isso permite que você defina a autenticação tanto no **elytron** e legado **security** subsistemas e usá-los em paralelo. Para obter mais informações sobre como configurar seu aplicativo para usar o Elytron e a segurança legada, consulte [Configurar aplicativos da Web para usar o Elytron ou a segurança legada para autenticação](#) dentro *Como configurar o gerenciamento de identidades* para o JBoss EAP.

Embora a autenticação do PicketBox continue sendo suportada, recomendamos que você alterne para o Elytron quando estiver pronto para migrar seus aplicativos. Uma das vantagens de usar a segurança Elytron é que ela fornece uma solução de segurança consistente em todo o servidor e seus aplicativos. Para obter informações sobre como migrar a autenticação e autorização do PicketBox para usar o Elytron, consulte [Migrar Configuração de Autenticação](#) neste guia.

Para obter uma visão geral dos novos recursos disponíveis no **elytron** subsistema, consulte [Recursos no subsistema Elytron](#) no JBoss EAP *Arquitetura de Segurança* guia.



IMPORTANTE

Esteja ciente de que, se você optar por usar tanto o legado **security** subsistema e Elytron em suas implementações, invocações entre implementações usando diferentes arquiteturas de segurança não são suportadas.

Para obter mais informações sobre o uso desses subsistemas em paralelo, consulte [Usando os subsistemas Elytron e Legacy Security em paralelo](#) dentro *Como configurar o gerenciamento de identidades* para o JBoss EAP.

7.2. MIGRAÇÃO DE COFRES E PROPRIEDADES SEGURAS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

7.2.1. Migrar Vaults para Proteger o Armazenamento de Credenciais [Esta é uma tradução automática]

O cofre que foi usado para armazenar criptografia de cadeia de texto simples no legado **security** O subsistema no JBoss EAP 7.0 não é compatível com o Elytron no JBoss EAP 7.1, que usa um novo armazenamento de credenciais para armazenar cadeias de caracteres. As lojas de credenciais criptografam com segurança credenciais em um arquivo de armazenamento fora dos arquivos de configuração do JBoss EAP. Você pode usar a implementação fornecida pelo Elytron ou pode personalizar a configuração usando as APIs e SPIs do repositório de credenciais. Cada servidor do JBoss EAP pode conter vários armazenamentos de credenciais.



NOTA

Se você usou anteriormente expressões de cofre para parametrizar dados não sensíveis, é recomendável substituir os dados por [Propriedades de segurança Elytron](#).

Se você continuar a usar o legado **security** subsistema, você não precisa modificar ou atualizar os dados do seu cofre. No entanto, se você planeja migrar seu aplicativo para usar o Elytron, é necessário converter seus cofres existentes em armazenamentos de credenciais para que possam ser processados pelo **elytron** subsistema. Para obter mais informações sobre armazenamentos de credenciais, consulte [Lojas de credenciais](#) dentro *Como configurar a segurança do servidor* para o JBoss EAP.

Migrar clientes para usar o arquivo de configuração do WildFly [Esta é uma tradução automática]

A ferramenta WildFly Elytron Tool, fornecida com o JBoss EAP, fornece **vault** comando para ajudá-lo a migrar o conteúdo do vault para armazenamentos de credenciais. Você executa a ferramenta executando o **elytron-tool** script, que está localizado no **EAP_HOME/bin** diretório.

```
$ EAP_HOME/bin/elytron-tool.sh vault VAULT_ARGUMENTS
```

Se preferir, você pode executar a ferramenta executando o **java -jar** comando.

```
$ java -jar EAP_HOME/bin/wildfly-elytron-tool.jar vault VAULT_ARGUMENTS
```

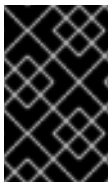
Você pode usar o seguinte comando para obter uma descrição de todos os argumentos disponíveis.

```
$ EAP_HOME/bin/elytron-tool.sh vault --help
```



NOTA

- A ferramenta WildFly Elytron não pode lidar com a primeira versão dos arquivos de dados do cofre de segurança.
- Você pode digitar o **--keystore-password** argumento em formato mascarado, como mostrado no exemplo abaixo para migrar um cofre único ou em texto não criptografado.
- o **--salt** e **--iteration** argumentos são fornecidos para fornecer informações para descriptografar a senha mascarada ou para gerar uma senha mascarada na saída. Se o **--salt** e **--iteration** argumentos são omitidos, valores padrão são usados.
- o **--summary** O argumento produz comandos CLI de gerenciamento formatados que podem ser usados para adicionar os armazenamentos de credenciais convertidos à configuração do JBoss EAP. Senhas de texto simples são mascaradas na saída de resumo.



IMPORTANTE

Esteja ciente de que os armazenamentos de credenciais só podem ser usados para proteger senhas. Eles não suportam o recurso de expressão do vault que pode ser usado em qualquer parte do modelo de gerenciamento.

Escolha uma das seguintes opções de migração:

- [Migrar um cofre de segurança único para um repositório de credenciais](#)
- [Migrar vários cofres de segurança para um repositório de credenciais em massa](#)

Migrar Vaults para Proteger o Armazenamento de Credenciais [Esta é uma tradução automática]

A seguir, um exemplo do comando usado para converter um único cofre de segurança em um armazenamento de credenciais.

```
$ EAP_HOME/bin/elytron-tool.sh vault --enc-dir vault_data/ --keystore
vault-jceks.keystore --keystore-password MASK-2hKo56F1a3jYGnJwhPmiF5 --
iteration 34 --salt 12345678 --alias test --location cs-v1.store --summary
```

Esse comando converte o cofre de segurança em um armazenamento de credenciais e imprime o resumo dos comandos da CLI de gerenciamento que foram usados para convertê-lo na saída.

```
Vault (enc-dir="vault_data/";keystore="vault-jceks.keystore") converted to
credential store "cs-v1.store"
Vault Conversion summary:
-----
Vault Conversion Successful
CLI command to add new credential store:
/subsystem=elytron/credential-store=test:add(relative-
to=jboss.server.data.dir,create=true,modifiable=true,location="cs-
v1.store",implementation-properties={"keyStoreType"=>"JCEKS"},credential-
reference={clear-text="MASK-2hKo56F1a3jYGnJwhPmiF5;12345678;34"})
```

Migrar Vaults para Proteger o Armazenamento de Credenciais [Esta é uma tradução automática]

Você pode converter vários cofres em um armazenamento de credenciais usando o **--bulk-convert** argumento e apontando para um arquivo descritor de conversão em massa.

Os exemplos nesta seção usam o seguinte arquivo descritor de conversão em massa.

Exemplo: bulk-vault-conversion-descriptor.txt Arquivo [Esta é uma tradução automática]

```
keystore:vault-v1/vault-jceks.keystore
keystore-password:MASK-2hKo56F1a3jYGnJwhPmiF5
enc-dir:vault-v1/vault_data/
salt:12345678
iteration:34
location:v1-cs-1.store
alias:test

keystore:vault-v1/vault-jceks.keystore
keystore-password:secretsecret
enc-dir:vault-v1/vault_data/
location:v1-cs-2.store
alias:test

# different vault vault-v1-more
keystore:vault-v1-more/vault-jceks.keystore
keystore-password:MASK-2hKo56F1a3jYGnJwhPmiF5
enc-dir:vault-v1-more/vault_data/
salt:12345678
iteration:34
location:v1-cs-more.store
alias:test
```

Uma nova conversão começa quando cada novo **keystore:** linha é encontrada. Todas as opções são obrigatórias, exceto para **salt**, **iteration**, e **properties**.

Para executar a conversão em massa e gerar saída que formata os comandos da CLI de gerenciamento, execute o seguinte comando.

```
$ EAP_HOME/bin/elytron-tool.sh vault --bulk-convert path/to/bulk-vault-
conversion-descriptor.txt --summary
```

Esse comando converte todas as áreas seguras especificadas no arquivo em um armazenamento de credenciais e imprime o resumo dos comandos da CLI de gerenciamento que foram usados para convertê-los na saída.

```
Vault (enc-dir="vault-v1/vault_data/";keystore="vault-v1/vault-
jceks.keystore") converted to credential store "v1-cs-1.store"
Vault Conversion summary:
-----
Vault Conversion Successful
CLI command to add new credential store:
/subsystem=elytron/credential-store=test:add(relative-
to=jboss.server.data.dir,create=true,modifiable=true,location="v1-cs-
1.store",implementation-properties={"keyStoreType"=>"JCEKS"},credential-
reference={clear-text="MASK-2hKo56F1a3jYGnJwhPmiF5;12345678;34"})
-----
```

```

Vault (enc-dir="vault-v1/vault_data/";keystore="vault-v1/vault-jceks.keystore") converted to credential store "v1-cs-2.store"
Vault Conversion summary:
-----
Vault Conversion Successful
CLI command to add new credential store:
/subsystem=elytron/credential-store=test:add(relative-to=jboss.server.data.dir,create=true,modifiable=true,location="v1-cs-2.store",implementation-properties={"keyStoreType"=>"JCEKS"},credential-reference={clear-text="secretsecret"})
-----

Vault (enc-dir="vault-v1-more/vault_data/";keystore="vault-v1-more/vault-jceks.keystore") converted to credential store "v1-cs-more.store"
Vault Conversion summary:
-----
Vault Conversion Successful
CLI command to add new credential store:
/subsystem=elytron/credential-store=test:add(relative-to=jboss.server.data.dir,create=true,modifiable=true,location="v1-cs-more.store",implementation-properties={"keyStoreType"=>"JCEKS"},credential-reference={clear-text="MASK-2hKo56F1a3jYGnJwhPmiF5;12345678;34"})
-----

```

7.2.2. Migrar propriedades de segurança para o Elytron [Esta é uma tradução automática]

Os exemplos nesta seção assumem que o `group.name` e `encoding.algorithm` propriedades de segurança são definidas como `security-properties` no legado `security` subsistema da seguinte forma.

Exemplo: propriedades de segurança definidas no security Subsistema [Esta é uma tradução automática]

```

<subsystem xmlns="urn:jboss:domain:security:2.0">
  ...
  <security-properties>
    <property name="group.name" value="engineering-group" />
    <property name="encoding.algorithm" value="BASE64" />
  </security-properties>
</subsystem>

```

Para definir as mesmas propriedades de segurança no **elytron** subsistema, defina o **security-properties** atributo do **elytron** subsistema usando o seguinte comando CLI de gerenciamento.

```

/subsystem=elytron:write-attribute(name=security-properties, value={
group.name = "engineering-group", encoding.algorithm = "BASE64" })

```

Isso configura o seguinte **security-properties** no **elytron** subsistema no arquivo de configuração do servidor.

```

<subsystem xmlns="urn:wildfly:elytron:1.2" final-providers="combined-

```



```
providers" disallowed-providers="OracleUcrypto">
  <security-properties>
    <security-property name="group.name" value="engineering-group"/>
    <security-property name="encoding.algorithm" value="BASE64"/>
  </security-properties>
  ...
</subsystem>
```

o **write-attribute** operação usada no comando anterior sobrescreve as propriedades existentes. Para adicionar ou alterar uma propriedade de segurança sem afetar outras propriedades de segurança, use o **map** operação no comando CLI de gerenciamento.

```
/subsystem=elytron:map-put(name=security-properties, key=group.name,
value=technical-support)
```

De maneira semelhante, você pode remover uma propriedade de segurança específica usando o **map-remove** Operação.

```
/subsystem=elytron:map-remove(name=security-properties, key=group.name)
```

7.3. MIGRAR CONFIGURAÇÃO DE AUTENTICAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

7.3.1. Migrar Autenticação e Autorização Baseadas em Propriedades para o Elytron [Esta é uma tradução automática]

7.3.1.1. Migrar a configuração baseada em propriedades do PicketBox para Elytron [Esta é uma tradução automática]

Esta seção descreve como migrar a autenticação baseada em propriedades do PicketBox para o Elytron. Você pode escolher [migrar parcialmente](#) autenticação baseada em propriedades expondo apenas o domínio de segurança PicketBox ao Elytron ou você pode [migrar totalmente](#) as configurações de autenticação baseadas em propriedades para usar o Elytron.

Os procedimentos a seguir presumem que o aplicativo da Web implementado que você planeja migrar está configurado para exigir autenticação baseada em formulário. O aplicativo está fazendo referência a um domínio de segurança PicketBox e está usando **UsersRolesLoginModule** para carregar informações do usuário do **example-users.properties** e **example-roles.properties** arquivos. Esses exemplos também assumem que o domínio de segurança é definido no legado **security** subsistema usando os seguintes comandos CLI de gerenciamento.

Exemplo: Comandos de Configuração Baseados em Propriedades do PicketBox [Esta é uma tradução automática]

```
/subsystem=security/security-domain=application-security:add
/subsystem=security/security-domain=application-
security/authentication=classic:add(login-modules=[{code=UsersRoles,
flag=Required, module-options=
{usersProperties=file://${jboss.server.config.dir}/example-
users.properties,
rolesProperties=file://${jboss.server.config.dir}/example-
roles.properties}}])
```

Isso resulta na seguinte configuração do servidor.

Exemplo: Configuração de Domínio de Segurança Baseada em Propriedades do PicketBox [Esta é uma tradução automática]

```
<security-domain name="application-security">
  <authentication>
    <login-module code="UsersRoles" flag="required">
      <module-option name="usersProperties"
value="file://${jboss.server.config.dir}/example-users.properties"/>
      <module-option name="rolesProperties"
value="file://${jboss.server.config.dir}/example-roles.properties"/>
    </login-module>
  </authentication>
</security-domain>
```

Escolha uma das seguintes opções de migração:

- [Migrar parcialmente expondo o domínio de segurança do PicketBox para Elytron.](#)
- [Migrar completamente a autenticação baseada em propriedades para o Elytron](#)

Migrar parcialmente expondo o domínio de segurança do PicketBox para Elytron

Você pode expor um domínio de segurança do PicketBox como um domínio de segurança Elytron para que ele possa ser conectado a uma configuração do Elytron; no entanto, isso cria uma dependência do legado **security** subsistema. Se você estiver migrando apenas a autenticação baseada em propriedades, é recomendável [migrar totalmente o aplicativo para Elytron](#) para evitar a dependência desnecessária do legado **security** subsistema. No entanto, uma migração parcial pode ser uma solução intermediária quando não for possível migrar completamente o aplicativo para usar o Elytron.

Siga este procedimento para adicionar uma configuração de região de segurança PicketBox existente como uma região de segurança Elytron.

1. Exemplo: propriedades de segurança definidas no **security** Subsistema [Esta é uma tradução automática]

```
/subsystem=security/elytron-realm=application-security:add(legacy-
jaas-config=application-security)
```

Isso configura o seguinte domínio de segurança Elytron no **security** subsistema do arquivo de configuração do servidor.

```
<subsystem xmlns="urn:jboss:domain:security:2.0">
  ...
  <elytron-integration>
    <security-realms>
      <elytron-realm name="application-security" legacy-jaas-
config="application-security"/>
    </security-realms>
  </elytron-integration>
  ...
</subsystem>
```


2. Definir um domínio de segurança no **elytron** subsistema que referencia a região de segurança exportada e uma fábrica de autenticação HTTP que suporta autenticação baseada em formulário.

```
/subsystem=elytron/security-domain=application-security:add(realms=[{realm=application-security}], default-realm=application-security, permission-mapper=default-permission-mapper)
/subsystem=elytron/http-authentication-factory=application-security-http:add(http-server-mechanism-factory=global, security-domain=application-security, mechanism-configurations=[{mechanism-name=DIGEST,mechanism-realm-configurations=[{realm-name=RealmName}]})]
```

Isso resulta nas seguintes **elytron** configuração do subsistema no arquivo de configuração do servidor.

```
<subsystem xmlns="urn:wildfly:elytron:1.2" final-
providers="combined-providers" disallowed-providers="OracleUcrypto">
  ...
  <security-domains>
    ...
    <security-domain name="application-security" default-
realm="application-security" permission-mapper="default-permission-
mapper">
      <realm name="application-security"/>
    </security-domain>
  </security-domains>
  ...
  <http>
    ...
    <http-authentication-factory name="application-security-http"
http-server-mechanism-factory="global" security-domain="application-
security">
      <mechanism-configuration>
        <mechanism mechanism-name="DIGEST">
          <mechanism-realm realm-name="RealmName"/>
        </mechanism>
      </mechanism-configuration>
    </http-authentication-factory>
    ...
  </http>
  ...
</subsystem>
```

3. Mapeie o domínio de segurança referenciado pela implementação para o factory de autenticação HTTP recém-definido no **undertow** subsistema.

```
/subsystem=undertow/application-security-domain=application-
security:add(http-authentication-factory=application-security-http)
```

Isso resulta na seguinte configuração no **undertow** subsistema do arquivo de configuração do servidor.

```
<subsystem xmlns="urn:jboss:domain:undertow:4.0">
  ...
```

```

<application-security-domains>
  <application-security-domain name="application-security" http-
authentication-factory="application-security-http"/>
</application-security-domains>
...
</subsystem>

```

4. Se o aplicativo já tiver sido implementado antes dessa configuração, você deverá recarregar o servidor ou reimplementar o aplicativo para que o mapeamento do novo domínio de segurança do aplicativo tenha efeito.
5. Verifique se o mapeamento foi aplicado à implantação usando o seguinte comando da CLI de gerenciamento. A implantação usada neste exemplo é **HelloWorld.war**. A saída do comando `this` mostra essa implementação referenciando o mapeamento Elytron.

```

/subsystem=undertow/application-security-domain=application-
security:read-resource(include-runtime=true)
{
  "outcome" => "success",
  "result" => {
    "enable-jacc" => false,
    "http-authentication-factory" => "application-security-
http",
    "override-deployment-config" => false,
    "referencing-deployments" => ["HelloWorld.war"],
    "setting" => undefined
  }
}

```

Nesta fase, o domínio de segurança previamente definido é usado para **LoginModule** configuração, mas é envolvido pelos componentes Elytron, que assumem a autenticação.

Migrar Autenticação e Autorização Baseadas em Propriedades para o Elytron [Esta é uma tradução automática]

Siga estas etapas para migrar completamente a autenticação baseada em propriedades do PicketBox para o Elytron. Este procedimento assume que você está começando com a configuração legada descrita na introdução desta seção e *não* migrou para o [parcialmente migrado](#) solução. Quando você concluir este processo, qualquer definição de domínio de segurança que exista no legado **security** subsistema permanece completamente independente da configuração do Elytron.

