# Red Hat JBoss Enterprise Application Platform 7.0 Guide de migration

---

À utiliser avec Red Hat JBoss Enterprise Application Platform 7

Red Hat Customer Content Services

# Red Hat JBoss Enterprise Application Platform 7.0 Guide de migration

À utiliser avec Red Hat JBoss Enterprise Application Platform 7

## Notice légale

Copyright © 2017 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.

## Résumé

Ce guide fournit des informations sur la migration de votre application à partir de versions plus anciennes de Red Hat JBoss Enterprise Application Platform.

# Table des matières

**CHAPITRE 1. INTRODUCTION ..... 4**
1.1. RED HAT JBOSS ENTERPRISE APPLICATION PLATFORM 7 4
1.2. GUIDE DE MIGRATION 4
1.3. À PROPOS DES MIGRATIONS ET DES MISES À NIVEAU 4
1.4. L'UTILISATION DE EAP_HOME DANS CE DOCUMENT 5

**CHAPITRE 2. PRÉPARATION À LA MIGRATION ..... 6**
2.1. VUE D'ENSEMBLE DES PRÉPARATIFS 6
2.2. VOIR LES FONCTIONNALITÉS JAVA EE 7 6
2.3. VOIR LES NOUVEAUTÉS DANS JBOSS EAP 7 6
2.4. VOIR LA LISTE DES FONCTIONNALITÉS DÉPRÉCIÉES ET NON PRISES EN CHARGE. 7
2.5. VOIR LES DOCUMENTS DE MISE EN ROUTE DE JBOSS EAP. 8
2.6. ANALYSE ET PLANIFICATION DE LA MIGRATION 8
2.7. SAUVEGARDEZ VOS DONNÉES IMPORTANTES ET VÉRIFIEZ L'ÉTAT DE VOTRE SERVEUR 10
2.8. MIGRER UNE INSTALLATION RPM 11
2.9. MIGRATION DE JBOSS EAP SI VOUS EXÉCUTEZ EAP EN TANT QUE SERVICE 11

**CHAPITRE 3. OUTILS POUR ASSISTER À LA MIGRATION ..... 12**
3.1. UTILISER WINDUP POUR ANALYSER LES APPLICATIONS POUR LA MIGRATION 12
3.2. UTILISER L'OUTIL DE MIGRATION SUR DU SERVEUR DE JBOSS POUR MIGRER LES CONFIGURATIONS DU SERVEUR. 12

**CHAPITRE 4. CHANGEMENTS DE CONFIGURATION DU SERVEUR ..... 14**
4.1. OPTIONS DE MIGRATION POUR LA CONFIGURATION DU SYSTÈME 14
4.2. OPÉRATION DE MIGRATION PAR LIGNE DE COMMANDES 14
4.3. CHANGEMENT AU NIVEAU DE LA JOURNALISATION DES PRÉFIXES DE MESSAGES 20
4.4. CHANGEMENTS DE CONFIGURATION DU SERVEUR WEB 20
4.5. CHANGEMENTS DE CONFIGURATION DU SERVEUR JGROUPS 27
4.6. CHANGEMENTS DE CONFIGURATION DU SERVEUR INFINISPAN 27
4.7. CHANGEMENTS DE CONFIGURATION DU SERVEUR DE MESSAGERIE 28
4.8. CHANGEMENTS DE CONFIGURATION DU SERVEUR ORB 37
4.9. MIGRER LA CONFIGURATION DU SOUS-SYSTÈME DE THREADS 40
4.10. MIGRER LA CONFIGURATION DU SOUS-SYSTÈME DISTANT 40
4.11. CHANGEMENTS DE CONFIGURATION DU SERVEUR WEBSOCKET 41
4.12. CHANGEMENTS DU SERVEUR « SINGLE SIGN-ON » 41
4.13. CHANGEMENTS DANS LA CONFIGURATION DE LA SOURCE DE DONNÉES 42
4.14. CHANGEMENTS DE CONFIGURATION DU SERVEUR DE SÉCURITÉ 43
4.15. MODIFICATIONS APPORTÉES AU SOUS-SYSTÈME DE TRANSACTIONS 43
4.16. CHANGEMENTS À LA CONFIGURATION DE MOD_CLUSTER 44

**CHAPITRE 5. CHANGEMENTS APPORTÉS À LA MIGRATION DES APPLICATIONS ..... 45**
5.1. CHANGEMENTS APPORTÉS AUX APPLICATIONS DE SERVICES WEB 45
5.2. METTEZ À JOUR LE PORT ET LE CONNECTEUR D'URL DISTANTS 50
5.3. CHANGEMENTS DANS L'APPLICATION DE MESSAGERIE 50
5.4. CHANGEMENTS DANS LES APPLICATIONSJAX-RS ET RESTEASY 52
5.5. CHANGEMENTS À L'APPLICATION CDI 1.2 58
5.6. MIGRER LES DÉPENDANCES DE MODULE EXPLICITES 60
5.7. CHANGEMENTS DE MIGRATION HIBERNATE ET JPA 61
5.8. CHANGEMENTS APPORTÉS À HIBERNATE SEARCH 61
5.9. MIGRER DES BEANS D'ENTITÉ CMP VERS JPA 68
5.10. CHANGEMENTS DANS LES PROPRIÉTÉS DE PERSISTENCE JPA 69
5.11. MIGREZ LE CODE DU CLIENT EJB 70
5.12. MIGRER LES CONFIGURATIONS DES PLANS DE DÉPLOIEMENT 72

5.13. MIGRER DES VALVES D'APPLICATIONS PERSONNALISÉES 72
5.14. CHANGEMENTS DANS LES APPLICATIONS DE SÉCURITÉ 73
5.15. CHANGEMENTS DE LA JOURNALISATION DANS JBOSS 73
5.16. CHANGEMENTS AU CODE JAVASERVER FACES (JSF) 74
5.17. CHANGEMENTS AU NIVEAU DU CHARGEMENT DES CLASSES DE MODULES 75
5.18. CHANGEMENTS CLUSTERING D'APPLICATIONS 76

**CHAPITRE 6. DIVERS CHANGEMENTS 81**
6.1. CHANGEMENT DANS LE MODE DE LIVRAISON DE JBOSS EAP NATIVES ET APACHE HTTP SERVER 81
6.2. CHANGEMENTS AUX DÉPLOIEMENTS DANS AMAZON EC2 82

**ANNEXE A. MATÉRIEL DE RÉFÉRENCE 83**
A.1. AVERTISSEMENTS POUR L'OPÉRATION DE MIGRATION DU SOUS-SYSTÈME JACORB 83
A.2. AVERTISSEMENTS POUR L'OPÉRATION DE MIGRATION DU SOUS-SYSTÈME DE MESSAGERIE 84
A.3. AVERTISSEMENTS POUR L'OPÉRATION DE MIGRATION DU SOUS-SYSTÈME WEB 89
A.4. COMPATIBILITÉ ET INTEROPÉRABILITÉ ENTRE LES VERSIONS 92

# CHAPITRE 1. INTRODUCTION

## 1.1. RED HAT JBOSS ENTERPRISE APPLICATION PLATFORM 7

Red Hat JBoss Enterprise Application Platform 7 (JBoss EAP) est une plateforme middleware générée sur la base de standards ouverts et conforme aux spécifications Java Enterprise Edition 7. JBoss EAP 7 intègre WildFly Application Server 10 avec une messagerie, un clustering de haute disponibilité et autres technologies.

JBoss EAP inclut une structure modulaire qui permet aux services d'être activés lorsqu'ils sont requis uniquement, améliorant ainsi la vitesse de démarrage.

La console de gestion et l'interface en ligne de commmande (CLI) rendent la modification des fichiers de configuration XML inutile et ajoutent la capacité d'encoder et d'automatiser des tâches.

JBoss EAP fournit deux modes d'opération pour les instances JBoss EAP : serveur autonome ou domaine géré. Le mode d'opération serveur autonome correspond à l'exécution de JBoss EAP en tant qu'unique instance du serveur. Le mode d'opération de domaine géré permet la gestion de multiples instances JBoss EAP à partir d'un point de contrôle unique.

En outre, JBoss EAP comprend des frameworks de développement et des API pour développer rapidement des applications Java EE sécurisées et évolutives.

## 1.2. GUIDE DE MIGRATION

Ce guide vise à documenter les changements requis pour exécuter et déployer correctement les applications Red Hat JBoss Enterprise Application Platform 6 sur Red Hat Enterprise Application Platform 7. Il fournit des informations sur la façon de résoudre des problèmes de runtime et de déploiement et comment éviter des changements de comportement de l'application. Il s'agit de la première étape du passage à la nouvelle plateforme. Une fois que l'application sera correctement déployée et en cours d'exécution, vous pourrez alors planifier la mise à niveau des composants individuels pour utiliser les nouvelles fonctions et fonctionnalités de JBoss EAP 7.