1. Defina um novo domínio de segurança no **elytron** subsistema que referencia os arquivos de propriedades do PicketBox.

```

/subsystem=elytron/properties-realm=application-
properties:add(users-properties={path=example-users.properties,
relative-to=jboss.server.config.dir, plain-text=true, digest-realm-
name="Application Security"}, groups-properties={path=example-
roles.properties, relative-to=jboss.server.config.dir}, groups-
attribute=Roles)

```

2. Definir um subsistema de domínio de segurança e um factory de autenticação HTTP no **elytron** subsistema.

```

/subsystem=elytron/security-domain=application-security:add(realms=
[{realm=application-properties}], default-realm=application-

```

```
properties, permission-mapper=default-permission-mapper)
/subsystem=elytron/http-authentication-factory=application-security-
http:add(http-server-mechanism-factory=global, security-
domain=application-security, mechanism-configurations=[{mechanism-
name=FORM}])
```

Isso resulta nas seguintes **elytron** configuração do subsistema no arquivo de configuração do servidor.

```
<subsystem xmlns="urn:wildfly:elytron:1.2" final-
providers="combined-providers" disallowed-providers="OracleUcrypto">
  ...
  <security-domains>
    ...
    <security-domain name="application-security" default-
realm="application-properties" permission-mapper="default-
permission-mapper">
      <realm name="application-properties"/>
    </security-domain>
  </security-domains>
  <security-realms>
    ...
    <properties-realm name="application-properties" groups-
attribute="Roles">
      <users-properties path="example-users.properties" relative-
to="jboss.server.config.dir" digest-realm-name="Application
Security" plain-text="true"/>
      <groups-properties path="example-roles.properties" relative-
to="jboss.server.config.dir"/>
    </properties-realm>
  </security-realms>
  ...
  <http>
    ...
    <http-authentication-factory name="application-security-http"
http-server-mechanism-factory="global" security-domain="application-
security">
      <mechanism-configuration>
        <mechanism mechanism-name="FORM"/>
      </mechanism-configuration>
    </http-authentication-factory>
    ...
  </http>
  ...
</subsystem>
```

3. Mapeie o domínio de segurança do aplicativo mencionado pela implementação para o factory de autenticação HTTP recém-definido no **undertow** subsistema.

```
/subsystem=undertow/application-security-domain=application-
security:add(http-authentication-factory=application-security-http)
```

Isso resulta nas seguintes **undertow** configuração do subsistema no arquivo de configuração do servidor.

```

<subsystem xmlns="urn:jboss:domain:undertow:4.0">
    ...
    <application-security-domains>
        <application-security-domain name="application-security" http-
authentication-factory="application-security-http"/>
    </application-security-domains>
    ...
</subsystem>

```

4. Você deve recarregar o servidor ou reimplementar o aplicativo para que o mapeamento do novo domínio de segurança do aplicativo tenha efeito.

A autenticação está agora configurada para ser equivalente à configuração do PicketBox; no entanto, os componentes Elytron são agora usados exclusivamente para autenticação.

7.3.1.2. Migrar a configuração baseada em propriedades herdadas para o Elytron [Esta é uma tradução automática]

Esta seção descreve como migrar uma região de segurança legada que carrega informações de usuário, senha e grupo dos arquivos de propriedades para o Elytron. Esse tipo de domínio de segurança herdado é normalmente usado para proteger as interfaces de gerenciamento ou os conectores remotos.

Estes exemplos assumem que o domínio de segurança legado é definido usando os seguintes comandos da CLI de gerenciamento.

Exemplo: Comandos do Reino de Segurança Legado [Esta é uma tradução automática]

```

/core-service=management/security-realm=ApplicationSecurity:add
/core-service=management/security-
realm=ApplicationSecurity/authentication=properties:add(relative-
to=jboss.server.config.dir, path=example-users.properties, plain-
text=true)
/core-service=management/security-
realm=ApplicationSecurity/authorization=properties:add(relative-
to=jboss.server.config.dir, path=example-roles.properties)

```

Isso resulta na seguinte configuração do servidor.

Exemplo: Comandos de Configuração do Reino de Segurança Legado [Esta é uma tradução automática]

```

<security-realm name="ApplicationSecurity">
    <authentication>
        <properties path="example-users.properties" relative-
to="jboss.server.config.dir" plain-text="true"/>
    </authentication>
    <authorization>
        <properties path="example-roles.properties" relative-
to="jboss.server.config.dir"/>
    </authorization>
</security-realm>

```

Uma das motivações para adicionar a segurança Elytron ao servidor de aplicativos é permitir que uma solução de segurança consistente seja usada em todo o servidor. As etapas iniciais para migrar um

domínio de segurança legado baseado em propriedades para o Elytron são semelhantes àsquelas usadas para migrar uma autenticação baseada em propriedades do PicketBox para o Elytron. Siga estas etapas para migrar um domínio de segurança legado baseado em propriedades para o Elytron.

1. Defina um novo domínio de segurança no **elytron** subsistema que referencia os arquivos de propriedades.

```
/subsystem=elytron/properties-realm=application-
properties:add(users-properties={path=example-users.properties,
relative-to=jboss.server.config.dir, plain-text=true, digest-realm-
name="Application Security"}, groups-properties={path=example-
roles.properties, relative-to=jboss.server.config.dir}, groups-
attribute=Roles)
```

2. Definir um subsistema de domínio de segurança e um factory de autenticação HTTP no **elytron** subsistema.

```
/subsystem=elytron/security-domain=application-security:add(realms=
[{{realm=application-properties}}, default-realm=application-
properties, permission-mapper=default-permission-mapper])
/subsystem=elytron/http-authentication-factory=application-security-
http:add(http-server-mechanism-factory=global, security-
domain=application-security, mechanism-configurations=[{{mechanism-
name=FORM}}])
```

Isso resulta na seguinte configuração Elytron.

```
<subsystem xmlns="urn:wildfly:elytron:1.2" final-
providers="combined-providers" disallowed-providers="OracleUcrypto">
...
  <security-domains>
    ...
    <security-domain name="application-security" default-
realm="application-properties" permission-mapper="default-
permission-mapper">
      <realm name="application-properties"/>
    </security-domain>
  </security-domains>
  <security-realms>
    ...
    <properties-realm name="application-properties" groups-
attribute="Roles">
      <users-properties path="example-users.properties" relative-
to="jboss.server.config.dir" digest-realm-name="Application
Security" plain-text="true"/>
      <groups-properties path="example-roles.properties" relative-
to="jboss.server.config.dir"/>
    </properties-realm>
  </security-realms>
  ...
  <http>
    ...
    <http-authentication-factory name="application-security-http"
http-server-mechanism-factory="global" security-domain="application-
security">
```

```

        <mechanism-configuration>
            <mechanism mechanism-name="FORM"/>
        </mechanism-configuration>
    </http-authentication-factory>
    ...
</http>
...
</subsystem>

```

3. Definir um **sasl-authentication-factory** para que o domínio de segurança legado também possa ser usado para autenticação SASL (Simple Authentication Security Layer).

```

/subsystem=elytron/sasl-authentication-factory=application-security-
sasl:add(sasl-server-factory=elytron, security-domain=application-
security, mechanism-configurations=[{mechanism-name=PLAIN}])

```

Isso resulta na seguinte configuração Elytron.

```

<subsystem xmlns="urn:wildfly:elytron:1.2" final-
providers="combined-providers" disallowed-providers="OracleUcrypto">
    ...
    <sasl>
        ...
        <sasl-authentication-factory name="application-security-sasl"
sasl-server-factory="elytron" security-domain="application-
security">
            <mechanism-configuration>
                <mechanism mechanism-name="PLAIN"/>
            </mechanism-configuration>
        </sasl-authentication-factory>
        ...
    </sasl>
</subsystem>

```

4. Configure um conector remoto para a autenticação SASL e remova a associação com a região de segurança legada.

```

/subsystem=remoting/http-connector=http-remoting-connector:write-
attribute(name=sasl-authentication-factory, value=application-
security-sasl)
/subsystem=remoting/http-connector=http-remoting-connector:undefine-
attribute(name=security-realm)

```

Isso resulta na seguinte configuração no **remoting** subsistema do arquivo de configuração do servidor.

```

<subsystem xmlns="urn:jboss:domain:remoting:4.0">
    ...
    <http-connector name="http-remoting-connector" connector-
ref="default" sasl-authentication-factory="application-security-
sasl"/>
</subsystem>

```

- Adicione as duas fábricas de autenticação e remova as referências de domínio de segurança legadas para proteger **http-interface** com Elytron.

```
/core-service=management/management-interface=http-interface:write-
attribute(name=http-authentication-factory, value=application-
security-http)
/core-service=management/management-interface=http-interface:write-
attribute(name=http-upgrade.sasl-authentication-factory,
value=application-security-sasl)
/core-service=management/management-interface=http-
interface:undefine-attribute(name=security-realm)
```

Isso resulta na seguinte configuração.

```
<management-interfaces>
  <http-interface http-authentication-factory="application-security-
http">
    <http-upgrade enabled="true" sasl-authentication-
factory="application-security-sasl"/>
    <socket-binding http="management-http"/>
  </http-interface>
</management-interfaces>
```



NOTA

Você deve escolher nomes mais adequados do que aqueles usados nesses exemplos ao proteger as interfaces de gerenciamento.

Migrar a configuração baseada em propriedades herdadas para o Elytron [Esta é uma tradução automática]

7.3.2. Migrar a configuração de autenticação LDAP para o Elytron [Esta é uma tradução automática]

Esta seção descreve como migrar a autenticação LDAP herdada para o Elytron para que ela possa gerenciar as informações como atributos de identidade. Muitas das informações fornecidas na seção intitulada [Migrar Autenticação e Autorização Baseadas em Propriedades para o Elytron](#) aplica-se aqui, particularmente sobre como definir domínios de segurança e fábricas de autenticação, e como mapeá-los para serem usados para autenticação. Esta seção não repete essas instruções, portanto, leia essa seção antes de continuar.

Os exemplos a seguir supõem que as informações de grupo ou função são carregadas diretamente do LDAP e que a autenticação LDAP herdada é configurada da seguinte maneira.

- O servidor LDAP contém as seguintes entradas de usuário e grupo.

Exemplo: Entradas do Usuário do Servidor LDAP [Esta é uma tradução automática]

```
dn: uid=TestUserOne,ou=users,dc=group-to-principal,dc=wildfly,dc=org
objectClass: top
objectClass: inetOrgPerson
objectClass: uidObject
objectClass: person
```

```
objectClass: organizationalPerson
cn: Test User One
sn: Test User One
uid: TestUserOne
userPassword: {SSHA}UG8ov2rnrnBKakcARVvraZHqTa7mFWJZlWt2HA==
```

Exemplo: Entradas do Grupo de Servidores LDAP [Esta é uma tradução automática]

```
dn: uid=GroupOne,ou=groups,dc=group-to-principal,dc=wildfly,dc=org
objectClass: top
objectClass: groupOfUniqueNames
objectClass: uidObject
cn: Group One
uid: GroupOne
uniqueMember: uid=TestUserOne,ou=users,dc=group-to-
principal,dc=wildfly,dc=org
```

Para fins de autenticação, o nome do usuário é comparado com o **uid** atributo e o nome do grupo resultante é retirado do **uid** atributo da entrada do grupo.

- A conexão com o servidor LDAP e a região de segurança relacionada é definida usando os seguintes comandos da CLI de gerenciamento.

Exemplo: Comandos de configuração do domínio de segurança do LDAP [Esta é uma tradução automática]

```
batch
/core-service=management/ldap-
connection=MyLdapConnection:add(url="ldap://localhost:10389",
search-dn="uid=admin,ou=system", search-credential="secret")

/core-service=management/security-realm=LDAPRealm:add
/core-service=management/security-
realm=LDAPRealm/authentication=ldap:add(connection="MyLdapConnection
", username-attribute=uid, base-dn="ou=users,dc=group-to-
principal,dc=wildfly,dc=org")

/core-service=management/security-
realm=LDAPRealm/authorization=ldap:add(connection=MyLdapConnection)
/core-service=management/security-
realm=LDAPRealm/authorization=ldap/username-to-dn=username-
filter:add(attribute=uid, base-dn="ou=users,dc=group-to-
principal,dc=wildfly,dc=org")
/core-service=management/security-
realm=LDAPRealm/authorization=ldap/group-search=group-to-
principal:add(base-dn="ou=groups,dc=group-to-
principal,dc=wildfly,dc=org", iterative=true, prefer-original-
connection=true, principal-attribute=uniqueMember, search-
by=DISTINGUISHED_NAME, group-name=SIMPLE, group-name-attribute=uid)
run-batch
```

Isso resulta na seguinte configuração do servidor.

Exemplo: Configuração do Realm de Segurança LDAP [Esta é uma tradução automática]

```

<management>
  <security-realms>
    ...
    <security-realm name="LDAPRealm">
      <authentication>
        <ldap connection="MyLdapConnection" base-
dn="ou=users,dc=group-to-principal,dc=wildfly,dc=org">
          <username-filter attribute="uid"/>
        </ldap>
      </authentication>
      <authorization>
        <ldap connection="MyLdapConnection">
          <username-to-dn>
            <username-filter base-dn="ou=users,dc=group-to-
principal,dc=wildfly,dc=org" attribute="uid"/>
          </username-to-dn>
          <group-search group-name="SIMPLE" iterative="true" group-
name-attribute="uid">
            <group-to-principal search-by="DISTINGUISHED_NAME" base-
dn="ou=groups,dc=group-to-principal,dc=wildfly,dc=org" prefer-
original-connection="true">
              <membership-filter principal-
attribute="uniqueMember"/>
            </group-to-principal>
          </group-search>
        </ldap>
      </authorization>
    </security-realm>
  </security-realms>
  <outbound-connections>
    <ldap name="MyLdapConnection" url="ldap://localhost:10389"
search-dn="uid=admin,ou=system" search-credential="secret"/>
  </outbound-connections>
  ...
</management>

```

- Os seguintes comandos CLI de gerenciamento são usados para configurar um domínio de segurança PicketBox, que usa o **LdapExtLoginModule** para verificar um nome de usuário e senha.

Exemplo: Comandos de Configuração do Domínio de Segurança [Esta é uma tradução automática]

```

/subsystem=security/security-domain=application-security:add
/subsystem=security/security-domain=application-
security/authentication=classic:add(login-modules=
[{code=LdapExtended, flag=Required, module-options={
java.naming.factory.initial=com.sun.jndi.ldap.LdapCtxFactory,
java.naming.provider.url=ldap://localhost:10389,
java.naming.security.authentication=simple,
bindDN="uid=admin,ou=system", bindCredential=secret,
baseCtxDN="ou=users,dc=group-to-principal,dc=wildfly,dc=org",

```

```
baseFilter="(uid={0})", rolesCtxDN="ou=groups,dc=group-to-
principal,dc=wildfly,dc=org", roleFilter="(uniqueMember={1})",
roleAttributeID="uid" }]])
```

Isso resulta na seguinte configuração do servidor.

Exemplo: Configuração do Domínio de Segurança [Esta é uma tradução automática]

```
<subsystem xmlns="urn:jboss:domain:security:2.0">
  ...
  <security-domains>
    ...
    <security-domain name="application-security">
      <authentication>
        <login-module code="LdapExtended" flag="required">
          <module-option name="java.naming.factory.initial"
value="com.sun.jndi.ldap.LdapCtxFactory"/>
          <module-option name="java.naming.provider.url"
value="ldap://localhost:10389"/>
          <module-option name="java.naming.security.authentication"
value="simple"/>
          <module-option name="bindDN" value="uid=admin,ou=system"/>
          <module-option name="bindCredential" value="secret"/>
          <module-option name="baseCtxDN" value="ou=users,dc=group-
to-principal,dc=wildfly,dc=org"/>
          <module-option name="baseFilter" value="(uid={0})"/>
          <module-option name="rolesCtxDN"
value="ou=groups,dc=group-to-principal,dc=wildfly,dc=org"/>
          <module-option name="roleFilter" value="(uniqueMember=
{1})"/>
          <module-option name="roleAttributeID" value="uid"/>
        </login-module>
      </authentication>
    </security-domain>
  </security-domains>
</subsystem>
```

7.3.2.1. Migre a Autenticação LDAP Legada para Elytron [Esta é uma tradução automática]

Siga estas etapas para migrar a configuração anterior do exemplo de autenticação LDAP para o Elytron. Esta seção aplica-se à migração de um [domínio LDAP de segurança legada](#) bem como um [Domínio de segurança do PicketBox LDAP](#).

1. Exemplo: propriedades de segurança definidas no **security** Subsistema [Esta é uma tradução automática]

```
/subsystem=elytron/dir-context=ldap-
connection:add(url=ldap://localhost:10389, principal="uid=admin,
ou=system", credential-reference={clear-text=secret})
```

2. Crie um domínio de segurança para pesquisar o LDAP e verifique a senha fornecida.

```
/subsystem=elytron/ldap-realm=ldap-realm:add(dir-context=ldap-connection, direct-verification=true, identity-mapping={search-base-dn="ou=users, dc=group-to-principal, dc=wildfly, dc=org", rdn-identifier="uid", attribute-mapping=[{filter-base-dn="ou=groups, dc=group-to-principal, dc=wildfly, dc=org", filter="(uniqueMember={1})", from="uid", to="Roles"}]})
```

Essas etapas resultam nas seguintes **elytron** configuração do subsistema no arquivo de configuração do servidor.

```
<subsystem xmlns="urn:wildfly:elytron:1.2" final-providers="combined-providers" disallowed-providers="OracleUcrypto">
  ...
  <security-realms>
    ...
    <ldap-realm name="ldap-realm" dir-context="ldap-connection" direct-verification="true">
      <identity-mapping rdn-identifier="uid" search-base-dn="ou=users,dc=group-to-principal,dc=wildfly,dc=org">
        <attribute-mapping>
          <attribute from="uid" to="Roles" filter="(uniqueMember={1})" filter-base-dn="ou=groups,dc=group-to-principal,dc=wildfly,dc=org"/>
        </attribute-mapping>
      </identity-mapping>
    </ldap-realm>
  </security-realms>
  ...
  <dir-contexts>
    <dir-context name="ldap-connection" url="ldap://localhost:10389" principal="uid=admin,ou=system">
      <credential-reference clear-text="secret"/>
    </dir-context>
  </dir-contexts>
</subsystem>
```



NOTA

Por padrão, se não **role-decoder** é definido para um dado **security-domain**, o atributo de identidade "Funções" é mapeado para as funções de identidade.

Informações carregadas do LDAP agora podem ser associadas a identidades como atributos. Esses atributos podem ser mapeados para funções, mas também podem ser carregados e usados para outras finalidades. O domínio de segurança recém-criado pode ser usado em um domínio de segurança da mesma forma que é descrito no [Migrar Autenticação e Autorização Baseadas em Propriedades para o Elytron](#) seção deste guia.

7.3.3. Migrar a configuração de autenticação de banco de dados para o Elytron [Esta é uma tradução automática]

Esta seção descreve como migrar a autenticação PicketBox baseada em origem de dados JDBC para o Elytron. Muitas das informações fornecidas na seção intitulada [Migrar Autenticação e Autorização Baseadas em Propriedades para o Elytron](#) aplica-se aqui, particularmente sobre como definir domínios

de segurança e fábricas de autenticação, e como mapeá-los para serem usados para autenticação. Esta seção não repete essas instruções, portanto, leia essa seção antes de continuar.

Os exemplos a seguir presumem que os dados de autenticação do usuário estão armazenados em uma tabela de banco de dados criada usando uma sintaxe semelhante ao exemplo a seguir.

Exemplo: sintaxe para criar a tabela de usuário do banco de dados [Esta é uma tradução automática]

```
CREATE TABLE User (
    id BIGINT NOT NULL,
    username VARCHAR(255),
    password VARCHAR(255),
    role ENUM('admin', 'manager', 'user'),
    PRIMARY KEY (id),
    UNIQUE (username)
)
```

Para fins de autenticação, o nome de usuário é comparado com os dados armazenados no **username** coluna, espera-se que a senha seja armazenada como um hash MD5 codificado hexadecimal no **password** coluna, ea função de usuário para fins de autorização é armazenada no **role** coluna.

O domínio de segurança PicketBox é configurado para usar uma origem de dados JDBC para recuperar dados da tabela de banco de dados e, em seguida, usá-lo para verificar o nome de usuário e a senha e para atribuir funções. Suponha que o domínio de segurança PicketBox esteja configurado usando os seguintes comandos da CLI de gerenciamento.

Exemplo: Comandos de Configuração do LoginModule do Banco de Dados do PicketBox [Esta é uma tradução automática]

```
/subsystem=security/security-domain=application-security:add
/subsystem=security/security-domain=application-
security/authentication=classic:add( login-modules=[ { code=Database,
flag=Required, module-options={
dsJndiName="java:jboss/datasources/ExampleDS", principalsQuery="SELECT
password FROM User WHERE username = ?", rolesQuery="SELECT role, 'Roles'
FROM User WHERE username = ?", hashAlgorithm=MD5, hashEncoding=base64 } } ]
)
```

Isso resulta nas seguintes **login-module** configuração no legado **security** subsistema.

Exemplo: Configuração do PicketBox LoginModule [Esta é uma tradução automática]

```
<subsystem xmlns="urn:jboss:domain:security:2.0">
  <security-domains>
    ...
    <security-domain name="application-security">
      <authentication>
        <login-module code="Database" flag="required">
          <module-option name="dsJndiName"
value="java:jboss/datasources/ExampleDS"/>
          <module-option name="principalsQuery" value="SELECT password
FROM User WHERE username = ?"/>
          <module-option name="rolesQuery" value="SELECT role, 'Roles'
FROM User WHERE username = ?"/>
        
```

```

        <module-option name="hashAlgorithm" value="MD5"/>
        <module-option name="hashEncoding" value="base64"/>
    </login-module>
</authentication>
</security-domain>
</security-domains>
</subsystem>

```

7.3.3.1. Migre a Autenticação de Banco de Dados Legada para Elytron [Esta é uma tradução automática]

Para migrar a configuração anterior do exemplo de autenticação do banco de dados para o Elytron, você deve definir uma região JDBC para ativar o acesso à origem de dados JDBC pelo Elytron.

Utilize o seguinte comando de gerenciamento para definir **jdbc-realm**.

```

/subsystem=elytron/jdbc-realm=jdbc-realm:add(principal-query=[ { data-
source=ExampleDS, sql="SELECT role, password FROM User WHERE username =
?", attribute-mapping=[{index=1, to=Roles } ] simple-digest-mapper=
{algorithm=simple-digest-md5, password-index=2} } ] )

```

Isso resulta nas seguintes **jdbc-realm** configuração no **elytron** subsistema do arquivo de configuração do servidor.

```

<subsystem xmlns="urn:wildfly:elytron:1.2" final-providers="combined-
providers" disallowed-providers="OracleUcrypto">
    ...
    <security-realms>
        ...
        <jdbc-realm name="jdbc-realm">
            <principal-query sql="SELECT role, password FROM User WHERE username
= ?" data-source="ExampleDS">
                <attribute-mapping>
                    <attribute to="Roles" index="1"/>
                </attribute-mapping>
                <simple-digest-mapper password-index="2"/>
            </principal-query>
        </jdbc-realm>
        ...
    </security-realms>
    ...
</subsystem>

```

O Elytron agora gerencia a autenticação do banco de dados usando a configuração do território JDBC. O Elytron é mais eficiente que o PicketBox porque usa uma consulta SQL para obter todos os atributos e credenciais do usuário e, em seguida, extrai dados dos resultados do SQL e cria um mapeamento dos atributos a serem usados para autenticação.

7.3.4. Migração da autenticação Kerberos para o Elytron [Esta é uma tradução automática]

Ao trabalhar com uma configuração do Kerberos, o servidor do JBoss EAP pode confiar nas informações de configuração do ambiente, ou a configuração da chave pode ser especificada usando as propriedades do sistema. Esta seção discute como migrar [HTTP Kerberos](#) e [Kerberos SASL](#).

autenticação.

Os exemplos a seguir pressupõem que o Kerberos esteja configurado usando as seguintes propriedades do sistema. Essas propriedades do sistema são aplicáveis à configuração herdada e à configuração migrada do Elytron.

Exemplo: Comandos CLI de gerenciamento de propriedades do sistema Kerberos [Esta é uma tradução automática]

```
# Enable debugging
/system-property=sun.security.krb5.debug:add(value=true)
# Identify the Kerberos realm to use
/system-property=java.security.krb5.realm:add(value=ELYTRON.ORG)
# Identify the address of the KDC
/system-property=java.security.krb5.kdc:add(value=kdc.elytron.org)
```

Exemplo: configuração do servidor de propriedades do sistema Kerberos [Esta é uma tradução automática]

```
<system-properties>
  <property name="sun.security.krb5.debug" value="true"/>
  <property name="java.security.krb5.realm" value="ELYTRON.ORG"/>
  <property name="java.security.krb5.kdc" value="kdc.elytron.org"/>
</system-properties>
```

Escolha uma das seguintes opções de migração:

- [Migração da Autenticação HTTP Kerberos.](#)
- [Migração da Autenticação SASL Kerberos Remoting.](#)

Migração da autenticação Kerberos para o Elytron [Esta é uma tradução automática]

Nas configurações de segurança herdadas, você pode definir um domínio de segurança para habilitar a autenticação SPNEGO para a interface de gerenciamento HTTP da seguinte maneira.

Exemplo: Ativar a autenticação SPNEGO para a interface de gerenciamento de HTTP [Esta é uma tradução automática]

```
/core-service=management/security-realm=Kerberos:add
/core-service=management/security-realm=Kerberos/server-
identity=kerberos:add
/core-service=management/security-realm=Kerberos/server-
identity=kerberos/keytab=HTTP\test-
server.elytron.org@ELYTRON.ORG:add(path=/path/to/test-server.keytab,
debug=true)
/core-service=management/security-
realm=Kerberos/authentication=kerberos:add(remove-realm=true)
```

Exemplo: configuração do território de segurança Kerberos [Esta é uma tradução automática]

```
<security-realms>
  ...
  <security-realm name="Kerberos">
```

```

<server-identities>
  <kerberos>
    <keytab principal="HTTP/test-server.elytron.org@ELYTRON.ORG"
path="/path/to/test-server.keytab" debug="true"/>
  </kerberos>
</server-identities>
<authentication>
  <kerberos remove-realm="true"/>
</authentication>
</security-realm>
</security-realms>

```

Você também pode definir um par de domínios de segurança legados para permitir que os aplicativos usem a autenticação HTTP do Kerberos.

Exemplo: definir vários domínios de segurança [Esta é uma tradução automática]

```

# Define the first security domain
/subsystem=security/security-domain=host:add
/subsystem=security/security-domain=host/authentication=classic:add
/subsystem=security/security-domain=host/authentication=classic/login-
module=1:add(code=Kerberos, flag=Required, module-options={storeKey=true,
useKeyTab=true, principal=HTTP/test-server.elytron.org@ELYTRON.ORG,
keyTab=path/to/test-server.keytab, debug=true})

# Define the second SPNEGO security domain
/subsystem=security/security-domain=SPNEGO:add
/subsystem=security/security-domain=SPNEGO/authentication=classic:add
/subsystem=security/security-domain=SPNEGO/authentication=classic/login-
module=1:add(code=SPNEGO, flag=requisite, module-options={password-
stacking=useFirstPass, serverSecurityDomain=host})
/subsystem=security/security-domain=SPNEGO/authentication=classic/login-
module=1:write-attribute(name=module,
value=org.jboss.security.negotiation)
/subsystem=security/security-domain=SPNEGO/authentication=classic/login-
module=2:add(code=UsersRoles, flag=required, module-options={password-
stacking=useFirstPass, usersProperties= /path/to/kerberos/spnego-
users.properties, rolesProperties= /path/to/kerberos/spnego-
roles.properties, defaultUsersProperties= /path/to/kerberos/spnego-
users.properties, defaultRolesProperties= /path/to/kerberos/spnego-
roles.properties})

```

Exemplo: Configuração usando um par de domínios de segurança [Esta é uma tradução automática]

```

<subsystem xmlns="urn:jboss:domain:security:2.0">
  <security-domains>
    ...
    <security-domain name="host">
      <authentication>
        <login-module name="1" code="Kerberos" flag="required">
          <module-option name="storeKey" value="true"/>
          <module-option name="useKeyTab" value="true"/>
          <module-option name="principal" value="HTTP/test-
server.elytron.org@ELYTRON.ORG"/>

```



```

        <module-option name="keyTab" value="/path/to/test-
server.keytab"/>
        <module-option name="debug" value="true"/>
    </login-module>
</authentication>
</security-domain>
<security-domain name="SPNEGO">
    <authentication>
        <login-module name="1" code="SPNEGO" flag="requisite"
module="org.jboss.security.negotiation">
            <module-option name="password-stacking" value="useFirstPass"/>
            <module-option name="serverSecurityDomain" value="host"/>
        </login-module>
        <login-module name="2" code="UsersRoles" flag="required">
            <module-option name="password-stacking" value="useFirstPass"/>
            <module-option name="usersProperties"
value="/path/to/kerberos/spnego-users.properties"/>
            <module-option name="rolesProperties" value="
/path/to/kerberos/spnego-roles.properties"/>
            <module-option name="defaultUsersProperties" value="
/path/to/kerberos/spnego-users.properties"/>
            <module-option name="defaultRolesProperties" value="
/path/to/kerberos/spnego-roles.properties"/>
        </login-module>
    </authentication>
</security-domain>
</security-domains>
</subsystem>

```

Os aplicativos legados são então implantados referenciando o domínio de segurança SPNEGO e protegidos com o mecanismo SPNEGO.

Migração da autenticação Kerberos para o Elytron [Esta é uma tradução automática]

Tanto a interface de gerenciamento quanto os aplicativos podem ser protegidos no Elytron usando uma região de segurança e uma fábrica de segurança Kerberos.

1. Defina uma região de segurança a ser usada para carregar informações de identidade.

```

/subsystem=elytron/properties-realm=spnego-properties:add(users-
properties={path=path/to/spnego-users.properties, plain-text=true,
digest-realm-name=ELYTRON.ORG}, groups-properties=
{path=path/to/spnego-roles.properties})

```

2. Defina uma fábrica de segurança Kerberos que permita ao servidor carregar sua própria identidade Kerberos.

```

/subsystem=elytron/kerberos-security-factory=test-
server:add(path=path/to/test-server.keytab, principal=HTTP/test-
server.elytron.org@ELYTRON.ORG, debug=true)

```

3. Defina um domínio de segurança para reunir a política, bem como uma fábrica de autenticação HTTP para a política de autenticação.

```

/subsystem=elytron/security-domain=SPNEGODomain:add(default-
realm=spnego-properties, realms=[{realm=spnego-properties, role-

```



```

decoder=groups-to-roles]], permission-mapper=default-permission-
mapper)
/subsystem=elytron/http-authentication-factory=spnego-http-
authentication:add(security-domain=SPNEGODomain, http-server-
mechanism-factory=global,mechanism-configurations=[{mechanism-
name=SPNEGO, credential-security-factory=test-server}])

```

Isso resulta na seguinte configuração no **elytron** subsistema do arquivo de configuração do servidor.

Exemplo: Configuração do Elytron Migrado [Esta é uma tradução automática]

```

<subsystem xmlns="urn:wildfly:elytron:1.2" final-
providers="combined-providers" disallowed-providers="OracleUcrypto">
  ...
  <security-domains>
    ...
    <security-domain name="SPNEGODomain" default-realm="spnego-
properties" permission-mapper="default-permission-mapper">
      <realm name="spnego-properties" role-decoder="groups-to-
roles"/>
    </security-domain>
  </security-domains>
  <security-realms>
    ...
    <properties-realm name="spnego-properties">
      <users-properties path="path/to/spnego-users.properties"
digest-realm-name="ELYTRON.ORG" plain-text="true"/>
      <groups-properties path="path/to/spnego-roles.properties"/>
    </properties-realm>
  </security-realms>
  <credential-security-factories>
    <kerberos-security-factory name="test-server"
principal="HTTP/test-server.elytron.org@ELYTRON.ORG"
path="path/to/test-server.keytab" debug="true"/>
  </credential-security-factories>
  ...
  <http>
    ...
    <http-authentication-factory name="spnego-http-authentication"
http-server-mechanism-factory="global" security-
domain="SPNEGODomain">
      <mechanism-configuration>
        <mechanism mechanism-name="SPNEGO" credential-security-
factory="test-server"/>
      </mechanism-configuration>
    </http-authentication-factory>
    ...
  </http>
  ...
</subsystem>

```

4. Para proteger o aplicativo, defina um domínio de segurança do aplicativo no **undertow** subsistema para mapear domínios de segurança para este **http-authentication-factory**. A interface de gerenciamento HTTP pode ser atualizada para referenciar **http-**

authentication-factory definido nesta configuração. Este processo está documentado no [Migrar Autenticação e Autorização Baseadas em Propriedades para o Elytron](#) seção deste guia.

Migração da autenticação Kerberos para o Elytron [Esta é uma tradução automática]

É possível definir uma região de segurança legada para que a autenticação Kerberos / GSSAPI SASL seja usada para autenticação remota, como a interface de gerenciamento nativa.

Exemplo: Autenticação Kerberos para Comandos CLI de Gerenciamento Remoto [Esta é uma tradução automática]

```
/core-service=management/security-realm=Kerberos:add
/core-service=management/security-realm=Kerberos/server-
identity=kerberos:add
/core-service=management/security-realm=Kerberos/server-
identity=kerberos/keytab=remote/test-
server.elytron.org@ELYTRON.ORG:add(path=path/to/remote-test-server.keytab,
debug=true)
/core-service=management/security-
realm=Kerberos/authentication=kerberos:add(remove-realm=true)
```

Exemplo: configuração de domínio de segurança remota do Kerberos [Esta é uma tradução automática]

```
<management>
  <security-realms>
    ...
    <security-realm name="Kerberos">
      <server-identities>
        <kerberos>
          <keytab principal="remote/test-server.elytron.org@ELYTRON.ORG"
path="path/to/remote-test-server.keytab" debug="true"/>
        </kerberos>
      </server-identities>
      <authentication>
        <kerberos remove-realm="true"/>
      </authentication>
    </security-realm>
  </security-realms>
  ...
</management>
```

Migração da autenticação Kerberos para o Elytron [Esta é uma tradução automática]

As etapas para definir a configuração equivalente do Elytron são muito semelhantes às descritas [Migração da Autenticação HTTP Kerberos](#).

1. Defina uma região de segurança a ser usada para carregar informações de identidade.

```
/path=kerberos:add(relative-to=user.home, path=src/kerberos)
/subsystem=elytron/properties-realm=kerberos-properties:add(users-
properties={path=kerberos-users.properties, relative-to=kerberos,
digest-realm-name=ELYTRON.ORG}, groups-properties={path=kerberos-
groups.properties, relative-to=kerberos})
```

2. Defina a fábrica de segurança Kerberos para a identidade do servidor.

■

```
/subsystem=elytron/kerberos-security-factory=test-
server:add(relative-to=kerberos, path=remote-test-server.keytab,
principal=remote/test-server.elytron.org@ELYTRON.ORG)
```

3. Exemplo: Domínio de Segurança Elytron e Comandos de Configuração de Fábrica de Autenticação [Esta é uma tradução automática]

```
/subsystem=elytron/security-domain=KerberosDomain:add(default-
realm=kerberos-properties, realms=[{realm=kerberos-properties, role-
decoder=groups-to-roles}], permission-mapper=default-permission-
mapper)
/subsystem=elytron/sasl-authentication-factory=gssapi-
authentication-factory:add(security-domain=KerberosDomain, sasl-
server-factory=elytron, mechanism-configurations=[{mechanism-
name=GSSAPI, credential-security-factory=test-server}])
```

Isso resulta na seguinte configuração no **elytron** subsistema do arquivo de configuração do servidor.

```
<subsystem xmlns="urn:wildfly:elytron:1.2" final-providers="combined-
providers" disallowed-providers="OracleUcrypto">
  ...
  <security-domains>
    ...
    <security-domain name="KerberosDomain" default-realm="kerberos-
properties" permission-mapper="default-permission-mapper">
      <realm name="kerberos-properties" role-decoder="groups-to-roles"/>
    </security-domain>
  </security-domains>
  <security-realms>
    ...
    <properties-realm name="kerberos-properties">
      <users-properties path="kerberos-users.properties" relative-
to="kerberos" digest-realm-name="ELYTRON.ORG"/>
      <groups-properties path="kerberos-groups.properties" relative-
to="kerberos"/>
    </properties-realm>
  </security-realms>
  <credential-security-factories>
    <kerberos-security-factory name="test-server" principal="remote/test-
server.elytron.org@ELYTRON.ORG" path="remote-test-server.keytab" relative-
to="kerberos"/>
  </credential-security-factories>
  ...
  <sasl>
    ...
    <sasl-authentication-factory name="gssapi-authentication-factory"
sasl-server-factory="elytron" security-domain="KerberosDomain">
      <mechanism-configuration>
        <mechanism mechanism-name="GSSAPI" credential-security-
factory="test-server"/>
      </mechanism-configuration>
    </sasl-authentication-factory>
    ...
  </sasl>
</subsystem>
```

A interface de gerenciamento ou os conectores remotos podem agora ser atualizados para fazer referência ao factory de autenticação SASL.

Os dois exemplos de Elytron definidos aqui também podem ser combinados para usar um domínio de segurança e um domínio de segurança compartilhados e apenas usar fábricas de autenticação específicas de protocolo, cada uma referenciando sua própria fábrica de segurança Kerberos.

7.3.5. Migrar lojas compostas para Elytron [Esta é uma tradução automática]

Esta seção descreve como migrar uma [PicketBox](#) ou [domínio de segurança legado](#) configuração que usa vários armazenamentos de identidades para [Élitro](#). Ao usar o PicketBox ou os domínios de segurança legados, é possível definir uma configuração em que a autenticação é executada em um armazenamento de identidades enquanto as informações usadas para autorização são carregadas de um armazenamento diferente. Ao migrar para o Elytron, isso pode ser feito usando um domínio de segurança agregado.

Os exemplos a seguir executam a autenticação do usuário usando o **example-users.properties** arquivo de propriedades e, em seguida, consulte o LDAP para carregar as informações de grupo e função.



NOTA

As configurações mostradas são baseadas nos exemplos das seções a seguir, que fornecem informações adicionais sobre o histórico:

- [Migrar Autenticação e Autorização Baseadas em Propriedades para o Elytron](#) [Esta é uma tradução automática]
- [Migrar a configuração de autenticação LDAP para o Elytron](#)

Exemplo: Configuração do PicketBox LoginModule [Esta é uma tradução automática]

O domínio de segurança PicketBox para este cenário é configurado usando os seguintes comandos da CLI de gerenciamento.

Exemplo: comandos de configuração do PicketBox [Esta é uma tradução automática]

```
/subsystem=security/security-domain=application-security:add

/subsystem=security/security-domain=application-
security/authentication=classic:add(login-modules=[ {code=UsersRoles,
flag=Required, module-options={ password-stacking=useFirstPass,
usersProperties=file://${jboss.server.config.dir}/example-
users.properties}} {code=LdapExtended, flag=Required, module-options={
password-stacking=useFirstPass,
java.naming.factory.initial=com.sun.jndi.ldap.LdapCtxFactory,
java.naming.provider.url=ldap://localhost:10389,
java.naming.security.authentication=simple, bindDN="uid=admin,ou=system",
bindCredential=secret, baseCtxDN="ou=users,dc=group-to-
principal,dc=wildfly,dc=org", baseFilter="(uid={0})",
rolesCtxDN="ou=groups,dc=group-to-
principal,dc=wildfly,dc=org", roleFilter="(uniqueMember={1})",
roleAttributeID="uid" }]])
```

Isso resulta na seguinte configuração do servidor.