## 1.3. À PROPOS DES MIGRATIONS ET DES MISES À NIVEAU

### Mises à niveau majeures

Une mise à niveau majeure ou une migration est requise lorsqu'une application passe d'une version à une autre, par exemple de JBoss EAP 6 à JBoss EAP 7. Ce guide s'adresse à ce type de migration. Si une application suit les spécifications Java EE, n'accède pas aux API déconseillées, et ne contient pas de code propriétaire, il est alors possible d'exécuter l'application dans JBoss EAP 7 sans aucun changement dans le code de l'application. Cependant, si la configuration du serveur ou tout paramètre de configuration du serveur ont été modifiés dans JBoss EAP 7, une migration s'impose.

### Mises à jour mineures

JBoss EAP fournit périodiquement de nouvelles versions, qui sont des mises à jour mineures incluant des correctifs de bogues et des correctifs de sécurité, ainsi que de nouvelles fonctionnalités. Le guide JBoss EAP *Guide de correctifs et de mise à niveau* décrit comment mettre à niveau d'une version à l'autre, comme par exemple comment passer de JBoss EAP 7.0 à JBoss EAP 7.1.

### Correctifs cumulatifs

JBoss EAP fournit également des patchs cumulatifs périodiquement. Ceux-ci contiennent des correctifs de bogues et des correctifs de sécurité. Les patchs cumulatifs incrémentent le dernier entier de la version, par exemple de 7.0.0 à 7.0.1. L'installation des patchs est expliquée en détails dans le guide JBoss EAP *Guide de correctifs et de mise à niveau*.

## 1.4. L'UTILISATION DE EAP_HOME DANS CE DOCUMENT

Dans ce document, la variable **EAP_HOME** est utilisée pour indiquer le chemin d'installation de JBoss EAP. Veuillez remplacer cette variable par le chemin réel dans votre installation JBoss EAP.

- Si vous avez installé JBoss EAP en utilisant la méthode ZIP, le répertoire d'installation **jboss-eap-7.0** est le répertoire d'où vous avez extrait l'archive zip.
- Si vous avez installé JBoss EAP en utilisant la méthode d'installation RPM, le répertoire d'installation est **/opt/rh/eap7/root/usr/share/wildfly/**.
- Si vous utilisez l'installateur pour installer JBoss EAP, le chemin par défaut de **EAP_HOME** est **${user.home}/EAP-7.0.0** :
  - Dans Red Hat Enterprise Linux, Solaris, et HP-UX : **/home/USER_NAME/EAP-7.0.0/**
  - Dans Microsoft Windows : **C:\Users\USER_NAME\EAP-7.0.0**
- Si vous utilisez l'installateur JBoss Developer Studio pour installer et configurer le serveur JBoss EAP, le chemin par défaut de **EAP_HOME** est **${user.home}/jbdevstudio/runtimes/jboss-eap** :
  - Pour Red Hat Enterprise Linux: **/home/USER_NAME/jbdevstudio/runtimes/jboss-eap/**
  - Pour Microsoft Windows: **C:\Users\USER_NAME\jbdevstudio\runtimes\jboss-eap** or **C:\Documents and Settings\USER_NAME\jbdevstudio\runtimes\jboss-eap**

**Note**

**EAP_HOME** n'est pas une variable d'envirionnement. **JBOSS_HOME** est la variable d'environnement utilisée dans les scripts.

# CHAPITRE 2. PRÉPARATION À LA MIGRATION

## 2.1. VUE D'ENSEMBLE DES PRÉPARATIFS

Sur JBoss EAP 7, un effort a été fourni pour offrir une rétro-compatibilité pour les applications JBoss EAP 6. Cependant, si votre application utilise des fonctionnalités déconseillées ou des fonctionnalités ayant été supprimées de JBoss EAP 7, vous devrez apporter des changements au code de votre application.

En outre, de nombreux éléments ont changé dans cette version et peuvent avoir un impact sur le déploiement des applications JBoss EAP 7. Il est recommandé de faire des recherches et de planifier correctement avant de tenter de migrer votre application.

- Se familiariser avec les fonctionnalités de Java EE 7.
- Voir les nouveautés de JBoss EAP 7.
- Voir la liste des fonctionnalités déconseillées et non prises en charge..
- Voir le matériel du guide JBoss EAP 7 *Guide de démarrage*.
- Voir les outils offrant de l'aide avec les tâches liées à la migration..

Une fois à l'aide avec les changements de fonctionnalités, les éléments de développement, et les outils pouvant vous assister pour effectuer la migratopm, vous pourrez commencer à évaluer vos applications et la configuration de votre serveur pour déterminer quels changements seront nécessaire pour exécuter JBoss EAP 7.

## 2.2. VOIR LES FONCTIONNALITÉS JAVA EE 7

Java EE 7 inclut de nombreuses améliorations pour faciliter le développement et l'exécution d'applications riches en fonctionnalités sur des clouds publics et privés. Java EE 7 incorpore de nouvelles fonctionnalités et les standards les plus récents, tels que HTML5, WebSocket, JSON, Batch, et Concurrency Utilities. Les mises à jour incluent 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, et Bean Validation 1.1.

Davantage d'informations sur Java EE 7, y compris des tutoriels, se trouvent sur le site web d'Oracle : « Java EE at a Glance »

## 2.3. VOIR LES NOUVEAUTÉS DANS JBOSS EAP 7

JBoss EAP 7 inclut quelques mise à niveau et améliorations remarquables en comparaison avec la version précédente.

**Java EE 7**

JBoss EAP 7 est une implémentation certifiée de Java EE 7, répondant au profil Web et et profil complet (profil « Full »). JBoss EAP 7 inclut également la prise en charge des itérations les plus récentes de CDI 1.2 et Web Sockets 1.1.

**Undertow**

Undertow est le nouveau serveur web léger, flexible, et performant inclus dans JBoss EAP 7, remplaçant ainsi JBoss Web. Écrit en Java, Undertow est conçu pour un débit et une évolutivité maximale. Il prend en charge les technologies plus récentes, comme le nouveau standard HTTP/2.

**Apache ActiveMQ Artemis**

Apache ActiveMQ Artemis est le nouveau courtier de messagrie inclus dans JBoss EAP 7. Basé sur une donation de code de HornetQ, ce sous-projet Apache fournir des performances exceptionnelles basées sur une architecture non-bloquante prouvée.

**IronJacamar 1.2**

La version la plus récente d'IronJacamar fournit une prise en charge stable et riche en fonctionnalités pour JCA et DataSources.

**JBossWS 5**

La cinquième génération de JBossWS est un réel bond en avant, offrant de nouvelles fonctionnalités et améliorations des performances à JBoss EAP 7 Web Services.

**RESTEasy 3**

JBoss EAP 7 inclut la dernière génération RESTEasy. Cette version va au-delà des API REST Java EE standards (JAX-RS 2.0) en fournissant un grand nombre d'extensions utiles, comme le cryptage JSON Web, Jackson, YAML, JSON-P, et Jettison.

**OpenJDK ORB**

JBoss EAP 7 a remplacé l'implémentation JacORB IIOP par une branche en aval d'OpenJDK ORB, permettant ainsi une meilleure interopérabilité avec JVM ORB et Java EE RI.

**Clustering riche en fonctionnalités**

La prise en charge du clustering a été largement remaniée sur JBoss EAP 7 et inclut désormais plusieurs API publiques pour permettre l'accès des applications.

**Réduction de port**

En utilisant la mise à niveau HTTP, JBoss EAP 7 a déplacé presque tous ses protocoles pour qu'ils soient multiplexés sur deux ports HTTP uniquement : un port de gestion (9990), et un port d'application (8080).

**Journalisation améliorée**

L'API de gestion prend désormais en charge la capacité de répertorier et d'afficher les fichiers journaux sur un serveur, ou même de définir des formateurs autres que le formateur de schémas par défaut. La configuration de la journalisation de déploiement est également fortement améliorée.

Pour une liste complète des nouvelles fonctionnalités, veuillez consulter Nouvelles fonctionnalités et améliorations dans le guide JBoss EAP 7 *Notes de sortie*.

## 2.4. VOIR LA LISTE DES FONCTIONNALITÉS DÉPRÉCIÉES ET NON PRISES EN CHARGE.

Avant de migrer votre application, prenez en compte le fait que certaines fonctionnalités qui étaient disponibles dans des versions précédentes de JBoss EAP peuvent être déconseillées ou ne plus être prises en charge.