Exemplo: Configuração do Domínio de Segurança do PicketBox [Esta é uma tradução automática]

```

<security-domain name="application-security">
  <authentication>
    <login-module code="UsersRoles" flag="required">
      <module-option name="password-stacking" value="useFirstPass"/>
      <module-option name="usersProperties"
value="file://${jboss.server.config.dir}/example-users.properties"/>
    </login-module>
    <login-module code="LdapExtended" flag="required">
      <module-option name="password-stacking" value="useFirstPass"/>
      <module-option name="java.naming.factory.initial"
value="com.sun.jndi.ldap.LdapCtxFactory"/>
      <module-option name="java.naming.provider.url"
value="ldap://localhost:10389"/>
      <module-option name="java.naming.security.authentication"
value="simple"/>
      <module-option name="bindDN" value="uid=admin,ou=system"/>
      <module-option name="bindCredential" value="secret"/>
      <module-option name="baseCtxDN" value="ou=users,dc=group-to-
principal,dc=wildfly,dc=org"/>
      <module-option name="baseFilter" value="(uid={0})"/>
      <module-option name="rolesCtxDN" value="ou=groups,dc=group-to-
principal,dc=wildfly,dc=org"/>
      <module-option name="roleFilter" value="(uniqueMember={1})"/>
      <module-option name="roleAttributeID" value="uid"/>
    </login-module>
  </authentication>
</security-domain>

```

Veja [Configuração do domínio de segurança agregada Elytron](#) para saber como configurar um domínio de segurança agregado no **elytron** subsistema para realizar isso.

Exemplo: Comandos de Configuração do Reino de Segurança Legado [Esta é uma tradução automática]

A configuração de região de segurança legada para este cenário é configurada usando os seguintes comandos da CLI de gerenciamento.

Exemplo: Comandos de Configuração do Reino de Segurança Legado [Esta é uma tradução automática]

```

/core-service=management/ldap-
connection=MyLdapConnection:add(url="ldap://localhost:10389", search-
dn="uid=admin,ou=system", search-credential="secret")

/core-service=management/security-realm=ApplicationSecurity:add
/core-service=management/security-
realm=ApplicationSecurity/authentication=properties:add(path=example-
users.properties, relative-to=jboss.server.config.dir, plain-text=true)

batch
/core-service=management/security-
realm=ApplicationSecurity/authorization=ldap:add(connection=MyLdapConnecti
on)

```

```

/core-service=management/security-
realm=ApplicationSecurity/authorization=ldap/username-to-dn=username-
filter:add(attribute=uid, base-dn="ou=users,dc=group-to-
principal,dc=wildfly,dc=org")
/core-service=management/security-
realm=ApplicationSecurity/authorization=ldap/group-search=group-to-
principal:add(base-dn="ou=groups,dc=group-to-principal,dc=wildfly,dc=org",
iterative=true, prefer-original-connection=true, principal-
attribute=uniqueMember, search-by=DISTINGUISHED_NAME, group-name=SIMPLE,
group-name-attribute=uid)
run-batch

```

Isso resulta na seguinte configuração do servidor.

Exemplo: Comandos de Configuração do Reino de Segurança Legado [Esta é uma tradução automática]

```

<security-realms>
...
  <security-realm name="ApplicationSecurity">
    <authentication>
      <properties path="example-users.properties" relative-
to="jboss.server.config.dir" plain-text="true"/>
    </authentication>
    <authorization>
      <ldap connection="MyLdapConnection">
        <username-to-dn>
          <username-filter base-dn="ou=users,dc=group-to-
principal,dc=wildfly,dc=org" attribute="uid"/>
        </username-to-dn>
        <group-search group-name="SIMPLE" iterative="true" group-name-
attribute="uid">
          <group-to-principal search-by="DISTINGUISHED_NAME" base-
dn="ou=groups,dc=group-to-principal,dc=wildfly,dc=org" prefer-original-
connection="true">
            <membership-filter principal-attribute="uniqueMember"/>
          </group-to-principal>
        </group-search>
      </ldap>
    </authorization>
  </security-realm>
</security-realms>
<outbound-connections>
  <ldap name="MyLdapConnection" url="ldap://localhost:10389" search-
dn="uid=admin,ou=system" search-credential="secret"/>
</outbound-connections>

```

Veja [Configuração do domínio de segurança agregada Elytron](#) para saber como configurar um domínio de segurança agregado no **elytron** subsistema para realizar isso.

Exemplo: Configuração do Realm de Segurança LDAP [Esta é uma tradução automática]

A configuração equivalente do Elytron para este cenário é configurada usando os seguintes comandos da CLI de gerenciamento.

Exemplo: Comandos de Configuração do Elytron [Esta é uma tradução automática]


```

/subsystem=elytron/dir-context=ldap-
connection:add(url=ldap://localhost:10389,
principal="uid=admin,ou=system", credential-reference={clear-text=secret})

/subsystem=elytron/ldap-realm=ldap-realm:add(dir-context=ldap-connection,
direct-verification=true, identity-mapping={search-base-
dn="ou=users,dc=group-to-principal,dc=wildfly,dc=org", rdn-
identifier="uid", attribute-mapping=[{filter-base-dn="ou=groups,dc=group-
to-principal,dc=wildfly,dc=org", filter="(uniqueMember=
{1})", from="uid", to="Roles"}]})

/subsystem=elytron/properties-realm=application-properties:add(users-
properties={path=example-users.properties, relative-
to=jboss.server.config.dir, plain-text=true, digest-realm-
name="Application Security"})

/subsystem=elytron/aggregate-realm=combined-realm:add(authentication-
realm=application-properties, authorization-realm=ldap-realm)

/subsystem=elytron/security-domain=application-security:add(realms=
[{realm=combined-realm}], default-realm=combined-realm, permission-
mapper=default-permission-mapper)
/subsystem=elytron/http-authentication-factory=application-security-
http:add(http-server-mechanism-factory=global, security-
domain=application-security, mechanism-configurations=[{mechanism-
name=BASIC}])

```

Isso resulta na seguinte configuração do servidor.

Exemplo: Configuração Elytron [Esta é uma tradução automática]

```

<subsystem xmlns="urn:wildfly:elytron:1.2" final-providers="combined-
providers" disallowed-providers="OracleUcrypto">
  ...
  <security-domains>
    ...
    <security-domain name="application-security" default-realm="combined-
realm" permission-mapper="default-permission-mapper">
      <realm name="combined-realm"/>
    </security-domain>
  </security-domains>
  <security-realms>
    <aggregate-realm name="combined-realm" authentication-
realm="application-properties" authorization-realm="ldap-realm"/>
    ...
    <properties-realm name="application-properties">
      <users-properties path="example-users.properties" relative-
to="jboss.server.config.dir" digest-realm-name="Application Security"
plain-text="true"/>
    </properties-realm>
    <ldap-realm name="ldap-realm" dir-context="ldap-connection" direct-
verification="true">
      <identity-mapping rdn-identifier="uid" search-base-
dn="ou=users,dc=group-to-principal,dc=wildfly,dc=org">
        <attribute-mapping>

```

```

        <attribute from="uid" to="Roles" filter="(uniqueMember={1})"
filter-base-dn="ou=groups,dc=group-to-principal,dc=wildfly,dc=org"/>
    </attribute-mapping>
</identity-mapping>
</ldap-realm>
</security-realms>
...
<http>
    ...
    <http-authentication-factory name="application-security-http" http-
server-mechanism-factory="global" security-domain="application-security">
        <mechanism-configuration>
            <mechanism mechanism-name="BASIC"/>
        </mechanism-configuration>
    </http-authentication-factory>
    ...
</http>
...
<dir-contexts>
    <dir-context name="ldap-connection" url="ldap://localhost:10389"
principal="uid=admin,ou=system">
        <credential-reference clear-text="secret"/>
    </dir-context>
</dir-contexts>
</subsystem>

```

No **elytron** subsistema, um **aggregate-realm** foi definido que especifica quais domínios de segurança devem ser usados para autenticação e quais usar para decisões de autorização.

7.3.6. Migrar os domínios de segurança que usam o cache para Elytron [Esta é uma tradução automática]

Ao usar o PicketBox, é possível definir um domínio de segurança e ativar o armazenamento em cache da memória para seu acesso. Isso permite que você acesse os dados de identidade na memória e evita o acesso direto adicional ao armazenamento de identidades. É possível obter uma configuração semelhante com o Elytron. Esta seção descreve como configurar o armazenamento em cache do domínio de segurança ao usar o Elytron.

Exemplo: Configuração do Domínio de Segurança em Cache do PicketBox [Esta é uma tradução automática]

Os comandos a seguir mostram como configurar um domínio de segurança PicketBox que permite o armazenamento em cache.

Exemplo: Comandos do Domínio de Segurança em Cache do PicketBox [Esta é uma tradução automática]

```

/subsystem=security/security-domain=application-security:add(cache-
type=default)
/subsystem=security/security-domain=application-
security/authentication=classic:add(login-modules=[{code=LdapExtended,
flag=Required, module-options={
java.naming.factory.initial=com.sun.jndi.ldap.LdapCtxFactory,
java.naming.provider.url=ldap://localhost:10389,
java.naming.security.authentication=simple, bindDN="uid=admin,ou=system",
bindCredential=secret, baseCtxDN="ou=users,dc=group-to-

```



```
principal,dc=wildfly,dc=org", baseFilter="(uid={0})",
rolesCtxDN="ou=groups,dc=group-to-principal,dc=wildfly,dc=org",
roleFilter="(uniqueMember={1})", roleAttributeID="uid" }]])
```

Isso resulta na seguinte configuração do servidor.

Exemplo: Configuração do Domínio de Segurança em Cache do PicketBox [Esta é uma tradução automática]

```
<subsystem xmlns="urn:jboss:domain:security:2.0">
  <security-domains>
    ...
    <security-domain name="application-security" cache-type="default">
      <authentication>
        <login-module code="LdapExtended" flag="required">
          <module-option name="java.naming.factory.initial"
value="com.sun.jndi.ldap.LdapCtxFactory"/>
          <module-option name="java.naming.provider.url"
value="ldap://localhost:10389"/>
          <module-option name="java.naming.security.authentication"
value="simple"/>
          <module-option name="bindDN" value="uid=admin,ou=system"/>
          <module-option name="bindCredential" value="secret"/>
          <module-option name="baseCtxDN" value="ou=users,dc=group-to-
principal,dc=wildfly,dc=org"/>
          <module-option name="baseFilter" value="(uid={0})"/>
          <module-option name="rolesCtxDN" value="ou=groups,dc=group-to-
principal,dc=wildfly,dc=org"/>
          <module-option name="roleFilter" value="(uniqueMember={1})"/>
          <module-option name="roleAttributeID" value="uid"/>
        </login-module>
      </authentication>
    </security-domain>
  </security-domains>
</subsystem>
```



NOTA

Este comando e configuração resultante é semelhante ao exemplo mostrado em [Migrar a configuração de autenticação LDAP para o Elytron](#); no entanto, aqui o atributo **cache-type** é definido com um valor de **default**. O tipo de cache é um cache na memória. Ao usar o PicketBox, você também pode especificar um **cache-type** do **infinispan**, no entanto, este tipo não é suportado pelo Elytron.

Exemplo: Configuração do Domínio de Segurança em Cache do Elytron [Esta é uma tradução automática]

Siga as etapas abaixo para criar uma configuração semelhante que armazena em cache um domínio de segurança ao usar o Elytron.

1. Defina um domínio de segurança e envolva o domínio de segurança em um domínio de armazenamento em cache. O domínio de armazenamento em cache pode ser usado em um domínio de segurança e, posteriormente, em uma fábrica de autenticação.

Exemplo: Comandos de Configuração do Elytron Security Realm [Esta é uma tradução automática]

```

/subsystem=elytron/dir-context=ldap-
connection:add(url=ldap://localhost:10389,
principal="uid=admin,ou=system", credential-reference={clear-
text=secret})
/subsystem=elytron/ldap-realm=ldap-realm:add(dir-context=ldap-
connection, direct-verification=true, identity-mapping={search-base-
dn="ou=users,dc=group-to-principal,dc=wildfly,dc=org", rdn-
identifier="uid", attribute-mapping=[{filter-base-
dn="ou=groups,dc=group-to-principal,dc=wildfly,dc=org", filter="
(uniqueMember={1})", from="uid", to="Roles"}]})
/subsystem=elytron/caching-realm=cached-ldap:add(realm=ldap-realm)

```

2. Defina um domínio de segurança e uma fábrica de autenticação HTTP que use o **cached-ldap** reino definido na etapa anterior.

Exemplo: Domínio de Segurança Elytron e Comandos de Configuração de Fábrica de Autenticação [Esta é uma tradução automática]

```

/subsystem=elytron/security-domain=application-security:add(realms=
[{{realm=cached-ldap}}, default-realm=cached-ldap, permission-
mapper=default-permission-mapper)
/subsystem=elytron/http-authentication-factory=application-security-
http:add(http-server-mechanism-factory=global, security-
domain=application-security, mechanism-configurations=[{{mechanism-
name=BASIC}}])

```



NOTA

Nesta etapa, é importante referenciar o **caching-realm** em vez do reino original. Caso contrário, o cache será ignorado.

Esses comandos resultam nas seguintes adições à configuração do servidor.

Exemplo: Configuração do Domínio de Segurança em Cache do Elytron [Esta é uma tradução automática]

```

<subsystem xmlns="urn:wildfly:elytron:1.2" final-providers="combined-
providers" disallowed-providers="OracleUcrypto">
...
<security-domains>
...
<security-domain name="application-security" default-realm="cached-
ldap" permission-mapper="default-permission-mapper">
<realm name="cached-ldap"/>
</security-domain>
</security-domains>
...
<security-realms>
....
<ldap-realm name="ldap-realm" dir-context="ldap-connection" direct-
verification="true">
<identity-mapping rdn-identifier="uid" search-base-

```

```

dn="ou=users,dc=group-to-principal,dc=wildfly,dc=org">
  <attribute-mapping>
    <attribute from="uid" to="Roles" filter="(uniqueMember={1})"
filter-base-dn="ou=groups,dc=group-to-principal,dc=wildfly,dc=org"/>
  </attribute-mapping>
</identity-mapping>
</ldap-realm>
<cached-ldap realm="ldap-realm"/>
</security-realms>
...
<http>
  ...
  <http-authentication-factory name="application-security-http" http-
server-mechanism-factory="global" security-domain="application-security">
    <mechanism-configuration>
      <mechanism mechanism-name="BASIC"/>
    </mechanism-configuration>
  </http-authentication-factory>
  ...
</http>
...
<dir-contexts>
  <dir-context name="ldap-connection" url="ldap://localhost:10389"
principal="uid=admin,ou=system">
    <credential-reference clear-text="secret"/>
  </dir-context>
</dir-contexts>
...

```

7.3.7. Migre a Segurança do JACC para o Elytron [Esta é uma tradução automática]

Por padrão, o JBoss EAP usa o legado **security** subsistema para configurar o fornecedor e a fábrica de políticas do Java Authorization Contract for Containers (JACC). A configuração padrão é mapeada para implementações da PicketBox.

o **elytron** O subsistema fornece um provedor de políticas integrado com base na especificação JACC. Antes de configurar seu servidor para permitir que o Elytron gerencie configurações do JACC e outras políticas, você deve primeiro desabilitar o JACC no **security** subsistema usando o seguinte comando CLI de gerenciamento.

```
/subsystem=security:write-attribute(name=initialize-jacc, value=false)
```

Não fazer isso pode resultar no seguinte erro no log do servidor: **MSC000004: Failure during stop of service org.wildfly.security.policy: java.lang.StackOverflowError.**

Para obter informações sobre como ativar o JACC e definir um provedor de política JACC no **elytron** subsistema, consulte [Ativando o JACC usando o elytron Subsistema](#) no *Guia de desenvolvimento* para o JBoss EAP.

7.4. MIGRAR CLIENTES DE APLICATIVOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

7.4.1. Migrar uma configuração de cliente de nomeação para Elytron [Esta é uma tradução automática]

Esta seção descreve como migrar um aplicativo cliente que executa uma consulta JNDI remota usando um `org.jboss.naming.remote.client.InitialContext` classe, que é apoiado por um `org.jboss.naming.remote.client.InitialContextFactory` classe, para Elytron.

Os exemplos a seguir assumem que o `InitialContextFactory` classe é criada especificando propriedades para as credenciais do usuário e para o URL do provedor de nomeação ao qual ele se conecta.

Exemplo: InitialContext Código usado na liberação anterior [Esta é uma tradução automática]

```
Properties properties = new Properties();
properties.put(Context.INITIAL_CONTEXT_FACTORY,
"org.jboss.naming.remote.client.InitialContextFactory");
properties.put(Context.PROVIDER_URL, "http-remoting://127.0.0.1:8080");
properties.put(Context.SECURITY_PRINCIPAL, "bob");
properties.put(Context.SECURITY_CREDENTIALS, "secret");
InitialContext context = new InitialContext(properties);
Bar bar = (Bar) context.lookup("foo/bar");
...
```

Você pode escolher uma das seguintes abordagens de migração:

- Migrar o cliente de nomeação usando a abordagem do arquivo de configuração [Esta é uma tradução automática]
- Migrar o cliente de nomeação usando a abordagem programática [Esta é uma tradução automática]

7.4.1.1. Migrar o cliente de nomeação usando a abordagem do arquivo de configuração [Esta é uma tradução automática]

Migrar o cliente de nomeação usando a abordagem do arquivo de configuração [Esta é uma tradução automática]

1. Crie um `wildfly-config.xml` arquivo no aplicativo cliente **META-INF/** diretório. O arquivo deve conter as credenciais do usuário a serem usadas ao estabelecer uma conexão com o provedor de nomenclatura.

Exemplo: Configuração Equivalente Usando `owildfly-config.xml` Arquivo [Esta é uma tradução automática]

```
<configuration>
  <authentication-client xmlns="urn:elytron:1.0.1">
    <authentication-rules>
      <rule use-configuration="namingConfig">
        <match-host name="127.0.0.1"/>
      </rule>
    </authentication-rules>
    <authentication-configurations>
      <configuration name="namingConfig">
        <set-user-name name="bob"/>
      </configuration>
    </authentication-configurations>
  </authentication-client>
</configuration>
```

```

        <credentials>
            <clear-password password="secret"/>
        </credentials>
    </configuration>
</authentication-configurations>
</authentication-client>
</configuration>

```

2. Criar um **InitialContext** como no exemplo a seguir. Note que o **InitialContext** é apoiado pelo **org.wildfly.naming.client.WildFlyInitialContextFactory** classe.

Exemplo: InitialContext Código usado na liberação anterior [Esta é uma tradução automática]

```

Properties properties = new Properties();
properties.put(Context.INITIAL_CONTEXT_FACTORY, "org.wildfly.naming.c
lient.WildFlyInitialContextFactory");
properties.put(Context.PROVIDER_URL, "remote+http://127.0.0.1:8080");
InitialContext context = new InitialContext(properties);
Bar bar = (Bar) context.lookup("foo/bar");
...

```

7.4.1.2. Migrar o cliente de nomeação usando a abordagem programática [Esta é uma tradução automática]

Usando essa abordagem, você fornece as credenciais do usuário que são usadas para estabelecer uma conexão com o provedor de nomenclatura diretamente no código do aplicativo.