La prise en charge de certaines technologies a été supprimée en raison des coûts de maintenance élevés, d'un faible interêt de la part de la communauté et de solutions alternatives bien meilleures. Voici un petit récapitulatif de certaines des fonctionnalités non prises en charge.

**EJB Entity Beans**

Les EJB Entity Beans ne sont plus pris en charge. Si votre application utilise des EJB Entity Beans, vous devrez migrer le code pour utiliser JPA, ce qui fournir une API bien plus performante et flexible.

**JAX-RPC**

Comme JAX-WS offre une solution bien plus précise et complète, le code écrit pour JAX-RPC devrait être migré pour utiliser JAX-WS.

**JSR-88**

La spécification de l'API de déploiement d'applications Java EE (JSR-88), qui définissait un contrat pour activer des outils à partir de multiples fournisseurs pour configurer et déployer des applications sur tout produit de plateforme Java EE, n'était pas largement adoptée. Il faut également utiliser une autre option prise en charge par JBoss EAP pour le déploiement d'applications, comme la console de gestion, l'interface de ligne de commande de gestion, le scanner de déploiement, ou Maven.

**Adaptateur de ressources générique JMS**

La possibilité de configurer un adaptateur de ressources JMS standard à un fournisseur JMS n'est plus prise en charge dans JBoss EAP 7.

Pour une liste complète des fonctionnalités dépréciées qui ne sont plus prises en charge, veuillez consulter Fonctionnalités dépréciées non prises en charge dans le guide JBoss EAP 7 *Notes de sortie*.

## 2.5. VOIR LES DOCUMENTS DE MISE EN ROUTE DE JBOSS EAP.

Veillez à bien lire le *Guide de démarrage*. Il contient des informations importantes :

- Comment télécharger et installerJBoss EAP 7
- Comment télécharger et installer Red Hat JBoss Developer Studio
- Comment configurer Maven pour votre environnement de déploiement, gérer des dépendances de projet, et configurer vos projets pour utiliser les artefacts JBoss EAP BOM (« Bill of Material »)
- Comment télécharger et exécuter des exemples d'applications Quickstart fournies avec ce produit

## 2.6. ANALYSE ET PLANIFICATION DE LA MIGRATION

Chaque configuration d'application et de serveur est unique et vous devez comprendre tous les composants et l'architecture de la plateforme d'application et de serveur avant de tenter d'effectuer une migration. Votre plan de migration doit inclure une feuille de route détaillée pour tester et passer à la production, qui prenne en compte les informations suivantes :

**Identifier les personnes disponibles pour le processus de migration**

Identifier les parties intéressées, les chefs de projet, les développeurs, les administrateurs, et tous ceux qui sont responsables de la migration.

**Vérifier la configuration de la plateforme de serveur d'applications et le matériel**

Examinez la configuration de la plateforme et du serveur d'applications existants pour voir comment ils sont affectés par des changements de fonctionnalités dans JBoss EAP 7. L'examen doit inclure les éléments suivants :

- Les versions et les systèmes d'exploitation
- La base de données utilisée par les applications
- Les serveurs web
- L'architecture de sécurité
- Le nombre et types de processeurs
- Le montant de mémoire
- Le montant de stockage de disques physiques
- Migration de la base de données ou des données de messagerie
- Autres composants qui ont pu être affectés par la migration

**Vérifier l'environnement de production actuel**

Vous devez prévoir de recréer l'environnement de production aussi proche que possible pour les essais et les mises en stage intermédiaires du processus de migration.

- Prendre en compte les configurations de clusterisation. Voir la section Mise à jour d'un cluster dans le *Guide de correctifs et de mise à jour* de JBoss EAP pour plus d'informations sur la façon de migrer des clusters.
- Si vous exécutez dans un grand domaine géré, considérez une approche graduelle pour la migration.
- Déterminer si vous avez besoin de migrer des données de messagerie ou de base de données.

**Examiner et comprendre l'application existante**

Bien examiner et comprendre l'application JBoss EAP 6 existante. Familiarisez-vous avec son architecture, ses fonctionnalités, ses caractéristiques et ses composants, y compris :

- La version JVM
- Intégration avec d'autres composants middleware de serveur d'applications
- Intégration avec un logiciel propriétaire de tierce partie
- Utilisation des fonctionnalités qui auront besoin d'être remplacées

- La configuration de l'application inclut des descripteurs de déploiement, JNDI, la persistance, la configuration et le pooling de JDBC, les sujets et files d'attente JMS, et la journalisation.

Identifier des incompatibilités de code ou de configuration qui auront besoin d'être modifiées pour la migration vers JBoss EAP 7.

**Créer un plan de test détaillé**

- Le plan doit inclure un test de régression et des critères d'acceptabilité.
- Il doit également comprendre un test de performance.
- Mettez en place un environnement de test intermédiaire aussi proche que possible de l'environnement de production pour tester la migration avant de la faire passer en production.
- Veillez bien à créer un plan de sauvegarde et un plan de retrait !

**Examinez les ressources disponibles pour le processus de migration**

- Évaluez les aptitudes de l'équipe de développement et planifiez la formation ou une aide supplémentaire de la part de consultants.
- Réalisez que vous aurez besoin de ressources et de matériel supplémentaires pour les étapes de staging et de test pendant le processus de migration jusqu'à ce que cette tâche soit complétée.
- Déterminer si une formation formelle est utile. Si tel est le cas, ajoutez-là au calendrier.

**Exécuter le plan**

Rassemblez les ressources nécessaires et implémentez le plan de migration.

**Important**

Avant d'apporter toute modification à votre application, assurez-vous de bien créer une copie de sauvegarde.

## 2.7. SAUVEGARDEZ VOS DONNÉES IMPORTANTES ET VÉRIFIEZ L'ÉTAT DE VOTRE SERVEUR

Avant de migrer votre application, vous devez être conscients des problèmes qui peuvent se présenter.

- La migration peut supprimer des dossiers temporaires. Tout déploiement stocké dans **`data/content/`** doit être sauvegardé avant la migration et restauré une fois que la migration est terminée. Sinon, le serveur ne pourra pas démarrer à cause du contenu manquant.
- Avant la migration, gérer les transactions en cours, et supprimer le répertoire de transactions **`data/tx-object-store/`**.
- Les données de minuteur persistantes **`data/timer-service-data`** doivent être vérifiées pour déterminer si elles sont compatibles. Avant la migration, vérifier les fichiers de **`deployment-*`** dans ce répertoire pour déterminer quels minuteurs sont en cours d'utilisation.

Veillez à sauvegarder les applications et la configuration du serveur actuelles avant de commencer.

## 2.8. MIGRER UNE INSTALLATION RPM

**Important**

Avoir plus d'une instance de RPM installée de JBoss EAP sur un seul serveur Red Hat Enterprise Linux n'est pas pris en charge. De ce fait, nous conseillons de migrer votre installation JBoss EAP vers une autre machine quand vous migrerez vers JBoss EAP 7.

Quand vous migrez une installation RMP de JBoss EAP de JBoss EAP 6 à JBoss EAP 7, veillez à ce que JBoss EAP 7 soit installé sur une machine qui n'a pas d'installation RPM de JBoss EAP existante.

Pour installer JBoss EAP 7 avec les RPM, voir le guide JBoss EAP *Guide d'installation*.

Le conseil de migration de ce guide s'applique également à la migration d'installations RPM de JBoss EAP, mais vous devrez modifier certaines étapes (comme la façon de démarrer JBoss EAP) pour que cela convienne à une installation RPM, comparé à un ZIP ou à une installation par programme d'installation.

## 2.9. MIGRATION DE JBOSS EAP SI VOUS EXÉCUTEZ EAP EN TANT QUE SERVICE

Si vous exécutez JBoss EAP 6 en tant que service, veillez à consulter à nouveau Configurer JBoss EAP en tant que Service dans le *Guide d'installation* sur la façon de configurer JBoss EAP 7.

# CHAPITRE 3. OUTILS POUR ASSISTER À LA MIGRATION

## 3.1. UTILISER WINDUP POUR ANALYSER LES APPLICATIONS POUR LA MIGRATION

Windup, qui est fourni avec JBoss Migration Toolkit, est un outil extensible et personnalisable basé sur des règles qui permet de simplifier la migration d'applications Java. Cet outil analyse les API, les technologies et les architectures utilisées par les applications que vous souhaitez migrer et fournit des rapports de migration détaillés pour chaque application. Ces rapports procurent les informations suivantes.