Migrar o cliente EJB usando a abordagem programática [Esta é uma tradução automática]

```

// Create the authentication configuration
AuthenticationConfiguration namingConfig =
AuthenticationConfiguration.empty().useName("bob").usePassword("secret");

// Create the authentication context
AuthenticationContext context =
AuthenticationContext.empty().with(MatchRule.ALL.matchHost("127.0.0.1"),
namingConfig);

// Create a callable that creates and uses an InitialContext
Callable<Void> callable = () -> {
    Properties properties = new Properties();

    properties.put(Context.INITIAL_CONTEXT_FACTORY, "org.wildfly.naming.client.
WildFlyInitialContextFactory");
    properties.put(Context.PROVIDER_URL, "remote+http://127.0.0.1:8080");
    InitialContext context = new InitialContext(properties);
    Bar bar = (Bar) context.lookup("foo/bar");
    ...
    return null;
};

```

```
// Use the authentication context to run the callable
context.runCallable(callable);
```

7.4.2. Migrar um cliente EJB para Elytron [Esta é uma tradução automática]

Este exemplo de migração assume que o aplicativo cliente está configurado para chamar um EJB implementado em um servidor remoto usando um **jboss-ejb-client.properties** Arquivo. Este arquivo, localizado no aplicativo cliente **META-INF/** diretório, contém as seguintes informações necessárias para conectar-se ao servidor remoto.

Exemplo: jboss-ejb-client.properties Arquivo [Esta é uma tradução automática]

```
remote.connectionprovider.create.options.org.xnio.Options.SSL_ENABLED=false
remote.connections=default
remote.connection.default.host=127.0.0.1
remote.connection.default.port = 8080
remote.connection.default.username=bob
remote.connection.default.password=secret
```

O cliente consulta o EJB e chama um de seus métodos usando código semelhante ao exemplo a seguir.

Exemplo: código do cliente que chama um EJB remoto [Esta é uma tradução automática]

```
// Create an InitialContext
Properties properties = new Properties();
properties.put(Context.URL_PKG_PREFIXES, "org.jboss.ejb.client.naming");
InitialContext context = new InitialContext(properties);

// Look up the EJB and invoke one of its methods
RemoteCalculator statelessRemoteCalculator = (RemoteCalculator)
context.lookup(
    "ejb:/ejb-remote-server-side//CalculatorBean!" +
    RemoteCalculator.class.getName());
int sum = statelessRemoteCalculator.add(101, 202);
```

Você pode escolher uma das seguintes abordagens de migração:

- Migrar o cliente EJB usando a abordagem do arquivo de configuração [Esta é uma tradução automática]
- Migrar o cliente EJB usando a abordagem programática [Esta é uma tradução automática]

7.4.2.1. Migrar o cliente EJB usando a abordagem do arquivo de configuração [Esta é uma tradução automática]

Migrar o cliente de nomeação usando a abordagem do arquivo de configuração [Esta é uma tradução automática]

1. Configurar um **wildfly-config.xml** arquivo no aplicativo cliente **META-INF/** diretório. O arquivo deve conter as credenciais do usuário a serem usadas ao estabelecer uma conexão com o provedor de nomenclatura.

Exemplo: Configuração Equivalente Usando owildfly-config.xml Arquivo [Esta é uma tradução automática]

```

<configuration>
  <authentication-client xmlns="urn:elytron:1.0.1">
    <authentication-rules>
      <rule use-configuration="ejbConfig">
        <match-host name="127.0.0.1"/>
      </rule>
    </authentication-rules>
    <authentication-configurations>
      <configuration name="ejbConfig">
        <set-user-name name="bob"/>
        <credentials>
          <clear-password password="secret"/>
        </credentials>
      </configuration>
    </authentication-configurations>
  </authentication-client>
  <jboss-ejb-client xmlns="urn:jboss:wildfly-client-ejb:3.0">
    <connections>
      <connection uri="remote+http://127.0.0.1:8080" />
    </connections>
  </jboss-ejb-client>
</configuration>

```

2. Criar um **InitialContext** como no exemplo a seguir. Note que o **InitialContext** é apoiado pelo **org.wildfly.naming.client.WildFlyInitialContextFactory** classe.

Exemplo: InitialContext Código usado na liberação anterior [Esta é uma tradução automática]

```

// Create an InitialContext
Properties properties = new Properties();
properties.put(Context.INITIAL_CONTEXT_FACTORY, "org.wildfly.naming.c
lient.WildFlyInitialContextFactory");
InitialContext context = new InitialContext(properties);

// Look up an EJB and invoke one of its methods
// Note that this code is the same as before
RemoteCalculator statelessRemoteCalculator = (RemoteCalculator)
context.lookup(
    "ejb:/ejb-remote-server-side//CalculatorBean!" +
    RemoteCalculator.class.getName());
int sum = statelessRemoteCalculator.add(101, 202);----

```

3. Exemplo: **jboss-ejb-client.properties** Arquivo de propriedades [Esta é uma tradução automática]

7.4.2.2. Migrar o cliente EJB usando a abordagem programática [Esta é uma tradução automática]

Usando essa abordagem, você fornece as informações necessárias para se conectar ao servidor remoto diretamente no código do aplicativo.

Migrar o cliente EJB usando a abordagem programática [Esta é uma tradução automática]

```
// Create the authentication configuration
AuthenticationConfiguration ejbConfig =
AuthenticationConfiguration.empty().useName("bob").usePassword("secret");

// Create the authentication context
AuthenticationContext context =
AuthenticationContext.empty().with(MatchRule.ALL.matchHost("127.0.0.1"),
ejbConfig);

// Create a callable that invokes the EJB
Callable<Void> callable = () -> {

    // Create an InitialContext
    Properties properties = new Properties();
    properties.put(Context.INITIAL_CONTEXT_FACTORY,
"org.wildfly.naming.client.WildFlyInitialContextFactory");
    properties.put(Context.PROVIDER_URL, "remote+http://127.0.0.1:8080");
    InitialContext context = new InitialContext(properties);

    // Look up the EJB and invoke one of its methods
    // Note that this code is the same as before
    RemoteCalculator statelessRemoteCalculator = (RemoteCalculator)
context.lookup(
    "ejb:/ejb-remote-server-side//CalculatorBean!" +
RemoteCalculator.class.getName());
    int sum = statelessRemoteCalculator.add(101, 202);
    ...
    return null;
};

// Use the authentication context to run the callable
context.runCallable(callable);
```

Exemplo: **jboss-ejb-client.properties** Arquivo de propriedades [Esta é uma tradução automática]

7.5. MIGRAR CONFIGURAÇÕES SSL [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

7.5.1. Migrar uma configuração SSL simples para o Elytron [Esta é uma tradução automática]

Se você protegeu conexões HTTP com o servidor do JBoss EAP usando uma região de segurança, pode migrar essa configuração para o Elytron usando as informações fornecidas nesta seção.

Os exemplos a seguir presumem que você tenha os seguintes **keystore** configurado no **security-realm**.

Exemplo: Configuração SSL Usando um Keystore de Realm de Segurança [Esta é uma tradução automática]

■


```
<security-realm name="ApplicationRealm">
  <server-identities>
    <ssl>
      <keystore path="server.keystore" relative-
to="jboss.server.config.dir" keystore-password="keystore_password"
alias="server" key-password="key_password" />
    </ssl>
  </server-identities>
</security-realm>
```

Siga os passos abaixo para obter a mesma configuração usando o Elytron.

1. Crie um **key-store** no **elytron** subsistema que especifica o local do keystore e a senha pela qual ele é criptografado. Este comando assume que o keystore foi gerado usando o comando `keytool` e seu tipo é **JKS**.

```
/subsystem=elytron/key-
store=LocalhostKeyStore:add(path=server.keystore,relative-
to=jboss.server.config.dir,credential-reference={clear-
text="keystore_password"},type=JKS)
```

2. Crie um **key-manager** no **elytron** subsistema que especifica o **key-store** definido na etapa anterior, o alias e a senha da chave.

```
/subsystem=elytron/key-manager=LocalhostKeyManager:add(key-
store=LocalhostKeyStore,alias-filter=server,credential-reference=
{clear-text="key_password"})
```

3. Crie um **server-ssl-context** no **elytron** subsistema que referencia o **key-manager** que foi definido na etapa anterior.

```
/subsystem=elytron/server-ssl-context=LocalhostSslContext:add(key-
manager=LocalhostKeyManager)
```

4. Mude o **https-listener** do legado **security-realm** para o recém-criado Elytron **ssl-context**.

```
batch
/subsystem=undertow/server=default-server/https-
listener=https:undefine-attribute(name=security-realm)
/subsystem=undertow/server=default-server/https-
listener=https:write-attribute(name=ssl-
context,value=LocalhostSslContext)
run-batch
```

5. Recarregue o servidor.

```
recarregar
```

Isso resulta nas seguintes **elytron** configuração do subsistema no arquivo de configuração do servidor.

```
<subsystem xmlns="urn:wildfly:elytron:1.2" ...>
```

```

...
<tls>
  <key-stores>
    <key-store name="LocalhostKeyStore">
      <credential-reference clear-text="keystore_password"/>
      <implementation type="JKS"/>
      <file path="server.keystore" relative-
to="jboss.server.config.dir"/>
    </key-store>
  </key-stores>
  <key-managers>
    <key-manager name="LocalhostKeyManager" key-
store="LocalhostKeyStore" alias-filter="server">
      <credential-reference clear-text="key_password"/>
    </key-manager>
  </key-managers>
  <server-ssl-contexts>
    <server-ssl-context name="LocalhostSslContext" key-
manager="LocalhostKeyManager"/>
  </server-ssl-contexts>
</tls>
</subsystem>

```

Isso resulta nas seguintes **undertow** configuração do subsistema no arquivo de configuração do servidor.

```

<https-listener name="https" socket-binding="https" ssl-
context="LocalhostSslContext" enable-http2="true"/>

```

Para mais informações, veja [Subsistema Elytron](#) e [Como proteger as interfaces de gerenciamento](#) dentro *Como configurar a segurança do servidor* para o JBoss EAP.

7.5.2. Migrar Autenticação SSL de Cliente-Certificado para Elytron [Esta é uma tradução automática]

Se você configurou a autenticação SSL do Client-Cert usando um truststore de security realm, poderá migrar essa configuração para o Elytron usando as informações fornecidas nesta seção.

Os passos abaixo assumem que você tem o seguinte **truststore** configurado no **security-realm**.

Exemplo: Configuração SSL Usando o Truststore de Realm de Segurança [Esta é uma tradução automática]

```

<security-realm name="ApplicationRealm">
  <server-identities>
    <ssl>
      <keystore path="server.keystore" relative-
to="jboss.server.config.dir" keystore-password="keystore_password"
alias="server" key-password="key_password" />
    </ssl>
  </server-identities>
  <authentication>
    <truststore path="server.truststore" relative-
to="jboss.server.config.dir" keystore-password="truststore_password" />
    <local default-user="$local"/>
  </authentication>
</security-realm>

```

```

    <properties path="application-users.properties" relative-
to="jboss.server.config.dir"/>
  </authentication>
  <authorization>
    <properties path="application-roles.properties" relative-
to="jboss.server.config.dir"/>
  </authorization>
</security-realm>

```



IMPORTANTE

As etapas abaixo apenas fornecem configuração para impedir que usuários sem um certificado válido e uma chave privada acessem o servidor. Eles não configuram a identidade do usuário para autenticação no servidor. Supõe-se que você já tenha configurado a autenticação HTTP CLIENT-CERT e a autenticação SASL externa para autenticação de identidade do usuário.

Siga estas etapas para configurar o servidor para impedir que usuários sem um certificado válido e uma chave privada acessem o servidor usando o Elytron.

1. Crie um **key-store** no **elytron** subsistema que especifica o local do keystore e a senha pela qual ele é criptografado. Este comando assume que o keystore foi gerado usando o comando `keytool` e seu tipo é **JKS**.

```

/subsystem=elytron/key-
store=LocalhostKeyStore:add(path=server.keystore,relative-
to=jboss.server.config.dir,credential-reference={clear-
text="keystore_password"},type=JKS)

```

2. Crie um **key-store** no **elytron** subsistema que especifica o local do armazenamento confiável e a senha pela qual ele é criptografado. Este comando assume que o keystore foi gerado usando o comando `keytool` e seu tipo é **JKS**.

```

/subsystem=elytron/key-
store=TrustStore:add(path=server.truststore,relative-
to=jboss.server.config.dir,credential-reference={clear-
text="truststore_password"},type=JKS)

```

3. Crie um **key-manager** no **elytron** subsistema que especifica o definido anteriormente **LocalhostKeyStore** keystore, o alias e a senha da chave.

```

/subsystem=elytron/key-manager=LocalhostKeyManager:add(key-
store=LocalhostKeyStore,alias-filter=server,credential-reference=
{clear-text="key_password"})

```

4. Crie um **trust-manager** no **elytron** subsistema que especifica o **key-store** do armazenamento confiável criado anteriormente.

```

/subsystem=elytron/trust-manager=TrustManager:add(key-
store=TrustStore)

```

5. Crie um **server-ssl-context** no **elytron** subsistema que faz referência ao anteriormente definido **key-manager**, define o **trust-manager** atributo e permite a autenticação do cliente.

```
/subsystem=elytron/server-ssl-context=LocalhostSslContext:add(key-
manager=LocalhostKeyManager,trust-manager=TrustManager,need-client-
auth=true)
```

6. Mude o **https-listener** do legado **security-realm** para o recém-criado Elytron **ssl-context**.

```
batch
/subsystem=undertow/server=default-server/https-
listener=https:undefine-attribute(name=security-realm)
/subsystem=undertow/server=default-server/https-
listener=https:write-attribute(name=ssl-
context,value=LocalhostSslContext)
run-batch
```

7. Recarregue o servidor.

```
recarregar
```

Isso resulta nas seguintes **elytron** configuração do subsistema no arquivo de configuração do servidor.

```
<subsystem xmlns="urn:wildfly:elytron:1.2" ...>
  ...
  <tls>
    <key-stores>
      <key-store name="LocalhostKeyStore">
        <credential-reference clear-text="keystore_password"/>
        <implementation type="JKS"/>
        <file path="server.keystore" relative-
to="jboss.server.config.dir"/>
      </key-store>
      <key-store name="TrustStore">
        <credential-reference clear-text="truststore_password"/>
        <implementation type="JKS"/>
        <file path="server.truststore" relative-
to="jboss.server.config.dir"/>
      </key-store>
    </key-stores>
    <key-managers>
      <key-manager name="LocalhostKeyManager" key-
store="LocalhostKeyStore" alias-filter="server">
        <credential-reference clear-text="key_password"/>
      </key-manager>
    </key-managers>
    <trust-managers>
      <trust-manager name="TrustManager" key-store="TrustStore"/>
    </trust-managers>
    <server-ssl-contexts>
      <server-ssl-context name="LocalhostSslContext" need-client-
auth="true" key-manager="LocalhostKeyManager" trust-
manager="TrustManager"/>
    </server-ssl-contexts>
  </tls>
</subsystem>
```

```
        </server-ssl-contexts>  
    </tls>  
</subsystem>
```

Isso resulta nas seguintes **undertow** configuração do subsistema no arquivo de configuração do servidor.

```
<https-listener name="https" socket-binding="https" ssl-  
context="LocalhostSslContext" enable-http2="true"/>
```

Para mais informações, veja [Subsistema Elytron](#) e [Usando um contexto SSL-cliente](#) dentro *Como configurar a segurança do servidor* para o JBoss EAP.

CAPÍTULO 8. MIGRANDO A PARTIR DE VERSÕES MAIS ANTIGAS DO JBOSS EAP [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

8.1. MIGRANDO DO JBOSS EAP 5 PARA O JBOSS EAP 7 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Este guia concentra-se nas alterações necessárias para executar e implementar com êxito os aplicativos JBoss EAP 6 no JBoss EAP 7. Se você planeja migrar seus aplicativos diretamente do JBoss EAP 5 para o JBoss EAP 7, há uma série de recursos disponíveis para ajudá-lo planejar e executar sua migração. Sugerimos que você adote a seguinte abordagem.

1. Veja [Resumo das alterações feitas em cada release](#) neste guia para uma visão geral rápida e de alto nível das mudanças feitas em cada versão do JBoss EAP.
2. Leia o JBoss EAP 6 [Guia de Migração](#) e este guia para se familiarizar com o conteúdo de cada um.
3. Use o [Referência de atualização de componentes do JBoss EAP 5](#) como uma referência rápida para informações de migração sobre componentes e recursos específicos.
4. O Red Hat Application Migration Toolkit baseado em regras continua a adicionar regras para ajudá-lo a migrar diretamente do JBoss EAP 5 para o JBoss EAP 7. Você pode usar essas ferramentas para analisar seu aplicativo e gerar relatórios detalhados sobre as mudanças necessárias para migrar para o JBoss EAP. 7. Para mais informações, consulte [Use o Red Hat Application Migration Toolkit para analisar aplicativos para migração](#).
5. o [Base de Conhecimento do Portal do Cliente](#) atualmente contém artigos e soluções para ajudar com a migração do JBoss EAP 5 para o JBoss EAP 6. Existem planos em vigor para adicionar conteúdo adicional para migração do JBoss EAP 5 para o JBoss EAP 7 ao longo do tempo.

8.2. RESUMO DAS ALTERAÇÕES FEITAS EM CADA LANÇAMENTO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Antes de planejar a sua migração, você deve estar ciente das alterações feitas no JBoss EAP 6 e no JBoss EAP 7.

o [Guia de migração do JBoss EAP 6](#) cobre as mudanças que foram feitas entre o JBoss EAP 5 e o JBoss EAP 6. O seguinte é uma lista condensada das mudanças mais significativas feitas no JBoss EAP 6.

- Implementado uma nova arquitetura construída no contêiner de serviço modular
- Uma implementação certificada da especificação Java Enterprise Edition 6
- Introduziu gerenciamento de domínio, nova configuração de implantação e nova estrutura de diretórios de arquivo e scripts
- Padronizado em novos namespaces JNDI portáteis

Veja [Rever o que há de novo e diferente no JBoss EAP 6](#) no JBoss EAP 6 [Guia de Migração](#) para uma lista detalhada das alterações feitas nesse lançamento.

O JBoss EAP 7 é construído na mesma estrutura modular como o JBoss EAP 6 e inclui o mesmo gerenciamento de domínio, configuração de implantação, estrutura de diretório de arquivos e scripts. Ele também usa os mesmos espaço de nomes JNDI padronizados. Porém, o JBoss EAP 7 introduz as seguintes alterações.

- Adiciona suporte para a especificação Java Enterprise Edition 7
- Substituição do subsistema web com Undertow [Esta é uma tradução automática]
- Substitui a implementação JacORB IIOP por um ramificação downstream do OpenJDK ORB
- Inclui o Apache ActiveMQ Artemis como o novo provedor de mensagens
- Remove o **cmp**, **jaxr**, e **threads** subsistemas
- Remove o suporte para beans de entidade EJB

Para obter uma lista mais completa de alterações, consulte [Rever o que há de novo no JBoss EAP 7](#)

8.3. REVISE O CONTEÚDO NOS GUIAS DE MIGRAÇÃO [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Revise todo o conteúdo do Guia de Migração para cada lançamento conhecer os recursos que foram adicionados ou preteridos e para entender a configuração do servidor e as alterações de aplicativo necessárias para executar os aplicativos existentes para a versão.

Como a arquitetura subjacente não foi alterada entre o JBoss EAP 6 e o JBoss EAP 7, muitas das alterações documentadas no JBoss EAP 6 [Guia de Migração](#) ainda se aplicam. Por exemplo, mudanças documentadas em [Alterações exigidas pela maioria das aplicações](#) estão relacionados às mudanças arquitetônicas subjacentes feitas no JBoss EAP 6, que ainda se aplicam a este lançamento. A mudança para o novo sistema de carregamento de classe modular é significativa e afeta o empacotamento e as dependências de quase todos os aplicativos do JBoss EAP 5. Muitas das alterações listadas [Alterações dependentes de sua arquitetura de aplicativos e componentes](#) também ainda são válidas. No entanto, como o JBoss EAP 7 substituiu o servidor da Web, o ORB e o provedor de mensagens, **cmp**, **threads**, e **jaxr** subsistemas e suporte removido para beans de entidade EJB, você deve consultar este guia para quaisquer alterações relacionadas a essas áreas de componentes. Preste especial atenção ao [Alterações na configuração do servidor](#) e [Alterações na migração de aplicativos](#) detalhado neste guia antes de começar.

8.4. REFERÊNCIA DE ATUALIZAÇÃO DO COMPONENTE JBOSS EAP 5 [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Use a tabela a seguir para encontrar informações sobre como migrar um recurso ou componente específico do JBoss EAP 5 para o JBoss EAP 7.1.

JBoss EAP 5 Recurso ou componente	Resumo de alterações e Onde encontrar informações de migração
---	--

JBoss EAP 5 Recurso ou componente	Resumo de alterações e Onde encontrar informações de migração
Alterações na migração dos aplicativos [Esta é uma tradução automática]	<p>No JBoss EAP 6, a estrutura de carregamento de classes hierárquica anterior foi substituída por uma arquitetura modular baseada nos JBoss Modules. A embalagem do aplicativo também mudou devido à nova estrutura de carregamento de classe modular. Esta arquitetura ainda é usada no JBoss EAP 7. Para informações sobre a nova arquitetura modular, veja o seguinte capítulo no JBoss EAP 7.1 <i>Guia de desenvolvimento</i>.</p> <ul style="list-style-type: none"> • Carregamento de Classe e Módulos <p>Para obter informações sobre como atualizar e reempacotar aplicativos para a nova arquitetura modular, consulte a seção a seguir no JBoss EAP 6 <i>Guia de Migração</i>.</p> <ul style="list-style-type: none"> • Mudanças no Carregamento de Classe
Alterações na migração dos aplicativos [Esta é uma tradução automática]	<p>Devido às mudanças no JBoss EAP 6 para usar o carregamento de classe modular, você pode precisar criar ou modificar um ou mais arquivos de configuração do aplicativo para adicionar dependências ou impedir o carregamento de dependências automáticas. Isso não mudou no JBoss EAP 7. Para detalhes, veja a seção a seguir no JBoss EAP 6 <i>Guia de Migração</i>.</p> <ul style="list-style-type: none"> • Alterações no Arquivo de Configuração
Cache e Infinispan	<p>O JBoss Cache foi substituído pelo Infinispan para uso interno pelo servidor apenas no JBoss EAP 6. Veja as seções a seguir no JBoss EAP 6 <i>Guia de Migração</i> para obter informações sobre como substituir o JBoss Cache no código do aplicativo.</p> <ul style="list-style-type: none"> • Alterações no Cache <p>A estratégia de cache Infinispan e as alterações de configuração do JBoss EAP 7 estão documentadas na seção seguinte deste guia.</p> <ul style="list-style-type: none"> • Alterações na configuração do servidor Infinispan
Configurando um Adaptador de Recursos JMS Genérico [Esta é uma tradução automática]	<p>O JBoss EAP 6 consolidou a configuração de fontes de dados e adaptadores de recursos em um único arquivo, e isso ainda é verdade no JBoss EAP 7. Veja a seção a seguir no JBoss EAP 6 <i>Guia de Migração</i> Para maiores informações.</p> <ul style="list-style-type: none"> • Mudanças na configuração da fonte de dados e do adaptador de recursos

JBoss EAP 5 Recurso ou componente	Resumo de alterações e Onde encontrar informações de migração
Migrar configurações de planos de implementação [Esta é uma tradução automática]	<p>No JBoss EAP 6, a estrutura de diretórios, scripts e configuração de implantação foram alterados. Essas mudanças ainda são válidas no JBoss EAP 7. Veja a seção a seguir do JBoss EAP 6 <i>Guia de Migração</i> Para maiores informações.</p> <ul style="list-style-type: none"> • Rever o que há de novo e diferente no JBoss EAP 6
EJB	<p>A especificação do Java EE 7 tornou os recursos do EJB 2.xe anteriores opcionais, portanto, é altamente recomendado que você reescreva o código do aplicativo para usar a especificação EJB 3.xe o JPA. Para obter informações sobre recursos e mudanças reprovados necessários para executar o EJB 2.x, consulte a seção a seguir no JBoss EAP 6 <i>Guia de Migração</i>.</p> <ul style="list-style-type: none"> • EJB 2.xe alterações anteriores <p>No JBoss EAP 6, o cache EJB com monitoração de estado e o tamanho do conjunto de beans de sessão stateless são configurados no ejb3 subsistema do arquivo de configuração do servidor. o jboss-ejb3.xml descritor de implantação substitui o jboss.xml arquivo descritor de implantação. Para mais informações sobre estas mudanças, veja a seção seguinte no JBoss EAP 6 <i>Guia de Migração</i>.</p> <ul style="list-style-type: none"> • Alterações no EJB <p>O conector ea porta remotos padrão foram alterados no JBoss EAP 7. Para obter mais informações sobre isso e as alterações na configuração do servidor, consulte as seções a seguir deste guia.</p> <ul style="list-style-type: none"> • Alterações na Configuração do Servidor EJB • Migrar o código do cliente EJB <p>Beans de entidade EJB não são suportados no JBoss EAP 7. Para obter informações sobre como migrar beans de entidade para o JPA, consulte a seção a seguir deste guia.</p> <ul style="list-style-type: none"> • Migrar Beans de Entidade para JPA

JBoss EAP 5 Recurso ou componente	Resumo de alterações e Onde encontrar informações de migração
<p>Alterações de migração JPA e Hibernate [Esta é uma tradução automática]</p>	<p>No JBoss EAP 6, o Hibernate foi atualizado da versão 3 para a versão 4. Esta versão do JBoss EAP também implementou a especificação JPA 2.0 e as alterações foram feitas nas propriedades de persistência JPA. Para obter informações sobre como modificar seu aplicativo para essas mudanças, consulte a seção a seguir no JBoss EAP 6 <i>Guia de Migração</i>.</p> <ul style="list-style-type: none"> • Alterações no Hibernate e JPA <p>O JBoss EAP 7 implementa o JPA 2.1 e inclui o Hibernate 5. Ele também atualiza o Hibernate Search da versão 4.6.x para a versão 5.5.x. Outras alterações incluem remoção de suporte para beans de entidade EJB e atualizações adicionais para propriedades de persistência JPA. Para obter informações sobre como essas alterações afetam seus aplicativos, consulte as seções a seguir neste guia.</p> <ul style="list-style-type: none"> • Mudanças de migração do Hibernate e do JPA • Alterações na Pesquisa do Hibernate • Migrar Beans de Entidade para JPA • Mudanças de propriedade de persistência do JPA
<p>Alterações de aplicativos JAX-RS e RESTEasy [Esta é uma tradução automática]</p>	<p>O JBoss EAP 6 empacotou o RESTEasy 2, que configurou automaticamente o RESTEasy e exigiu mudanças na configuração do aplicativo. Veja a seção a seguir no JBoss EAP 6 <i>Guia de Migração</i> para informação.</p> <ul style="list-style-type: none"> • Mudanças no JAX-RS e no RESTEasy <p>O JBoss EAP 7 inclui o RESTEasy 3 e muitas classes foram descontinuadas. A versão do Jackson mudou da versão 1.9.9 para a versão 2.6.3 ou superior. Para obter detalhes sobre essas alterações, consulte a seção a seguir deste guia.</p> <ul style="list-style-type: none"> • Mudanças no aplicativo JAX-RS e RESTEasy
<p>JBoss AOP</p>	<p>O JBoss AOP (Aspect Oriented Programming) foi removido no JBoss EAP 6. Para obter informações sobre como refatorar aplicativos que usam o JBoss AOP, veja a seção a seguir no JBoss EAP 6 <i>Guia de Migração</i>.</p> <ul style="list-style-type: none"> • Mudanças no JBoss AOP

JBoss EAP 5 Recurso ou componente	Resumo de alterações e Onde encontrar informações de migração
Alterações de Canais JGroups [Esta é uma tradução automática]	<p>A maneira como você habilita o clustering e especifica os endereços de bind alterados no JBoss EAP 6. Veja a seção a seguir no JBoss EAP 6 <i>Guia de Migração</i> Para maiores informações.</p> <ul style="list-style-type: none"> • Alterações de Cluster <p>No JBoss EAP 7, o JGroups agora usa como padrão uma interface de rede privada em vez de uma interface de rede pública e também introduz <channel> elementos para o jgroups subsistema. O JBoss EAP 7 também inclui a implementação Undertow mod_cluster, introduz uma nova API para construir serviços singleton e outros novos recursos de clustering. Essas alterações estão documentadas nas seções a seguir deste guia.</p> <ul style="list-style-type: none"> • Mudanças na configuração do servidor JGroups • Alterações no Cluster de Aplicativos
JNDI	<p>O JBoss EAP 6 implementou um novo namespace JNDI global padronizado e uma série de namespaces relacionados que mapeiam para os vários escopos de um aplicativo Java EE. Veja a seção seguinte do JBoss EAP 6 <i>Guia de Migração</i> para obter informações sobre alterações de aplicativo necessárias para usar as novas regras de espaço de nomes JNDI.</p> <ul style="list-style-type: none"> • Alterações no JNDI
JSF	<p>O JBoss EAP 6 incluiu o JSF 2.0 e permitiu que você configurasse seu aplicativo para usar uma versão mais antiga. Isso não é mais possível no JBoss EAP 7, que agora inclui o JSF 2.2. Veja a seção a seguir neste guia para mais informações.</p> <ul style="list-style-type: none"> • Alterações no Código do JavaServer Faces (JSF)
Alterações de registro [Esta é uma tradução automática]	<p>O JBoss EAP 6 introduziu uma nova estrutura do JBoss Logging que ainda é usada no JBoss EAP 7. Os aplicativos que usam estruturas de criação de log de terceiros podem ser impactados pelas alterações de carregamento de classe modular. Revise a seção a seguir no JBoss EAP 6 <i>Guia de Migração</i> para obter informações sobre essas alterações.</p> <ul style="list-style-type: none"> • Alterações de registro <p>No JBoss EAP 7, anotações no org.jboss.logging O pacote agora está obsoleto, o que afeta o código-fonte e os Maven GAVs (groupId: artifactId: version). Os prefixos para todas as mensagens de log também foram alterados. Para obter mais informações sobre essas alterações, consulte as seções a seguir deste guia.</p> <ul style="list-style-type: none"> • Mudanças no registro do JBoss • Registrando Alterações no Prefixo de Mensagem

JBoss EAP 5 Recurso ou componente	Resumo de alterações e Onde encontrar informações de migração
Mensagens e JMS	<p>No JBoss EAP 6, o HornetQ substituiu o JBoss Messaging como a implementação padrão do JMS. Então, no JBoss EAP 7, o ActiveMQ Artemis substituiu o HornetQ como o provedor de mensagens embutido.</p> <p>A melhor abordagem para migrar sua configuração de mensagens é começar com a configuração do servidor padrão do JBoss EAP 7 e usar o seguinte guia para aplicar suas alterações de configuração de mensagens atuais.</p> <ul style="list-style-type: none"> • Configurando o Messaging para o JBoss EAP 7.1 <p>Se você quiser entender as mudanças necessárias para passar do JBoss Messaging para o HornetQ, revise a seção seguinte do JBoss EAP 6 <i>Guia de Migração</i>.</p> <ul style="list-style-type: none"> • Mudanças do HornetQ <p>Em seguida, analise as seguintes informações sobre como migrar a configuração do HornetQ e os dados de mensagens relacionadas neste guia.</p> <ul style="list-style-type: none"> • Alterações na configuração do servidor de mensagens • Alterações no aplicativo de mensagens
ORB	<p>No JBoss EAP 6, a configuração do JacORB foi movida do <i>EAP_HOME/server/production/conf/jacorb.properties</i> arquivo para o arquivo de configuração do servidor. O JBoss EAP 7 substituiu a implementação JacORB IIOP por uma ramificação downstream do OpenJDK ORB.</p> <p>A melhor abordagem para migrar sua configuração do ORB é começar com a configuração do servidor padrão do JBoss EAP 7 e usar a seguinte seção no JBoss EAP 7.1 <i>Guia de configuração</i> para aplicar as alterações da sua configuração atual do ORB.</p> <ul style="list-style-type: none"> • Configuração do ORB

JBoss EAP 5 Recurso ou componente	Resumo de alterações e Onde encontrar informações de migração
Invocação Remota	<p>Uma nova API de cliente EJB foi introduzida no JBoss EAP 6 para invocações remotas; no entanto, se você preferir não reescrever o código do aplicativo para usar a nova API, poderá modificar o código existente para usar o <code>ejb:BEAN_REFERENCE</code> para acesso remoto aos EJBs. Veja a seção a seguir no JBoss EAP 6 <i>Guia de Migração</i> Para maiores informações.</p> <ul style="list-style-type: none"> • Mudanças de Invocação Remota <p>No JBoss EAP 7, o conector padrão e a porta de conexão remota padrão foram alterados. Para mais informações, consulte as seguintes seções deste guia.</p> <ul style="list-style-type: none"> • Atualizar o conector e porta de URL remoto • Atualizar clientes externos • Migrar o código do cliente EJB
Seam 2.x	<p>Embora o suporte oficial aos aplicativos Seam 2.2 tenha sido descartado no JBoss EAP 6, ainda era possível configurar as dependências do JSF 1.2 e do Hibernate 3 para permitir que os aplicativos do Seam 2.2 fossem executados naquela versão. O JBoss EAP 7, que agora inclui o JSF 2.2 e o Hibernate 5, não suporta o Seam 2.2 ou o Seam 2.3 devido ao fim da vida útil do Red Hat JBoss Web Framework Kit. É recomendável que você reescreva seus componentes do Seam usando os beans Weld CDI.</p>
Segurança	<p>As atualizações de segurança no JBoss EAP 6 incluíram mudanças nos nomes de domínio de segurança e mudanças em como configurar segurança para autenticação básica. A configuração da região de segurança do LDAP foi movida para o arquivo de configuração do servidor. Veja as seções a seguir no JBoss EAP 6 <i>Guia de Migração</i> Para maiores informações.</p> <ul style="list-style-type: none"> • Mudanças de Segurança • Alterações do Realm de Segurança LDAP <p>As atualizações que impactam a segurança no JBoss EAP 7 incluem alterações na configuração do servidor e alterações no aplicativo. Informações podem ser encontradas nas seguintes seções deste guia.</p> <ul style="list-style-type: none"> • Alterações na configuração do servidor de segurança • Alterações no aplicativo de segurança

JBoss EAP 5 Recurso ou componente	Resumo de alterações e Onde encontrar informações de migração
Alterações de aplicativos de segurança [Esta é uma tradução automática]	<p>O Spring 4.2.x é a primeira versão estável do Spring suportada pelo JBoss EAP 7. Para obter informações sobre os serviços da Web do Apache CXF Spring e as alterações de integração do Spring RESTEasy, consulte as seções a seguir neste guia.</p> <ul style="list-style-type: none"> • Alterações dos Serviços da Web do Apache CXF Spring • Mudanças na Integração do Spring
Transações	<p>A configuração de transação consolidada do JBoss EAP 6 foi movida para o arquivo de configuração do servidor. Outras atualizações incluíram alterações nas configurações do identificador de nó JTA e como habilitar o JTS. Para detalhes, veja a seção seguinte no JBoss EAP 6 <i>Guia de Migração</i>.</p> <ul style="list-style-type: none"> • JTS e JTA Changes <p>Alguns atributos de configuração do Gerenciador de Transações que estavam disponíveis no transactions O subsistema no JBoss EAP 6 foi alterado no JBoss EAP 7. Para obter mais informações, consulte a seção a seguir deste guia.</p> <ul style="list-style-type: none"> • Alterações do subsistema de transações
Válvulas	<p>O Undertow substituiu o JBoss Web no JBoss EAP 7 e as válvulas não são mais suportadas. Veja as seções a seguir neste guia.</p> <ul style="list-style-type: none"> • Migrar Válvulas Globais • Migrar válvulas de aplicativos personalizados • Migrar válvulas do autenticador
Serviços da Web	<p>O JBoss EAP 6 incluiu o JBossWS 4. Para obter informações sobre as mudanças requeridas por essa atualização de versão, veja a seção a seguir no JBoss EAP 6 <i>Guia de Migração</i>.</p> <ul style="list-style-type: none"> • Alterações nos Serviços da Web <p>O JBoss EAP 7 introduziu o JBossWS 5. Veja a seção a seguir neste guia para atualizações necessárias.</p> <ul style="list-style-type: none"> • Alterações nos Aplicativos de Serviços da Web

APÊNDICE A. MATERIAL DE REFERÊNCIA [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

A.1. AVISOS DE OPERAÇÃO DE MIGRAÇÃO DE SUBSISTEMA JACORB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

o **migrate** operação não é capaz de processar todos os recursos e atributos. A tabela a seguir lista alguns dos avisos que você pode ver ao executar o **migrate** ou **describe-migration** operação para o **jacorb** subsistema.



NOTA

Se você vir as entradas "Não foi possível migrar" ou "Não pode migrar" na saída do **migrate** operação, isso indica que a migração da configuração do servidor foi concluída com êxito, mas não conseguiu migrar automaticamente todos os elementos e atributos. Você deve seguir as sugestões fornecidas pelos "avisos de migração" para modificar essas configurações.

Mensagens de aviso	O que significa / Como resolver
o iiop migração pode ser executada quando o servidor está em admin-only modo	o migrate operação requer iniciar o servidor em admin-only modo, o que é feito adicionando --start-mode=admin-only para o comando start do servidor: <pre>\$ EAP_HOME/bin/standalone.sh --start-mode=admin-only</pre>
Propriedades <i>X</i> não pode ser emulado usando OpenJDK ORB e não são suportados	A configuração da propriedade especificada não é suportada e não está incluída no novo iiop-openjdk configuração do subsistema. O comportamento exibido por esta propriedade na versão anterior do JBoss EAP não é migrado e o administrador deve verificar se o novo iiop-openjdk O subsistema no JBoss EAP 7 é capaz de operar corretamente sem esse comportamento. Propriedades não suportadas incluem: cache-poa-names , cache-typecodes , chunk-custom-rmi-valuetypes , client-timeout , comet , indirection-encoding-disable , iona , lax-boolean-encoding , max-managed-buf-size , max-server-connections , max-threads , outbuf-cache-timeout , outbuf-size , queue-max , queue-min , poa-monitoring , print-version , retries , retry-interval , queue-wait , server-timeout , strict-check-on-tc-creation , use-bom , use-imr .

Mensagens de aviso	O que significa / Como resolver
As propriedades <i>X</i> use expressões. As propriedades de configuração usadas para resolver essas expressões devem ser transformadas manualmente para o novo iiop-openjdk formato do subsistema	<p>Propriedades que usam expressões devem ser configuradas manualmente pelo administrador.</p> <p>Por exemplo, o jacorb subsistema no JBoss EAP 6 definiu um giop-minor-version propriedade. o iiop-openjdk subsistema no JBoss EAP 7 define um giop-version propriedade. Suponha que o jacorb atributo de versão secundária do subsistema está definido como \${iiop-giop-minor-version} e a propriedade do sistema é configurada no standalone.conf arquivo como -Diiop-giop-minor-version=1. Depois de migrate operação, o administrador deve alterar o valor da propriedade do sistema para 1.1 para garantir que o novo subsistema esteja configurado corretamente.</p>
Não é possível migrar: o novo iiop-openjdk subsistema já está definido	A mensagem contém a explicação.