- Explications détaillées des changements de migration utiles
- Si le changement rapporté est obligatoire ou en option
- Si le changement rapporté est complexe ou trivial
- Liens vers le code qui requiert le changement de migration
- Astuces et liens vers des informations qui exliquent comment effectuer les changements nécessaires
- Une estimation sur le niveau d'effort pour chaque problème de migration trouvé et le montant total d'efforts pour migrer l'application

Vous pouvez utiliser Windup pour analyser le code et l'architecture de vos applications JBoss EAP 7. La règle Windup définie pour la migration de JBoss EAP 6 à JBoss EAP 7 se rapporte à des descripteurs XML et à des paramètres et codes d'application spécifiques qui doivent être remplacés par une autre configuration lors de la migration à JBoss EAP 7.

Pour obtenir des informations supplémentaires sur la façon d'utiliser Windup pour analyser vos applications JBoss EAP6, consulter *Windup User Guide*.

## 3.2. UTILISER L'OUTIL DE MIGRATION SUR DU SERVEUR DE JBOSS POUR MIGRER LES CONFIGURATIONS DU SERVEUR.

L'outil JBoss Server Migration, actuellement en cours de développement, sera l'outil de choix pour mettre votre configuration à jour, pour qu'elle inclue les nouvelles fonctionnalités et les nouveaux paramètres de configuration dans JBoss EAP 7, tout en conservant votre configuration existante.

L'outil JBoss Server Migration lit votre fichier de configuration de serveur JBoss EAP 6 et ajoute des configurations supplémentaires aux nouveaux sous-systèmes, met à jour les configurations du sous-système existant avec les nouvelles fonctionnalités, et supprime les configurations obsolètes du sous-système.

La dernière distribution binaire de la pré-version de l'outil JBoss Server Migration Tool est prête pour le téléchargement https://github.com/wildfly/wildfly-server-migration/releases. Cette version supporte la migration des serveurs autonomes de JBoss EAP 6.4 à JBoss EAP 7.0. Pour obtenir des informations générales sur la façon d'exécuter l'outil, consulter le guide d'utilisation Outil de migration du serveur JBoss Server Migration Tool *Guide d'utilisateur*.

**Note**

La version pré-version de l'outil JBoss Server Migration Tool n'est pas prise en charge pour le moment.

# CHAPITRE 4. CHANGEMENTS DE CONFIGURATION DU SERVEUR

## 4.1. OPTIONS DE MIGRATION POUR LA CONFIGURATION DU SYSTÈME

Pour migrer votre configuration de serveur de JBoss EAP 6 à JBoss EAP 7, vous pouvez soit utiliser l'outil JBoss Server Migration ou procéder à une migration manuelle par l'opération d'interface en ligne de commande (CLI) **`migrate`**.

### Outil JBoss Server Migration

L'outil JBoss Server Migration, qui est actuellement en cours de développement, est la méthode préconisée afin de mettre à jour votre configuration, pour qu'elle puisse inclure les dernières fonctionnalités et paramètres de configuration de JBoss EAP 7, tout en conservant votre configuration existante.

### Opération de Migration par l'interface en ligne de commande

Vous pouvez utiliser l'opération d'interface CLI **`migrate`** pour mettre à jour les sous-systèmes **`jacorb`**, **`messaging`**, et **`web`** du fichier de configuration de JBoss EAP 6 pour qu'ils exécutent sur la nouvelle version, mais sachez que le résultat n'est pas une configuration JBoss EAP 7 complète. Par exemple :

- L'opération ne met pas les configurations de port et le protocole **`distant`** à jour au nouveau **`http-remoting`**, et les nouvelles configurations de port sont maintenant utilisées dans JBoss EAP 7.
- La configuration n'inclut pas les nouveaux sous-systèmes JBoss EAP ou les fonctionnalités comme les déploiements singleton clusterisés, ou encore, la fermeture normale.
- La configuration n'inclut pas les nouvelles fonctionnalités de Java EE 7 comme le traitement par lots.

Pour plus d'informations sur la façon d'utiliser l'opération **`migrate`** pour migrer la configuration du serveur, voir Opération de Migration par l'Interface en ligne de commandes.

## 4.2. OPÉRATION DE MIGRATION PAR LIGNE DE COMMANDES

Vous pouvez utiliser l'interface en ligne de commandes pour mettre à jour les fichiers de configuration du serveur JBoss EAP 6 afin qu'ils puissent être exécutés sur JBoss EAP 7. L'interface de ligne de commandes offre une opération **`migrate`**, permettant de mettre à jour automatiquement les sous-systèmes **`jacorb`**, **`messaging`**, et **`web`** de la version précédente à la nouvelle configuration. Vous pouvez également exécuter l'opération **`describe-migration`** pour les sous-systèmes **`jacorb`**, **`messaging`**, et **`web`** pour examiner les changements de configuration proposés de la migration avant d'effectuer la migration. Il n'y a pas de remplacement pour les sous-systèmes **`cmp`**, **`jaxr`**, ou **`threads`** et ils doivent être supprimés de la configuration du serveur.

**Important**

Voir Options de migration pour la configuration du serveur pour avoir des informations sur les limitations de l'opération **`migrate`**. L'outil JBoss Server Migration, qui est actuellement en cours de développement, est la méthode préconisée pour mettre à jour votre configuration, afin qu'elle puisse inclure les dernières fonctionnalités et paramètres de configuration de JBoss EAP 7, tout en conservant votre configuration existante.

**Tableau 4.1. Opérations de l'interface de ligne de commandes et de migration de sous-systèmes**

| Sous-système JBoss EAP 6 | Sous-système JBoss EAP 7 | Opération en ligne de commandes |
|---|---|---|
| cmp | no replacement | remove |
| jacorb | iiop-openjdk | migrate |
| jaxr | no replacement | remove |
| messaging | messaging-activemq | migrate |
| threads | no replacement | remove |
| web | undertow | migrate |

Effectuez les étapes ci-dessous pour mettre à jour votre configuration de serveur JBoss EAP 6 pour une exécution sur JBoss EAP 7.

1. Avant de commencer, vérifier Back Up Important Data and Review Server State. Il contient des informations importantes sur la façon de vérifier que le serveur est en bon état, et que les bons fichiers sont sauvegardés.
2. Lancez le serveur JBoss EAP 7 avec la configuration JBoss EAP 6.
    a. Effectuez une copie de sauvegarde des fichiers de configuration du serveur JBoss EAP 7.
    b. Copiez le fichier de configuration de la version précédente dans le répertoire JBoss EAP 7.

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

c. Rendez-vous sur le répertoire d'installation JBoss EAP 7 et lancez le serveur avec l'argument **`--admin-only`**.

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

**Note**

Vous remarquerez les erreurs **`org.jboss.as.controller.management-operation`** suivantes dans le journal du serveur quand vous démarrez le serveur. Ces erreurs sont prévues et indiquent que les configurations du sous-système héritées doivent être supprimées et migrées dans JBoss EAP 7.

- WFLYCTL0402 : Les sous-systèmes [cmp] fournis par l'extension héritée « org.jboss.as.cmp » ne sont pas pris en charge sur les serveurs exécutant cette version. Le sous-système et l'extension doivent être supprimés ou doivent migrer pour que le serveur puisse fonctionner.
- WFLYCTL0402 : Les sous-systèmes [jacorb] fournis par l'extension héritée « org.jboss.as.jacorb » ne sont pas pris en charge sur les serveurs exécutant cette version. Le sous-système et l'extension doivent être supprimés ou doivent migrer pour que le serveur puisse fonctionner.
- WFLYCTL0402 : Les sous-systèmes [jaxr] fournis par l'extension héritée « org.jboss.as.jaxr » ne sont pas pris en charge sur les serveurs exécutant cette version. Le sous-système et l'extension doivent être supprimés ou doivent migrer pour que le serveur puisse fonctionner.
- WFLYCTL0402 : Les sous-systèmes [messaging] fournis par l'extension héritée « org.jboss.as.messaging » ne sont pas pris en charge sur les serveurs exécutant cette version. Le sous-système et l'extension doivent être supprimés ou doivent migrer pour que le serveur puisse fonctionner.
- WFLYCTL0402 : Les sous-systèmes [threads] fournis par l'extension héritée « org.jboss.as.threads » ne sont pas pris en charge sur les serveurs exécutant cette version. Le sous-système et l'extension doivent être supprimés ou doivent migrer pour que le serveur puisse fonctionner.
- WFLYCTL0402 : Les sous-systèmes [web] fournis par l'extension héritée « org.jboss.as.web » ne sont pas pris en charge sur les serveurs exécutant cette version. Le sous-système et l'extension doivent être supprimés ou doivent migrer pour que le serveur puisse fonctionner.

3. Ouvrez un nouveau terminal, rendez-vous sur le répertoire d'installation JBoss EAP 7, et lancez l'interface de ligne de commande de gestion en utilisant les arguments **`--controller=remote://localhost:9999`**.

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

4. Effectuez la migration des sous-systèmes **`jacorb`**, **`messaging`**, et **`web`**.

    a. Pour vérifier les changements de configuration qui seront apportés au sous-système avant d'effectuer la migration, veuillez exécuter l'opération **`describe-migration`**.

    L'opération **`describe-migration`** utilise la syntaxe suivante.

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

    L'exemple suivant décrit les changements de configuration qui seront apportés au fichier de configuration JBoss EAP 6.4 **`standalone-full.xml`** lorsqu'il est migré sur JBoss EAP 7. Les entrées ont été supprimées de la sortie pour améliorer la lisibilité et pour gagner de l'espace.

    **Exemple : opération describe-migration**

    ```
    [standalone@localhost:9999 /]
    /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")]
                }
            ]
        }
    }
    ```

b. Exécutez l'opération **`migrate`** pour migrer la configuration du sous-système qui le remplace dans JBoss EAP 7. L'opération utilise la syntaxe suivante.

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

**Note**

Les opérations du sous-système de **`messagerie`** **`describe-migration`** et **`migrate`** vous permettent de faire passer un argument pour configurer l'accès aux anciens clients. Pour plus d'informations sur la syntaxe de la commande, voir Migration du sous-système de messagerie et compatibilté ascendante.

c. Examinez le résultat de la commande. Assurez-vous que l'opération se termine correctement et qu'il n'y ait pas d'entrées « migration-warning ». Cela signifie que la configuration de la migration pour le sous-système est terminée.

### Exemple : opération de migration sans avertissement réussie

```
[standalone@localhost:9999 /] /subsystem=messaging:migrate
{
    "outcome" => "success",
    "result" => {"migration-warnings" => []}
}
```

Si vous voyez des entrées « migration-warnings » dans ce journal, cela indique que la migration de la configuration du serveur s'est correctement terminée, mais qu'il n'a pas été possible de migrer tous les éléments et attributs. Veuillez suivre les suggestions offertes par les entrées « migration-warnings » et exécutez des commandes de l'interface en ligne de commandes pour modifier ces configurations. Vous trouverez ci-dessous un exemple d'opération **`migrate`** qui retourne des « migration-warnings ».

### Exemple : opération migrate avec avertissements

```
[standalone@localhost:9999 /] /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."
    ]}
}
```

**Note**

La liste des avertissements **`migrate`** et **`describe-migration`** de chaque sous-système qui se trouve dans la section Matériel de référence à la fin de ce guide.

- Jacorb Subsystem Migration Operation Warnings
- Messaging Subsystem Migration Operation Warnings
- Web Subsystem Migration Operation Warnings

d. Examiner le fichier de configuration du serveur pour vérifier que l'extension, le sous-système, et l'espace-nom ont bien été mis à jour que la configuration existante du sous-système a effectivement migré sur JBoss EAP 7.

**Note**

Vous devrez répéter ce processus pour chaque sous-système **`jacorb`**, **`messaging`**, et **`web`** à l'aide des commandes suivantes.

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

5. Supprimez les sous-systèmes et extensions **`cmp`**, **`jaxr`**, et **`threads`** de la configuration du serveur.

   Alors que vous vous trouvez toujours dans l'invite de l'interface de ligne de commandes, supprimez les sous-systèmes obsolètes **`cmp`**, **`jaxr`**, et **`threads`** en exécutant les commandes suivantes.

```
/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
```

**Important**

Vous devez effectuer la migration des sous-systèmes **`messaging`**, **`jacorb`**, et **`web`** et supprimer les extensions et sous-systèmes **`cmp`**, **`jaxr`**, et **`threads`** avant de pouvoir redémarrer le serveur pour une opération normale. Si vous devez redémarrer le serveur avant de terminer ce processus, assurez-vous de bien inclure l'argument **`--admin-only`** sur la ligne de commande du démarrage du serveur. Ceci vous permet de continuer avec les changements de configuration.

## 4.3. CHANGEMENT AU NIVEAU DE LA JOURNALISATION DES PRÉFIXES DE MESSAGES

Les messages de journalisation ont pour préfixes le code du projet du sous-système qui rapporte le message. Les préfixes de tous les messages ont changé dans JBoss EAP 7.

Pour obtenir une liste de tous les préfixes de code de projets de messages de journalisation de JBoss EAP 7, voir Codes de projet utilisés dans JBoss EAP dans le *Guide de développement* de JBoss EAP.

## 4.4. CHANGEMENTS DE CONFIGURATION DU SERVEUR WEB

### 4.4.1. Remplacer le sous-système web d'Undertow

Undertow remplace JBoss Web comme serveur web dans JBoss EAP 7. Cela signifie que la configuration du sous-système **`web`** hérité doit être migrée vers la nouvelle configuration du sous-système **`undertow`** de JBoss EAP 7.

- L'espace-nom de la configuration du sous-système **`urn:jboss:domain:web:2.2`** dans le fichier de configuration du serveur a été remplacé par l'espace-nom **`urn:jboss:domain:undertow:3.1`**.
- Le module d'extension **`org.jboss.as.web`**, situé dans **`EAP_HOME/modules/system/layers/base/`**, a été remplacé par le module d'extension **`org.wildfly.extension.undertow`**.

Vous pouvez utiliser l'opération **`migrate`** dans l'interface en ligne de commande (CLI) pour migrer le sous-système **`web`** à **`undertow`** dans le fichier de configuration du serveur. Cependant, sachez que cette opération ne vous permettra pas de migrer toutes les configurations du sous-système de JBoss Web. Si vous apercevez des entrées "migration-warning", vous devrez exécuter des commandes supplémentaires pour migrer ces configurations vers Undertow. Pour obtenir davantage d'informations sur l'opération **`migrate`** de l'interface de ligne de commande de gestion, veuillez consulter les Opérations de migration en ligne de commande.

Ci-dessous figure un exemple de la configuration par défaut du sous-système **`web`** dans 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>
```

Ci-dessous figure un exemple de la configuration par défaut du sous-système **`undertow`** dans JBoss EAP 7.

```
<subsystem xmlns="urn:jboss:domain:undertow:3.1">
    <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>
```

### 4.4.2. Migrer les conditions de ré-écriture web de JBoss Web

L'opération **`migrate`** de ligne de commande ne peut pas migrer les conditions de 'rewrite' automatiquement. Elles sont rapportées en tant que "migration-warnings" et vous devez les migrer manuellement. Vous pouvez créer une configuration équivalente dans JBoss EAP 7 en utilisant Undertow Predicates Attributes and Handlers.

Ci-dessous figure un exemple de la configuration par défaut du sous-système **`web`** de JBoss EAP 6, qui contient une configuration **`rewrite`**.

```
<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>
```

Suivre les instructions de Opérations de migration en ligne de commande pour démarrer votre serveur et l'outil de ligne de commande, puis migrer le fichier de configuration du sous-système **`web`** en utilisant la commande suivante.

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

Les "migration-warnings" sont rapportés quand vous exécutez l'opération **`migrate`** sur la configuration ci-dessus.

```
e /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\")
    ]
}"
    ]}
}
```

Vérifiez le fichier de configuration du serveur et vous verrez la configuration suivante pour le sous-système **undertow**.

**Note**

La configuration de 'rewrite' a été abandonnée.

```
<subsystem xmlns="urn:jboss:domain:undertow:3.1">
     <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>
```

Utiliser l'outil de ligne de commande (CLI) pour créer le filtre qui remplacera la configuration de 'rewrite' dans le sous-système d'**Undertow**. Vous devrez voir "{"outcome" ⇒ "success"}" pour chaque commande.

```
# 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
```

Vérifiez le fichier de configuration du serveur mis à jour. Le sous-système de JBos Web est maintenant complètement migré et configuré dans le sous-système **`undertow`**.