A.2. AVISOS PARA A OPERAÇÃO DE MIGRAÇÃO DE SUBSISTEMA DE MENSAGEM [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

o **migrate** operação não é capaz de processar todos os recursos e atributos. A tabela a seguir lista alguns dos avisos que você pode ver ao executar o **migrate** ou **describe-migration** operação para o **messaging** subsistema.



NOTA

Se você vir as entradas "Não foi possível migrar" ou "Não pode migrar" na saída do **migrate** operação, isso indica que a migração da configuração do servidor foi concluída com êxito, mas não conseguiu migrar automaticamente todos os elementos e atributos. Você deve seguir as sugestões fornecidas pelos "avisos de migração" para modificar essas configurações.

Mensagens de aviso	O que significa / Como resolver
o migrate operação não pode ser executada: o servidor deve estar em admin-only modo	<p>o migrate operação requer iniciar o servidor em admin-only modo, o que é feito adicionando --start-mode=admin-only para o comando start do servidor:</p> <pre>\$ EAP_HOME/bin/standalone.sh --start-mode=admin-only</pre>

Mensagens de aviso	O que significa / Como resolver
Não é possível migrar o atributo local-bind-address do recurso X. Use em vez disso socket-binding atributo para configurar este broadcast-group .	A mensagem contém explicação e indica como fazer para corrigir o problema.
Não é possível migrar o atributo local-bind-port do recurso X. Use em vez disso socket-binding atributo para configurar este broadcast-group .	A mensagem contém explicação e indica como fazer para corrigir o problema.
Não é possível migrar o atributo group-address do recurso X. Use em vez disso socket-binding atributo para configurar este broadcast-group .	A mensagem contém explicação e indica como fazer para corrigir o problema.
Não é possível migrar o atributo group-port do recurso X. Use em vez disso socket-binding atributo para configurar este broadcast-group .	o broadcast-group recurso não aceita mais local-bind-address , local-bind-port , group-address , ou group-port atributos. Só aceita um socket-binding atributo. O aviso é a notificação de que o recurso X tem um atributo não suportado. Você deve definir manualmente socket-binding atributo no recurso e garantir que corresponde a um definido socket-binding recurso.
Classes que fornecem o X são descartados durante a migração. Para usá-los no novo messaging-activemq subsistema, você terá que estender a base de Artemis Interceptor .	O suporte a interceptores de mensagens é significativamente diferente no JBoss EAP 7. Quaisquer interceptores configurados na versão anterior do subsistema são descartados durante a migração. Veja Migrar Interceptores de Mensagens Para maiores informações.
Não é possível migrar a configuração HA de X. Está shared-store e backup atributos contém expressões e não é possível determinar inequivocamente como criar o correspondente ha-policy para o servidor do messaging-activemq.	Isso significa que o hornetq-server X'S shared-store ou backup os atributos continham uma expressão, como \$ {xxx}, e a operação de migração não conseguiu resolvê-lo para uma expressão concreta. O valor é descartado e o ha-policy para o messaging-activemq deve ser atualizado manualmente.
Não é possível migrar o atributo local-bind-address do recurso X. Use em vez disso socket-binding atributo para configurar este discovery-group .	A mensagem contém explicação e indica como fazer para corrigir o problema.
Não é possível migrar o atributo local-bind-port do recurso X. Use em vez disso socket-binding atributo para configurar este discovery-group .	A mensagem contém explicação e indica como fazer para corrigir o problema.

Mensagens de aviso	O que significa / Como resolver
<p>Não é possível migrar o atributo group-address do recurso <i>X</i>. Use em vez disso socket-binding atributo para configurar este discovery-group.</p>	<p>A mensagem contém explicação e indica como fazer para corrigir o problema.</p>
<p>Não é possível migrar o atributo group-port do recurso <i>X</i>. Use em vez disso socket-binding atributo para configurar este discovery-group.</p>	<p>o discovery-group os recursos não aceitam mais local-bind-address, local-bind-port, group-address, ou group-port atributos. Só aceita um socket-binding. O aviso é a notificação de que o recurso <i>X</i> tem um atributo não suportado. Você deve definir manualmente socket-binding atributo no recurso e garante que corresponde a um definido socket-binding recurso.</p>
<p>Não é possível criar um legacy-connection-factory baseado em connection-factory <i>X</i>. Ele usa um HornetQ in-vm conector que não é compatível com Artemis in-vm conector</p>	<p>O remoto legado do HornetQ connection-factory recursos são migrados para legacy-connection-factory recursos para permitir que os clientes do JBoss EAP 6 se conectem ao JBoss EAP 7. No entanto, legacy-connection-factory recursos são criados apenas quando o connection-factory está usando conectores remotos. Qualquer connection-factory usando in-vm não é migrado porque in-vm clientes são baseados no JBoss EAP 7, não no JBoss EAP 6. Este aviso é a notificação de que o in-vm connection-factory não foi migrado.</p>
<p>Não é possível migrar o atributo <i>X</i> do recurso <i>Y</i>. O atributo usa uma expressão que pode ser resolvida de forma diferente, dependendo das propriedades do sistema. Após a migração, esse atributo deve ser adicionado novamente com um valor real em vez da expressão.</p>	<p>Este aviso aparece quando a migração não pode resolver o atributo <i>X</i> a um valor concreto durante o processo de migração. O valor é descartado e o atributo deve ser migrado manualmente. Isso acontece nos seguintes casos:</p> <ul style="list-style-type: none"> • cluster-connection forward-when-no-consumers: Este atributo booleano foi substituído pelo message-load-balancing-type atributo, que é um enum com um valor de OFF, STRICT, ou ON_DEMAND. • broadcast-group e discovery-group's jgroups-stack e jgroups-channel atributos Eles referenciam outros recursos e JBoss EAP 7 não aceita mais estas expressões.

Mensagens de aviso	O que significa / Como resolver
Não é possível migrar o atributo <i>X</i> do recurso <i>Y</i> . Este atributo não é suportado pelo novo messaging-activemq subsistema.	<p>Alguns atributos não são mais suportados no novo messaging-activemq subsistema e são simplesmente descartados:</p> <ul style="list-style-type: none"> Exemplo: @DateBridge e @CalendarBridge Anotação [Esta é uma tradução automática] Exemplo: @DateBridge e @CalendarBridge Anotação [Esta é uma tradução automática] Exemplo: @DateBridge e @CalendarBridge Anotação [Esta é uma tradução automática] remote-connector's use-nio atributo remote-acceptor's use-nio atributo
Não é possível migrar o atributo failback-delay do recurso <i>X</i> . O Artemis detecta o failback deterministicamente e não precisa mais especificar um atraso para que o failback ocorra.	A mensagem contém a explicação.

Substituir os atributos preteridos **broadcast-group** ou **discovery-group**

Se você é aconselhado a substituir o obsoleto **broadcast-group** ou **discovery-group** atributos com o **socket-binding** atributo, você pode adicionar o novo atributo usando a CLI de gerenciamento.

Este exemplo assume que você está migrando um servidor independente que contém o seguinte **discovery-group** configuração no **messaging** subsistema.

```
<discovery-groups>
  <discovery-group name="my-discovery-group">
    <group-address>224.0.1.105</group-address>
    <group-port>56789</group-port>
  </discovery-group>
</discovery-groups>
```

Quando você executa o **migrate** operação para o **messaging** subsistema, você vê a seguinte saída e avisos:

```
/subsystem=messaging:migrate
{
  "outcome" => "success",
  "result" => {"migration-warnings" => [
    "WFLYMSG0084: Can not migrate attribute group-address from
resource [
  (\\"subsystem\\" => \\"messaging-activemq\\"),
  (\\"server\\" => \\"default\\"),
  (\\"discovery-group\\" => \\"my-discovery-group\\")
]. Use instead the socket-binding attribute to configure this discovery-
```

```
group.",
    "WFLYMSG0084: Can not migrate attribute group-port from resource [
        (\\"subsystem\\" => \\"messaging-activemq\\"),
        (\\"server\\" => \\"default\\"),
        (\\"discovery-group\\" => \\"my-discovery-group\\")
    ]. Use instead the socket-binding attribute to configure this discovery-
    group."
    ]}
}
```

o **migrate** operação cria um **discovery-group** chamado "my-discovery-group" no novo **messaging-activemq** subsistema que agora está configurado como o seguinte.

```
<discovery-group name="my-discovery-group"/>
```

Agora você deve usar o seguinte comando CLI de gerenciamento para criar um **socket-binding** elemento no arquivo de configuração do servidor chamado "my-discovery-group-socket-binding".

```
/socket-binding-group=standard-sockets/socket-binding=my-discovery-group-
socket-binding:add(multicast-address=224.0.1.105, multicast-port=56789)
```

Em seguida, adicione o recém-criado **socket-binding** ao **discovery-group** chamado "my-discovery-group" no **messaging-activemq** subsistema no arquivo de configuração do servidor usando o seguinte comando da CLI de gerenciamento.

```
/subsystem=messaging-activemq/server=default/discovery-group=my-discovery-
group:write-attribute(name=socket-binding,value=my-discovery-group-socket-
binding)
```

Estes comandos criam o seguinte XML no arquivo de configuração do servidor.

```
<subsystem xmlns="urn:jboss:domain:messaging-activemq:2.0">
    <server name="default">
        ...
        <discovery-group name="my-discovery-group" socket-binding="my-
discovery-group-socket-binding"/>
        ...
    </server>
</subsystem>
...
<socket-binding-group name="standard-sockets" default-interface="public"
port-offset="\${jboss.socket.binding.port-offset:0}">
    ...
    <socket-binding name="my-discovery-group-socket-binding" multicast-
address="224.0.1.105" multicast-port="56789"/>
    ...
</socket-binding-group>
```

A.3. AVISOS PARA OPERAÇÃO DE MIGRAÇÃO DE SUBSISTEMA WEB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

o **migrate** operação não é capaz de processar todos os recursos e atributos. A tabela a seguir lista alguns dos avisos que você pode ver ao executar o **migrate** ou **describe-migration** operação para o **web** subsistema.



NOTA

Se você vir as entradas "Não foi possível migrar" ou "Não pode migrar" na saída do **migrate** operação, isso indica que a migração da configuração do servidor foi concluída com êxito, mas não conseguiu migrar automaticamente todos os elementos e atributos. Você deve seguir as sugestões fornecidas pelos "avisos de migração" para modificar essas configurações.

Mensagens de aviso	O que significa / Como resolver
Operação de migração permitida somente em modo admin	o migrate operação requer iniciar o servidor em admin-only modo, o que é feito adicionando o parâmetro --admin-only para o comando start do servidor: \$ EAP_HOME/bin/standalone.sh --admin-only
Não foi possível migrar recurso X	O comportamento exibido por este recurso na versão anterior do JBoss EAP não foi migrado. O administrador deve verificar se o novo undertow O subsistema no JBoss EAP 7 é capaz de operar corretamente sem esse comportamento ou se o comportamento deve ser migrado manualmente.
Não foi possível migrar o atributo X do recurso Y.	O comportamento exibido por este atributo de recurso na versão anterior do JBoss EAP não foi migrado. O administrador deve verificar se o novo undertow O subsistema no JBoss EAP 7 é capaz de operar corretamente sem esse comportamento ou se o comportamento deve ser migrado manualmente. Veja Avisos de Atributos da Operação de Migração do Subsistema da Web para a lista de atributos que não são migrados.
Não foi possível migrar conector SSL pois nenhuma configuração SSL está definida.	A mensagem contém a explicação.
Não foi possível migrar verify-client atributo X ao equivalente Undertow	A mensagem contém a explicação.
Não foi possível migrar verify-client expressão X	A mensagem contém a explicação.

Mensagens de aviso	O que significa / Como resolver
<p>Não foi possível migrar a válvula <i>X</i></p>	<p>O comportamento exibido por esta válvula na versão anterior do JBoss EAP não foi migrado. O administrador deve verificar se o novo undertow O subsistema no JBoss EAP 7 é capaz de operar corretamente sem esse comportamento ou se o comportamento deve ser migrado manualmente.</p> <p>Este aviso pode ocorrer para as seguintes válvulas:</p> <ul style="list-style-type: none"> • org.apache.catalina.valves.RemoteAddrValve Deve ter pelo menos um valor permitido ou negado. • org.apache.catalina.valves.RemoteHostValve Deve ter pelo menos um valor permitido ou negado. • org.apache.catalina.authenticator.BasicAuthenticator • org.apache.catalina.authenticator.DigestAuthenticator • org.apache.catalina.authenticator.FormAuthenticator • org.apache.catalina.authenticator.SSLAuthenticator • org.apache.catalina.authenticator.SpnegoAuthenticator • Válvulas personalizadas

Mensagens de aviso	O que significa / Como resolver
<p>Não foi possível migrar o atributo <i>X</i> da válvula <i>Y</i></p>	<p>O comportamento exibido por este atributo de válvula na versão anterior do JBoss EAP não foi migrado. O administrador deve verificar se o novo undertow O subsistema no JBoss EAP 7 é capaz de operar corretamente sem esse comportamento ou se o comportamento deve ser migrado manualmente. Este aviso pode ocorrer para os seguintes atributos da válvula:</p> <ul style="list-style-type: none"> • org.apache.catalina.valves.AccessLogValve <ul style="list-style-type: none"> ◦ resolveHosts ◦ fileDateFormat ◦ renameOnRotate ◦ encoding ◦ locale ◦ requestAttributesEnabled ◦ buffered • org.apache.catalina.valves.ExtendedAccessLogValve <ul style="list-style-type: none"> ◦ resolveHosts ◦ fileDateFormat ◦ renameOnRotate ◦ encoding ◦ locale ◦ requestAttributesEnabled ◦ buffered • org.apache.catalina.valves.RemoteIpValve <ul style="list-style-type: none"> ◦ httpServerPort ◦ httpsServerPort ◦ remoteIpHeader Se estiver definido porém não configurado para "x-forwarded-for" ◦ protocolHeader Se estiver definido porém não configurado para "x-forwarded-proto"

Mensagens de aviso	O que significa / Como resolver
Avisos para operação de migração de subsistema web [Esta é uma tradução automática]	
o migrate a operação não é capaz de processar todos os atributos do JBoss Web. Consulte as tabelas de referência a seguir para obter informações sobre como migrar manualmente os atributos não processados.	

Atributos do Conector SSL da Web

Os seguintes atributos foram usados no JBoss EAP 6 para configurar o conector SSL. As bibliotecas nativas do OpenSSL não são suportadas no JBoss EAP 7, portanto não há configurações equivalentes.

Atributos para remover [Esta é uma tradução automática]	Descrição	Undertow Equivalent
ca-revocation-url	O arquivo ou URL que contém a lista de revogação.	Nenhum equivalente em Undertow.
certificate-file	Ao usar a criptografia OpenSSL, o caminho para o arquivo que contém o certificado do servidor.	Nenhum equivalente em Undertow.
ssl-protocol	A cadeia do protocolo SSL.	Nenhum equivalente em Undertow.
verificar profundidade	O número máximo de emissores de certificados intermediários verificados antes de decidir que os clientes não possuem um certificado válido.	Nenhum equivalente em Undertow.

Atributos de recursos estáticos da Web

Os seguintes **static-resources** atributos de elemento foram usados para descrever como os recursos estáticos eram manipulados pelo **DefaultServlet** ou pelo **WebdavServlet**. Não há equivalentes para esses atributos porque o WebDAV não é suportado pelo Undertow. Para mais informações, veja <https://issues.jboss.org/browse/JBEAP-1036>.

Atributos para remover [Esta é uma tradução automática]	Descrição	Undertow Equivalent
Desativado	Ative o mapeamento de Servlet padrão.	Nenhuma configuração equivalente no Undertow.
codificação de arquivos	Codificação de arquivo a ser usada ao ler arquivos estáticos.	Nenhuma configuração equivalente no Undertow.
max-depth	Recursão máxima para PROPFIND .	Esta é uma configuração do WebDAV e o WebDAV não é suportado pelo Undertow.

Atributos para remover [Esta é uma tradução automática]	Descrição	Undertow Equivalent
somente leitura	Permitir gravação de métodos HTTP (PUT, DELETE).	Esta é uma configuração do WebDAV e o WebDAV não é suportado pelo Undertow.
segredo	Segredo para operações de bloqueio do WebDAV.	Esta é uma configuração do WebDAV e o WebDAV não é suportado pelo Undertow.
sendfile	Ative o sendfile, se possível, para arquivos maiores que o tamanho do byte especificado.	Isso é definido como um valor padrão sensato em Undertow e não é configurável.
webdav	Ativar a funcionalidade do WebDAV.	O WebDAV não é suportado pelo Undertow.

Atributos de Recurso SSO da Web

O SSO é tratado de maneira diferente da versão anterior e não há configurações de atributos equivalentes no JBoss EAP 7.

Atributo da Web do JBoss	Descrição	Undertow Equivalent
cache-container	Nome do contêiner de cache a ser usado para o SSO em cluster.	Essa configuração não é mais necessária no Undertow. Isso funciona por padrão em um ambiente de cluster distribuído.
cache-name	Nome do cache a ser usado para o SSO em cluster.	Essa configuração não é mais necessária no Undertow. Isso funciona por padrão em um ambiente de cluster distribuído.
Migrar válvulas do autenticador [Esta é uma tradução automática]	Se cada solicitação deve causar uma nova autenticação.	Não há configuração equivalente em Undertow, que se comporta de maneira semelhante à reauthenticate=true configuração no JBoss EAP 6. Enquanto reauthenticate=false poderia melhorar o desempenho, também poderia criar problemas de segurança.

Mapeamento dos atributos de mensagens [Esta é uma tradução automática]

Atributo da Web do JBoss	Descrição	Undertow Equivalent
resolve-hosts	Se deve ativar os hosts de resolução para o log de acesso.	Use a configuração no conector para obter o mesmo comportamento.

Atributos do Conector da Web

Atributo da Web do JBoss	Descrição	Undertow Equivalent
executor	O nome do executor que deve ser usado para processar os encadeamentos desse conector.	Exemplo: propriedades de segurança definidas no security Subsistema [Esta é uma tradução automática] Vejo Migrar a configuração do subsistema de threads Para maiores informações.
ligação de proxy	A ligação do soquete para definir o host e a porta usados ao enviar um redirecionamento.	Não há equivalente direto. Vejo Atributos do https-listener no JBoss EAP <i>Guia de configuração</i> para opções de configuração disponíveis.
porta de redirecionamento	A porta para redirecionamento para um conector seguro.	Este atributo foi preterido no JBoss EAP 6 e substituído por redirect-binding . Undertow fornece o redirect-socket atributo no http-listener elemento, que é um substituto para redirect-binding . Vejo Atributos do https-listener no JBoss EAP <i>Guia de configuração</i> Para maiores informações.

A.4. MIGRAR REFERÊNCIA DE PROPRIEDADES DO SISTEMA JBOSS WEB [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Esta referência descreve como mapear as propriedades do sistema anteriormente usadas para a configuração do JBoss Web para a configuração equivalente do Undertow no JBoss EAP 7.