```
<subsystem xmlns="urn:jboss:domain:undertow:3.1">
    <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>
```

Pour plus d'informations sur la façon de configurer les filtres et les handlers en utilisant la ligne de commande, voir Configurer le serveur web dans le *Guide de configuration* de JBoss EAP 7.

### 4.4.3. Migrer des valves globales

Versions précédentes des valves prises en charge par JBoss EAP. Les valves sont des classes personnalisées insérées dans le pipeline de traitement de requêtes pour une application avant que les filtres servlet apportent des changements à la requête ou effectuent un traitement supplémentaire.

- Les valves globales sont insérées dans le pipeline de traitement de requête de toutes les applications déployées et sont configurées dans le fichier de configuration du serveur.

- Les valves d'authentification authentifient les informations d'identification de la requête.
- Des valves d'applications personnalisées ont été créées en étendant la classe **`org.apache.catalina.valves.ValveBase`** et configurées dans l'élément **`<valve>`** du fichier de descripteur **`jboss-web.xml`**. Ces valves doivent être migrées manuellement.

Cette section décrit la façon de migrer les valves globales. La migration de valves d'authenficateur personnalisées est expliquée dans la section Migration des valves d'applications personnalisées de ce guide.

Undertow, qui remplace JBoss Web dans JBoss EAP 7, ne prend pas en charge les valves globales. Cependant, vous devriez pouvoir utiliser des fonctionnalités similaires en utilisant les gestionnaires Undertow. Undertow inclut un certain nombre de gestionnaires intégrés qui fournissent des fonctionnalités . Undertow offre également la possibilité de créer des gestionnaires personnalisés, qui peuvent être utilisés pour remplacer les fonctionnalités des valves personnalisées.

Si votre application utilise des valves, vous devez les remplacer par le code du gestionnaire Undertow approprié afin d'atteindre les mêmes fonctionnalités lorsque vous effectuez la migration vers JBoss EAP 7.

Pour plus d'informations sur la façon de configurer les handlers, voir Configuration des Handlers du *Guide de configuration* de JBoss EAP 7.

Pour plus d'informations sur la façon de configurer les filtres, voir Configuration des Filtres du *Guide de configuration* de JBoss EAP 7.

#### Migrer les valves de JBoss Web

Le tableau suivant répertorie les valves fournies par JBoss Web dans la version précédente de JBoss EAP et le handler intégré d'Undertow correspondant. Les valves de JBoss Web se trouvent dans le paquet **`org.apache.catalina.valves`**.

**Tableau 4.2. Mapper les valves avec les gestionnaires**

| Valve | Gestionnaire |
|---|---|
| AccessLogValve | io.undertow.server.handlers.accesslog.AccessLogHandler |
| CrawlerSessionManagerValve | io.undertow.servlet.handlers.CrawlerSessionManagerHandler |
| ExtendedAccessLogValve | io.undertow.server.handlers.accesslog.AccessLogHandler |
| JDBCAccessLogValve | Voir JDBCAccessLogValve Manual Migration Procedure ci-dessous pour obtenir des instructions. |
| RemoteAddrValve | io.undertow.server.handlers.IPAddressAccessControlHandler |

| Valve | Gestionnaire |
|---|---|
| RemoteHostValve | io.undertow.server.handlers.AccessControlListHandler |
| RemoteIpValve | io.undertow.server.handlers.ProxyPeerAddressHandler |
| RequestDumperValve | io.undertow.server.handlers.RequestDumpingHandler |
| RewriteValve | Voir Migrate JBoss Web Rewrite Conditions pour obtenir des instructions sur la façon de migrer ces valves manuellement. |
| StuckThreadDetectionValve | io.undertow.server.handlers.StuckThreadDetectionHandler |

Vous pouvez utiliser l'opération **`migrate`** de l'interface de ligne de commandes pour automatiquement migrer les valves globales qui remplissent les conditions suivantes :

- Elles se limitent aux valves listées dans le tableau précédent, et qui ne requièrent pas de traitement manuel.
- Elles doivent être définies dans le sous-système **`web`** du fichier de configuration du serveur.

Pour obtenir davantage d'informations sur l'opération **`migrate`**, veuillez consulter Opération de migration en interface de ligne de commande.

#### Procédure de migration manuelle JDBCAccessLogValve

La valve **`org.apache.catalina.valves.JDBCAccessLogValve`** est une exception à la règle et ne peut pas être migrée automatiquement dans le **`io.undertow.server.handlers.JDBCLogHandler`**. Suivre les étapes suivantes pour migrer l'exemple de valve suivant.