- [Mapear propriedades do sistema Container e conectores servlet](#)
- [Mapear as propriedades do sistema EL](#)
- [Mapear propriedades do sistema JSP](#)
- [Mapear propriedades do sistema de segurança](#)

Tabela A.1. Mapear propriedades do sistema Container e conectores servlet [Esta é uma tradução automática]

Propriedade de Sistema do JBoss EAP 6	Descrição
	Equivalente no JBoss EAP 7
jvmRoute	<p>Fornece um valor padrão para o jvmRoute atributo. Ele não substitui o valor gerado automaticamente ao usar o standalone-ha.xml arquivo de configuração.</p> <p>Suporta reload.</p> <p>Operação de migração da CLI de gerenciamento [Esta é uma tradução automática]</p> <pre>/subsystem=undertow:write-attribute(name=instance-id,value=VALUE)</pre>
org.apache.tomcat.util.buf.StringCache.byte.enabled	<p>E se true, o cache String está habilitado para ByteChunk. Se o valor não for especificado, o valor padrão de false é usado.</p> <p>Substituir as configurações Netty Servlet [Esta é uma tradução automática]</p>
org.apache.tomcat.util.buf.StringCache.char.enabled	<p>E se true, o cache String está habilitado para CharChunk. Se o valor não for especificado, o valor padrão de false é usado.</p> <p>Substituir as configurações Netty Servlet [Esta é uma tradução automática]</p>
org.apache.tomcat.util.buf.StringCache.cacheSize	<p>O tamanho do cache da cadeia. Se o valor não for especificado, o valor padrão de 5000 é usado.</p> <p>Substituir as configurações Netty Servlet [Esta é uma tradução automática]</p>
org.apache.tomcat.util.buf.StringCache.maxStringSize	<p>O comprimento máximo de String que será armazenado em cache. Se o valor não for especificado, o valor padrão de 128 é usado.</p> <p>Substituir as configurações Netty Servlet [Esta é uma tradução automática]</p>
org.apache.tomcat.util.http.FastDateFormat.CACHE_SIZE	<p>O tamanho do cache para usar valor de data analisado e formatado. Se o valor não for especificado, o valor padrão de 1000 é usado.</p> <p>Substituir as configurações Netty Servlet [Esta é uma tradução automática]</p>

org.apache.catalina.core.StandardService.DELAY_CONNECTOR_STARTUP	E se true , a inicialização do conector não é feita automaticamente. É útil no modo incorporado.
	Substituir as configurações Netty Servlet [Esta é uma tradução automática]
org.apache.catalina.connector.Request.SSESSION_ID_CHECK	E se true , o contêiner Servlet verifica se existe uma sessão em um contexto com o ID de sessão especificado antes de criar uma sessão com esse ID.
	Substituir as configurações Netty Servlet [Esta é uma tradução automática]
org.apache.coyote.USE_CUSTOM_STAT US_MSG_IN_HEADER	E se true , mensagens de status HTTP personalizadas são usadas em cabeçalhos HTTP. Os usuários devem garantir que qualquer mensagem seja ISO-8859-1 codificado, especialmente se a entrada fornecida pelo usuário estiver incluída na mensagem, para evitar uma possível vulnerabilidade do XSS. Se o valor não for especificado, o valor padrão de false é usado.
	Deve ser ativado programaticamente implementando um <code>io.undertow.servlet.ServletExtension</code> . Em seguida, use a extensão para chamar <code>setSendCustomReasonPhraseOnError(true)</code> no <code>io.undertow.servlet.api.DeploymentInfo</code> instância de estrutura.
org.apache.tomcat.util.http.Parameters.MAX_COUNT	O número máximo de parâmetros que podem ser analisados em um corpo de postagem. Se excedido, a análise falhará usando um IllegalStateException . o valor padrão é 512 parâmetros.
	<p>Operação de migração da CLI de gerenciamento [Esta é uma tradução automática]</p> <pre> /subsystem=undertow/server=default-server/http-listener=default:write-attribute(name=max-parameters,value=VALUE) /subsystem=undertow/server=default-server/https-listener=default:write-attribute(name=max-parameters,value=VALUE) /subsystem=undertow/server=default-server/ajp-listener=default:write-attribute(name=max-parameters,value=VALUE) </pre>

<p>org.apache.tomcat.util.http.MimeHeaders. MAX_COUNT</p>	<p>O número máximo de cabeçalhos que podem ser enviados na solicitação HTTP. Se excedido, a análise falhará usando um IllegalStateException. o valor padrão é 128 cabeçalhos.</p> <p>Operação de migração da CLI de gerenciamento [Esta é uma tradução automática]</p> <pre>/subsystem=undertow/server=default-server/http-listener=default:write-attribute(name=max-headers,value=VALUE) /subsystem=undertow/server=default-server/https-listener=default:write-attribute(name=max-headers,value=VALUE) /subsystem=undertow/server=default-server/ajp-listener=default:write-attribute(name=max-headers,value=VALUE)</pre>
<p>org.apache.tomcat.util.net.MAX_THREADS</p>	<p>O número máximo de encadeamentos que um conector vai usar para processar solicitações. o valor padrão é 32 x 512. (512 x Runtime.getRuntime().availableProcessors() para o JIO conector)</p> <p>Operação de migração da CLI de gerenciamento [Esta é uma tradução automática]</p> <pre>/subsystem=io/worker=default:write-attribute(name=task-max-threads,value=VALUE)</pre>
<p>org.apache.coyote.http11.Http11Protocol. MAX_HEADER_SIZE</p>	<p>O tamanho máximo dos cabeçalhos HTTP, em bytes. Se excedido, a análise falhará usando um ArrayOutOfBoundsException. o valor padrão é 8192 bytes.</p> <p>Operação de migração da CLI de gerenciamento [Esta é uma tradução automática]</p> <pre>/subsystem=undertow/server=default-server/http-listener=default:write-attribute(name=max-header-size,value=VALUE) /subsystem=undertow/server=default-server/https-listener=default:write-attribute(name=max-header-size,value=VALUE) /subsystem=undertow/server=default-server/ajp-listener=default:write-attribute(name=max-header-size,value=VALUE)</pre>

org.apache.coyote.http11.Http11Protocol. COMPRESSION	<p>Permite o uso de compactação simples com o conector HTTP. o valor padrão é off, e a compactação pode ser ativada usando o valor on para habilitá-lo condicionalmente, ou forçar sempre habilitá-lo.</p> <p>Configure um filtro usando a CLI de gerenciamento:</p> <pre># Create a filter /subsystem=undertow/configuration=filter/ gzip=gzipfilter:add() /subsystem=undertow/server=default- server/host=default-host/filter- ref=gzipfilter:add()</pre>
org.apache.coyote.http11.Http11Protocol. COMPRESSION_RESTRICTED_UA	<p>Os agentes do usuário regexexam que não receberão conteúdo compactado. O valor padrão está vazio.</p> <p>Configure um predicado em um filtro usando a CLI de gerenciamento:</p> <pre># Use a predicate in a filter /subsystem=undertow/configuration=filter/ gzip=gzipfilter:add() /subsystem=undertow/server=default- server/host=default-host/filter- ref=gzipfilter:add(predicate="regex[patte rn='AppleWebKit', value=%{i, User-Agent}]")</pre>
org.apache.coyote.http11.Http11Protocol. COMPRESSION_MIME_TYPES	<p>Prefixos de tipo de conteúdo de conteúdo compactável. o valor padrão é text/html, text/xml, text/plain.</p> <p>Configure um predicado em um filtro usando a CLI de gerenciamento:</p> <pre># Use a predicate in a filter /subsystem=undertow/configuration=filter/ gzip=gzipfilter:add() /subsystem=undertow/server=default- server/host=default-host/filter- ref=gzipfilter:add(predicate="regex[patte rn='text/html', value=%{o, Content-Type}]")</pre>
org.apache.coyote.http11.Http11Protocol. COMPRESSION_MIN_SIZE	<p>Tamanho mínimo do conteúdo que será compactado. o valor padrão é 2048 bytes.</p>

	<p>Configure um predicado em um filtro usando a CLI de gerenciamento:</p> <pre># Use a predicate in a filter /subsystem=undertow/configuration=filter/ gzip=gzipfilter:add() /subsystem=undertow/server=default- server/host=default-host/filter- ref=gzipfilter:add(predicate="max- content-size[value=MIN_SIZE]")</pre>
org.apache.coyote.http11.DEFAULT_CONNECTION_TIMEOUT	<p>Tempo limite do soquete padrão. o valor padrão é 60000 Senhora.</p> <p>Operação de migração da CLI de gerenciamento [Esta é uma tradução automática]</p> <pre>/subsystem=undertow/server=default- server/http-listener=default:write- attribute(name=no-request- timeout,value=VALUE) /subsystem=undertow/server=default- server/https-listener=default:write- attribute(name=no-request- timeout,value=VALUE) /subsystem=undertow/server=default- server/ajp-listener=default:write- attribute(name=no-request- timeout,value=VALUE)</pre>
org.jboss.as.web.deployment.DELETE_WORK_DIR_ONCONTEXTDESTROY	<p>Use esta propriedade para remover .java e .class arquivos para garantir que as fontes JSP sejam recompiladas. o valor padrão é false. Tempo limite de soquete padrão para keep-alive. o valor padrão é -1 ms, o que significa que usará o tempo limite do soquete padrão.</p> <p>Substituir as configurações Netty Servlet [Esta é uma tradução automática]</p>
org.apache.tomcat.util.buf.StringCache.trimThreshold	<p>Especifica o número de vezes toString() deve ser chamado antes de ativar o cache. o valor padrão é 100000.</p> <p>Substituir as configurações Netty Servlet [Esta é uma tradução automática]</p>

Tabela A.2. Mapear as propriedades do sistema EL [Esta é uma tradução automática]

Propriedade de Sistema do JBoss EAP 6	Descrição
	Equivalente no JBoss EAP 7

org.apache.el.parser.COERCE_TO_ZERO	E se true , quando forçar expressões a números, strings vazias (""), e null serão forçadas a zero, conforme exigido pela especificação. Se um valor não for especificado, o valor padrão de true é usado.
	A propriedade do sistema ainda é válida e processada pelo EL

Tabela A.3. Mapear propriedades do sistema JSP [Esta é uma tradução automática]

Propriedade de Sistema do JBoss EAP 6	Descrição
	Equivalente no JBoss EAP 7
org.apache.jasper.compiler.ExpressionFactory	O nome da variável a ser usada para a expressão expression language factory. Se o valor não for especificado, o valor padrão de _el_expressionfactory é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.compiler.InstanceManager	O nome da variável a ser usada para a fábrica do gerenciador de instâncias. Se o valor não for especificado, o valor padrão de _jsp_instancemanager é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.compiler.Parser.STRICT_QUOTE_ESCAPING	E se false , os requisitos para as citações de escape nos atributos JSP são relaxados, de forma que uma citação requerida ausente não cause um erro. Se o valor não for especificado, o padrão de especificação compatível true é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.Constants.DEFAULT_TAG_BUFFER_SIZE	Qualquer buffer de tag que se expande além org.apache.jasper.Constants.DEFAULT_TAG_BUFFER_SIZE é destruído e um novo buffer é criado do tamanho padrão. Se o valor não for especificado, o valor padrão de 512 é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.runtime.JspFactoryImpl.USE_POOL	E se true , um pool de PageContext ThreadLocal é usado. Se o valor não for especificado, o valor padrão de true é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.runtime.JspFactoryImpl.POOL_SIZE	O tamanho do PageContext de ThreadLocal. Se o valor não for especificado, o valor padrão de 8 é usado.

	A propriedade do sistema não foi alterada
org.apache.jasper.Constants.JSP_SERVLET_BASE	A classe base dos Servlets gerada a partir das JSPs. Se o valor não for especificado, o valor padrão de org.apache.jasper.runtime.HttpJspBase é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.Constants.SERVICE_METHOD_NAME	O nome do método de serviço chamado pela classe base. Se o valor não for especificado, o valor padrão de _jspService é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.Constants.SERVLET_CLASSPATH	O nome do atributo ServletContext que fornece o caminho de classe para o JSP. Se o valor não for especificado, o valor padrão de org.apache.catalina.jsp_classpath é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.Constants.JSP_FILE	O nome do atributo de solicitação para <jsp-file> elemento de uma definição de servlet. Se presente em uma solicitação, isso substitui o valor retornado por request.getServletPath() para selecionar a página JSP a ser executada. Se o valor não for especificado, o valor padrão de org.apache.catalina.jsp_file é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.Constants.PRECOMPILE	O nome do parâmetro de consulta que faz com que o mecanismo JSP apenas pré-gerencie o servlet, mas não o invoque. Se o valor não for especificado, o valor padrão de org.apache.catalina.jsp_precompile é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.Constants.JSP_PACKAGE_NAME	O nome do pacote padrão para páginas JSP compiladas. Se o valor não for especificado, o valor padrão de org.apache.jsp é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.Constants.TAG_FILE_PACKAGE_NAME	O nome do pacote padrão para manipuladores de tag gerados a partir de arquivos de tags. Se o valor não for especificado, o valor padrão de org.apache.jsp.tag é usado.
	A propriedade do sistema não foi alterada

org.apache.jasper.Constants.TEMP_VARIABLE_NAME_PREFIX	Prefixo a ser usado para nomes de variáveis temporários gerados. Se o valor não for especificado, o valor padrão de _jspx_temp é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.Constants.USE_INSTANCE_MANAGER_FOR_TAGS	E se true , o gerenciador de instâncias é usado para obter instâncias do manipulador de tags. Se o valor não for especificado, true é usado.
	A propriedade do sistema não foi alterada
org.apache.jasper.Constants.INJECT_TAGS	E se true , as anotações especificadas nas tags serão processadas e injetadas. Isso pode ter um impacto no desempenho ao usar tags simples ou se o pool de tags estiver desativado. Se o valor não for especificado, false é usado.
	A propriedade do sistema não foi alterada

Tabela A.4. Mapear propriedades do sistema de segurança [Esta é uma tradução automática]

Propriedade de Sistema do JBoss EAP 6	Descrição
	Equivalente no JBoss EAP 7
org.apache.catalina.connector.RECYCLE_FACADES	Se isso é true ou se um gerenciador de segurança estiver em uso, um novo objeto de fachada é criado para cada solicitação. Se o valor não for especificado, o valor padrão de false é usado.
	Substituir as configurações Netty Servlet [Esta é uma tradução automática]
org.apache.catalina.connector.CoyoteAdapter.ALLOW_BACKSLASH	Se isso é true o caractere '\' é permitido como um delimitador de caminho. Se o valor não for especificado, o valor padrão de false é usado.
	Substituir as configurações Netty Servlet [Esta é uma tradução automática]
org.apache.tomcat.util.buf.UDecoder.ALLOW_ENCODED_SLASH	Se isso é true , '%2F' e '%5C' é permitido como delimitadores de caminho. Se o valor não for especificado, o valor padrão de false é usado.

	<p>Operação de migração da CLI de gerenciamento [Esta é uma tradução automática]</p> <pre> /subsystem=undertow/server=default-server/http-listener=default:write-attribute(name=allow-encoded-slash,value=VALUE) /subsystem=undertow/server=default-server/https-listener=default:write-attribute(name=allow-encoded-slash,value=VALUE) /subsystem=undertow/server=default-server/ajp-listener=default:write-attribute(name=allow-encoded-slash,value=VALUE) </pre>
org.apache.catalina.STRICT_SERVLET_COMPLIANCE	<p>Se o valor não for especificado, true é usado. Se isso é true as seguintes ações ocorrerão: qualquer solicitação ou resposta envolvida transferida para um distribuidor de aplicativos é verificada para garantir que ela tenha envolvido a solicitação ou resposta original. (SRV.8.2 / SRV.14.2.5.1) uma chamada para Response.getWriter() se nenhuma codificação de caracteres tiver sido especificada, resultará em chamadas subsequentes para Response.setCharacterEncoding() retornando ISO-8859-1 e a Content-Type cabeçalho de resposta irá incluir um charset=ISO-8859-1 componente. (SRV.15.2.22.1) cada solicitação associada a uma sessão faz com que o último horário de acesso da sessão seja atualizado, independentemente de a solicitação acessar explicitamente a sessão. (SRV.7.6)</p> <p>Compliant por padrão</p>
org.apache.catalina.core.StandardWrapperValve.SERVLET_STATS	<p>E se true ou se org.apache.catalina.STRICT_SERVLET_COMPLIANCE for true, o wrapper coletará as estatísticas da JSR-77 para servlets individuais. Se o valor não for especificado, o valor padrão de false é usado.</p> <p>Substituir as configurações Netty Servlet [Esta é uma tradução automática]</p>
org.apache.catalina.session.StandardSession.ACTIVITY_CHECK	<p>Se isso é true ou se org.apache.catalina.STRICT_SERVLET_COMPLIANCE é true O Tomcat rastreia o número de solicitações ativas para cada sessão. Ao determinar se uma sessão é válida, qualquer sessão com pelo menos uma solicitação ativa sempre será considerada válida. Se o valor não for especificado, o valor padrão de false é usado.</p>

Substituir as configurações Netty Servlet [Esta é uma tradução automática]

A.5. COMPATIBILIDADE E INTEROPERABILIDADE ENTRE OS LANÇAMENTOS [ESTA É UMA TRADUÇÃO AUTOMÁTICA]

Esta seção descreve a compatibilidade e interoperabilidade de cliente e servidor EJB e componentes de mensagem entre os lançamentos JBoss EAP 5, JBoss EAP 6, e JBoss EAP 7.

EJB remoto via IIOP

Você não deve encontrar problemas em quaisquer das seguintes configurações.

- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

EJB remoto usando JNDI

Você não deve encontrar problemas em quaisquer das seguintes configurações.

- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

JBoss EAP 6 fornecia suporte para a especificação EJB 3.1 e introduziu o uso de espaço de nomes JNDI globais padronizados, que ainda são usados em JBoss EAP 7. Devido à alteração dos espaço de nomes JNDI, as seguintes configurações não são compatíveis:

- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Conexão a partir de um cliente JBoss EAP 7 ou JBoss EAP 6 para um servidor JBoss EAP 5

Para obter mais informações sobre alterações de namespace JNDI padronizadas, consulte [Alterações no JNDI](#) no JBoss EAP 6 *Guia de Migração*.

EJB remoto utilizando @WebService

Você não deve encontrar problemas em quaisquer das seguintes configurações.

- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

Cliente autônomo de mensagem

Você não deve encontrar problemas em quaisquer das seguintes configurações.

- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

Nas seguintes configurações, se o cliente estiver usando a API HornetQ de mensagem específica ao invés da API JMS genérica, a conexão é possível. Contudo, as pesquisas JNDI devem ser endereçadas utilizando a extensão de nome JNDI herdada de JBoss EAP que é distribuída com JBoss EAP 7.

- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

O sistema de mensagens integrado ao JBoss EAP 7 não pode conectar-se ao HornetQ 2.2.x que é distribuído com JBoss EAP 5 devido a questões de incompatibilidade de protocolo. Por esta razão, as seguintes configurações não são compatíveis.

- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

Migrar dados de mensagem [Esta é uma tradução automática]

Você não deve encontrar problemas em quaisquer das seguintes configurações.

- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

Nas seguintes configurações, se o cliente estiver usando a API HornetQ de mensagem específica ao invés da API JMS genérica, a conexão é possível. Contudo, as pesquisas JNDI devem ser endereçadas utilizando a extensão de nome JNDI herdada de JBoss EAP que é distribuída com JBoss EAP 7.

- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

O sistema de mensagens integrado ao JBoss EAP 7 não pode conectar-se ao HornetQ 2.2.x que é distribuído com JBoss EAP 5 devido a questões de incompatibilidade de protocolo. Por esta razão, as seguintes configurações não são compatíveis.

- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

Pontes JMS

Você não deve encontrar problemas em quaisquer das seguintes configurações.

- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]
- Migrando do JBoss EAP 5 para o JBoss EAP 7 [Esta é uma tradução automática]

Revised on 2018-07-26 08:43:30 EDT