```
<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. Créer un module de pilote pour la base de données qui va stocker les entrées de journalisation.
2. Configurer la source de données de la base de données et ajouter le pilote à la liste des pilotes disponibles dans le sous-système des **`sources de données`**.

```
<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. Configurer un **`expression-filter`** dans le sous-système **`undertow`** avec l'expression suivante : **`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. CHANGEMENTS DE CONFIGURATION DU SERVEUR JGROUPS

### 4.5.1. JGroups est, par défaut, une Interface de réseau privé

Dans la confuguration par défaut deJBoss EAP 6, JGroups utilisait l'interface **`publique`** définie dans la section **`<interfaces>`** du fichier de configuration du serveur.

Comme il est normalement conseillé d'utiliser une interface de réseau dédiée exclusivement, par défaut, JGroups utilise maintenant l'interface **`privée`** définie dans la section **`<interfaces>`** du fichier de configuration du serveur dans JBoss EAP 7.

### 4.5.2. Changements aux canaux de JGroups

JGroups fournit un support de communication de groupe pour les services HA, sous forme de canaux de JGroups. JBoss EAP 7 introduit des éléments de **`< canaux>`** dans le sous-système **`jgroups`** qui se trouve dans le fichier de configuration de serveur. Vous pouvez ajouter, supprimer ou modifier la configuration de canaux JGroups par l'outil de ligne de commandes.

Pour obtenir des informations sur la façon de configurer JGroups, voir Communication Cluster avec JGroups dans le *Guide de configuration* de JBoss EAP.

## 4.6. CHANGEMENTS DE CONFIGURATION DU SERVEUR INFINISPAN

### 4.6.1. Changements dans la configuration des caches par défaut d'Infinispan

Dans JBoss EAP 6, la valeur par défaut des caches en cluster des réplications de sessions web et EJB étaient des caches **ASYNC** répliqués. Cela a changé dans JBoss EAP 7. Les caches en cluster par défaut sont maintenant des caches **ASYNC** distribués. Les caches répliqués ne sont même plus configurés par défaut. Voir Configurer le mode de cache dans le *Guide de configuration* pour obtenir des informations sur la façon d'ajouter un cache répliqué et de le rendre 'par défaut'.

Cela ne vous affectera que lorsque vous utiliserez la nouvelle configuration par défaut de JBoss EAP 7. Si vous migrez la configuration de JBoss EAP 6, la configuration du sous-système **infinispan** sera préservée.

### 4.6.2. Changements dans la stratégie de caches d'Infinispan

Le comportement de la stratégie de cache **ASYNC** a changé dans JBoss EAP 7.

Dans JBoss EAP 6, les lectures de caches **ASYNC** n'avaient pas de verrous. Bien qu'elles ne pouvaient être bloquées, elles étaient prônes aux lectures erronées des données obsolètes, par exemple, en cas de basculement. C'est parce que cela permettait aux demandes consécutives d'un même utilisateur de démarrer avant que la requête précédente fût terminée. Cette permissivité n'est pas acceptable en mode distribué, étant donné que les modifications de topologie de cluster peuvent avoir un impact sur l'affinité de session et facilement entraîner des données obsolètes.

Dans JBoss EAP 7, les lectures de caches **ASYNC** requièrent des verrous. Comme elles bloquent maintenant les nouvelles demandes de la part d'un même utilisateur jusqu'à ce que la réplication se termine, cela évite les lectures erronnées.

## 4.7. CHANGEMENTS DE CONFIGURATION DU SERVEUR DE MESSAGERIE

Sur JBoss EAP 7, ActiveMQ Artemis remplace HornetQ comme fournisseur de support JMS. Cette section décrit comment migrer la configuration et les données connexes de messagerie.

### 4.7.1. Changements de configuration du serveur du sous-système de messagerie

Le module d'extension **org.jboss.as.messaging**, situé dans **EAP_HOME/modules/system/layers/base/**, a été remplacé par le module d'extentsion **org.wildfly.extension.messaging-activemq**.

L'espace-nom de la configuration du sous-système **urn:jboss:domain:messaging:3.0** a été remplacé par l'espace-nom **urn:jboss:domain:messaging-activemq:1.0**.

#### Modèle de gestion

Dans la plupart des cas, un effort a été fait pour garder les noms d'éléments et d'attributs aussi proches que possible de ceux utilisés dans des versions précédentes. Le tableau suivant répertorie certains de ces changements.

**Tableau 4.3. Mapper les attributs de messagerie**

| Nom HornetQ | Nom ActiveMQ |
|---|---|
| hornetq-server | serveur |
| hornetq-serverType | serverType |
| connecteurs | connecteur |
| discovery-group-name | discovery-group |

Les opérations de gestion invoquées sur le nouveau sous-système **`messaging-activemq`** sont passées de **`/subsystem=messaging/hornetq-server=`** à **`/subsystem=messaging-activemq/server=`**.

Vous pouvez migrer une configuration du sous-système de messagerie **`messaging`** de JBoss EAP 6 dans le sous-système **`messaging-activemq`** d'un serveur JBoss EAP 7 en invoquant son opération **`migrate`**.

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

Avant d'exécuter l'opération **`migrate`**, vous devriez invoquer l'opération **`describe-migration`** pour réviser la liste des opérations de gestion qui vont être effectuées pour migrer la configuration du sous-système de messagerie **`messaging`** JBoss EAP 6 dans le sous-système **`messaging-activemq`** du serveur JBoss EAP 7.

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

Les opérations **`migrate`** et **`describe-migration`** affichent également une liste d'avertissements **`migration-warnings`** pour les ressources ou attributs qui ne peuvent pas être migrés automatiquement.

#### Migration du sous-système de messagerie et Compatibilté ascendante

Les opérations **`describe-migration`** et **`migrate`** du sous-système **`messaging`** fournissent un argument de configuration supplémentaire. Si vous souhaitez configurer la messagerie pour permettre aux anciens clients JBoss EAP 6 de se connecter au serveur JBoss EAP 7, vous pouvez ajouter un argument **`add-legacy-entries`** à l'opération **`describe-migration`** ou **`migrate`**, comme suit.

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

Si l'argument de booléen **`add-legacy-entries`** est défini sur **`true`**, le sous-système **`messaging-activemq`** crée la ressource **`legacy-connection-factory`** et ajoute les **`legacy-entries`** aux ressources **`jms-queue`** et **`jms-topic`**.

Si l'argument de booléen **`add-legacy-entries`** est défini sur **`false`**, aucune ancienne ressource ne sera créée dans le sous-système **`messaging-activemq`** et les anciens clients JMS ne seront pas en mesure de communiquer avec les serveurs de JBoss EAP 7. Il s'agit de la valeur par défaut.

Pour obtenir des informations sur la compatibilité ascendante et descendante, consulter Compatibilité ascendante et descendante dans *Configurer la messagerie* de JBoss EAP.

Pour obtenir davantage d'informations sur les opérations **`migrate`** et **`describe-migration`** en ligne de commande, veuillez consulter Opération de migration en interface de ligne de commande.

**Configuration du sous-système de messagerie XML**

La configuration XML a changé de manière significative avec le nouveau sous-système **`messaging-activemq`** afin de fournir un schéma XML plus consistant avec d'autres sous-systèmes JBoss EAP.

Il est fortement déconseillé de tenter de modifier la configuration XML du sous-système de messagerie **`messaging`** JBoss EAP afin de se conformer au nouveau sous-système **`messaging-activemq`**. Au lieu de cela, veuillez invoquer l'opération **`migrate`** du sous-système hérité. Cette opération écrira la configuration XML du nouveau sous-système messaging-activemq comme faisant partie de son exécution.

### 4.7.2. Changements de la journalisation de la messagerie

Le préfixe des messages de journalisation du sous-système de messagerie, **`messaging`**, est passé de "JBAS" à "WFLYMSGAMQ". Pour obtenir une liste de tous les préfixes de code de projets de messages de journalisation de JBoss EAP 7, voir Codes de projets utilisés dans JBoss EAP dans le *Guide de développement* de JBoss EAP.

### 4.7.3. Migrer les données de messagerie

En raison du passage du fournisseur de Support JMS d'HornetQ à ActiveMQ Artemis, le format et l'emplacement des données de messagerie ont changé dans JBoss EAP 7.

Vous pouvez adopter les deux approches suivantes pour migrer les données :

- Vous pouvez exporter les données de messagerie depuis la dernière version et l'importer par l'opération **`import-journal`** en ligne de commande. Voir Migrer les données de messagerie par moyen d'exportation et d'importation pour obtenir plus d'informations.
- Vous pouvez migrer les données de la version précédente en configurant un pont JMS. Migrer les données de messagerie par un pont JMS.

Voir Mappage des noms de dossier de messagerie pour obtenir des détails sur les changements à apporter aux dossiers de données de messagerie entre les différentes versions.

#### 4.7.3.1. Migrer les données de messagerie par moyen d'exportation ou d'importation

Avec cette approche, vous exportez les données de la version précédente par l'utilitaire **`exporter`** d'HornetQ. Vous pouvez ensuite importer le fichier de sortie XML formaté par l'opération **`import-journal`**.

**Avertissement**

Il y a actuellement un problème connu avec cette approche. Pour vérifier l'état de cette question, voir JBEAP-4407 - Consumer crashes with IndexOutOfBoundsException when reading large messages from imported journal.

### Exportation des données de messagerie de la version précédente

**Important**

Le serveur JBoss EAP 6 doit être arrêté avant d'exporter les données de messagerie.

L'utilitaire **`exporter`** d'HornetQ génère et exporte les données de messagerie de JBoss EAP 6 vers un fichier au format XML. Cette commande nécessite que vous spécifiez les chemins vers les JAR HornetQ nécessaires fournis dans JBoss EAP 6, que vous communiquiez les chemins aux dossiers **`messagingbindings /`**, **`messagingjournal /`**, **`messagingpaging /`**, et **`messaginglargemessages /`** de la version précédente en tant qu'arguments, et que vous spécifiez un fichier de sortie dans lequel écrire les données XML exportées.

Voici la syntaxe requise par l'utilitaire **`exporter`** de HornetQ.

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

Créez un module personnalisé afin de s'assurer que les bonnes versions de JAR HornetQ, y compris les JAR installées avec des correctifs ou des mises à niveau, sont chargées et mises à la disposition de l'utilitaire **`exporter`**. À l'aide de votre éditeur favori, créez un nouveau fichier **`module.xml`** dans le répertoire **`EAP6_HOME/modules/org/hornetq/exportateur/main/`** et copier le contenu suivant :

```
<?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>
```

**Note**

Le module personnalisé est créé dans le répertoire **`modules/`**, et non pas dans le répertoire **`modules/system/layers/base/`**.

Suivre les étapes ci-dessous pour exporter les données.

1. Stopper le serveur JBoss EAP 6.
2. Créer le module personnalisé comme décrit ci-dessus.
3. Exécutez la commande suivante pour exporter les données.

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

4. Veillez à ce qu'il n'y ait aucun message d'erreur ou d'avertissement dans le fichier journal une fois que la commande est complétée.
5. Utiliser les outils disponibles dans votre système d'exploitation pour valider l'XML dans le fichier de sorties généré.

**Importer les données de messagerie formatées en XML**

Vous pouvez ensuite importer le fichier de sorties formatées XML par l'opération **`import-journal`**, comme suit.

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

### 4.7.3.2. Migrer les données de messagerie par un pontage JMS

En utilisant cette approche, vous configurez et déployez un pont JMS vers le serveur JBoss EAP 7 qui déplace les messages de la file d'attente de JBoss EAP 6 HornetQ à la file d'attente de JBoss EAP 7 ActiveMQ Artemis.

Un pontage JMS consomme des messages d'une file d'attente ou d'un topic JMS source et les envoie à une file d'attente JMS de cible ou sujet, se trouvant en général sur un autre serveur. Il peut être utilisé pour faire un pontage entre les messages et les serveurs JMS, tant qu'ils sont compatibles avec JMS 1.1. Les ressources JMS de source et de destination sont recherchées à l'aide de JNDI et les classes de client doivent être regroupées dans un module pour la recherche JNDI. Le nom du module est ensuite déclaré dans la configuration de pontage JMS.

Cette section décrit comment configurer les serveurs et déployer un pont JMS pour déplacer les données de messagerie de JBoss EAP 6 à JBoss EAP 7.

**Configurer un serveur JBoss EAP 6**

1. Stopper le serveur JBoss EAP 6.
2. Sauvegardez les fichiers de configuration et de journalisation d'HornetQ.
   - Par défaut, le journal HornetQ se trouve dans le répertoire **`EAP6_HOME/standalone/data/`**.
   - Voir Mappage des noms de dossiers de messagerie pour trouver les emplacements de dossiers de messagerie par défaut de chaque version.

3. Veillez à ce que la file d'attente **`InQueue`** JMS contenant les messages JMS soit bien définie dans le serveur JBoss EAP 6.

4. Veillez à ce que la configuration du sous-système de **`messaging`** contienne une entrée pour la **`RemoteConnectionFactory`** qui ressemble à ce qui suit.

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

Si elle ne contient pas cette entrée, créez-en une par la commande suivante :

```
/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)
```

### Configurer le serveur JBoss EAP 7

1. La configuration de pont JMS a besoin du module **`org.hornetq`** pour se connecter au serveur HornetQ de la version précédente. Ce module et ses dépendances directes ne sont pas présents dans JBoss EAP 7, donc vous devez copier les modules suivants de la version précédente.

   - Copier le module **`org.hornetq`** dans le répertoire JBoss EAP 7 **`EAP_HOME/modules/org/`**.

     - Si vous n'avez pas appliqué de correctif à ce module, copiez ce dossier à partir du serveur JBoss EAP 6.4 : **`EAP6_HOME/modules/system/layers/base/org/hornetq/`**

     - Si vous avez appliqué des correctifs à ce module, copiez ce dossier à partir du serveur JBoss EAP 6.4 : **`EAP6_HOME/modules/system/layers/base/.overlays/layer-base-jboss-eap-6.4.x.CP/org/hornetq/`**

**Note**

Pour la version corrigée, vous devez supprimer la ligne suivante du fichier JBoss EAP 7 **`modules/system/layers/base/org/hornetq/main/module.xml`** :

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

   - Copier le module **`org.jboss.netty`** dans le répertoire JBoss EAP 7 **`EAP_HOME/modules/org/jboss`**.

- Si vous n'avez pas appliqué de correctifs à ce module, copiez ce dossier à partir du serveur JBoss EAP 6.4 : **`EAP6_HOME/modules/system/layers/base/org/jboss/netty`**
- Si vous avez appliqué des correctifs à ce module, copiez ce dossier à partir du serveur JBoss EAP 6.4 : **`EAP6_HOME/modules/system/layers/base/.overlays/layer-base-jboss-eap-6.4.x.CP/org/jboss/netty`**

2. Créer la file d'attente JMS pour contenir les messages reçus du serveur JBoss EAP 6. Voici un exemple de commande qui crée la file d'attente JMS **`MigratedMessagesQueue`** pour recevoir le message.

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

   Cela aura comme effet de créer la configuration **`jms-queue`** du serveur par défaut dans le sous-système **`messaging-activemq`** du serveur JBoss EAP 7.

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

3. Veillez à ce que le serveur du sous-système **`messaging-activemq`** par **`défaut`** contienne une configuration de **`InVmConnectionFactory`** **`connection-factory`** qui ressemble à ceci :

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

   Si elle ne contient pas cette entrée, créez-en une par la commande suivante :

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

4. Créer et déployer un pont JMS qui puisse lire les messages de la file JMS **`InQueue`** configurée dans le serveur JBoss EAP 6 et qui les transfère dans la file d'attente **`MigratedMessagesQueue`** configurée dans le serveur JBoss EAP 7 .

```
/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.jboss.naming.remote.client.
InitialContextFactory"),
("java.naming.provider.url"=>"remote://127.0.0.1:4447")],target-
destination=jms/queue/MigratedMessagesQueue,target-connection-
factory=java:/ConnectionFactory)
```

Cela aura pour effet de créer la configuration **`jms-bridge`** dans le sous-système **`messaging-activemq`** du serveur JBoss EAP 7.

```
<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.jboss.naming.remote.client.InitialContextFactory"/>
            <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. Si la sécurité est configurée pour JBoss EAP 6, vous devez également configurer l'élément de configuration **`<source >`** du pont JMS pour qu'il puisse inclure un **`contexte source`** qui spécifie le nom d'utilisateur et mot de passe à utiliser pour la recherche JNDI quand on crée la connexion.

**Migrer les données**

1. Vérifier que les informations que vous fournissez pour les configurations suivantes soient correctes.
   - Les noms de file d'attente et de sujets
   - Le java.naming.provider.url pour la recherche JNDI
2. Veillez à bien déployer la destination JMS cible dans le serveur JBoss EAP 7.
3. Démarrez à la fois les serveurs JBoss EAP 6 et JBoss EAP 7.

### 4.7.3.3. Mappage des noms de dossiers de messagerie

Le tableau suivant montre les noms de répertoires de messagerie utilisés dans les versions précédentes et dans la version actuelle de JBoss EAP. Les répertoires sont relatifs au répertoire **`jboss.server.data.dir`**, qui est définit par défaut dans **`EAP_HOME/standalone/data/`**, s'il n'est pas spécifié.

| Nom de répertoire de JBoss EAP 6 | Nom de répertoire de JBoss EAP 7 |
|---|---|
| messagingbindings/ | activemq/bindings/ |
| messagingjournal/ | activemq/journal/ |

| Nom de répertoire de JBoss EAP 6 | Nom de répertoire de JBoss EAP 7 |
|---|---|
| messaginglargemessages/ | activemq/largemessages/ |
| messagingpaging/ | activemq/paging/ |

**Note**

Les répertoires **`messaginglargemessages/`** et **`messagingpaging/`** risquent de ne pas être présents s'il n'y a pas de messages volumineux ou si la pagination est désactivée.

### 4.7.4. Migrer les destinations JMS

Dans les versions précédentes de JBoss EAP, les files d'attente des destinations JMS étaient configurées dans l'élément **`<jms-destinations>`** situé sous l'élément **`<hornetq-server>`** dans le sous-système de messagerie **`messaging`**.

```
<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>
```

Dans JBoss EAP 7, la file d'attente de destination JMS est configurée dans l'élément par défaut **`<server>`** du sous-système **`messaging-activemq`**.

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

### 4.7.5. Migrer les intercepteurs de messagerie

Les intercepteurs de messagerie ont beaucoup changé dans JBoss EAP 7, suite au remplacement d'HornetQ par ActiveMQ Artemis comme fournisseur de messagerie JMS.

Le sous-système de **`messagerie`** d'HorneQ inclus dans la dernière version de JBoss EAP exige que vous installiez les intercepteurs d'HornetQ en les ajoutant à un JAR et en modifiant ensuite le fichier **`module.xml`** d'HornetQ.

Le sous-système de **`messagerie-activemq`** inclus dans JBoss EAP 7 ne nécessite pas de modification de fichier **`module.xml`**. Les classes d'intercepteur d'utilisateur qui implémentent maintenant l'interface d'intercepteur Apache ActiveMQ Artemis Interceptor peuvent maintenant être chargées depuis n'importe quel module de serveur. Vous spécifiez le module à partir duquel l'intercepteur doit être chargé dans le sous-système du fichier de configuration du serveur, **`messagerie-activemq`**.

**Exemple de configuration d'intercepteur**

```
<subsystem xmlns="urn:jboss:domain:messaging-activemq:1.0">
  <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>
```

### 4.7.6. Remplacer la configuration de Netty Servlet

Dans JBoss EAP 6, vous pouvez configurer un moteur de servlet pour qu'il puisse travailler avec le transport du Netty Servlet. Comme ActiveMQ Artemis remplace HornetQ en tant que fournisseur de messagerie intégrée dans JBoss EAP 7, cette configuration n'est plus disponible. Vous devez remplacer la configuration de servlet pour utiliser les nouveaux connecteurs http de messagerie intégrés et les accepteurs de http à la place.

## 4.8. CHANGEMENTS DE CONFIGURATION DU SERVEUR ORB

Dans JBoss EAP 7, l'implémentation de JacORB a été remplacée par une branche en aval d'ORB d'OpenJDK.

Le module d'extension **`org.jboss.as.jacorb`** situé dans **`EAP_HOME/modules/system/layers/base/`** a été remplacé par le module d'extension **`org.wildfly.iiop-openjdk`**.

L'espace de noms de la configuration du sous-système **`urn:jboss:domain:jacorb:1.4`** dans le fichier de configuration du serveur a été remplacé par l'espace de noms **`urn:jboss:domain:iiop-openjdk:1.0`**.

Ci-dessous figure un exemple de la configuration par défaut du système **`jacorb`** dans 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>
```

Ci-dessous figure un exemple de la configuration par défaut du sous-système **`iiop-openjdk`** dans 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>
```

La nouvelle configuration du sous-système **`iiop-openjdk`** accepte uniquement un sous-ensemble des éléments et attributs hérités. Ci-dessous figure un exemple de configuration du sous-système **`jacorb`** dans la version précédente de JBoss EAP qui contient tous les éléments et les attributs valides :

```
<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>
```

Les attributs de l'élément ci-dessous ne sont plus pris en charge et doivent être supprimés.

**Tableau 4.4. Attributs à supprimer**

| Élément | Attributs non pris en charge |
| --- | --- |
| **<orb>** | client-timeout<br>max-managed-buf-size<br>max-server-connections<br>outbuf-cache-timeout<br>outbuf-size<br>connection retries<br>retry-interval<br>name<br>server-timeout |
| **<poa>** | queue-min<br>queue-max<br>pool-size<br>max-threads |

Les attributs suivants de **on/off** ne sont plus pris en charge et ne seront donc pas migrés lorsque vous exécutez l'opération **migrate** en ligne de commande. S'ils sont définis sur **on**, vous obtiendrez un avertissement de migration. D'autres attributs **on/off** qui ne sont pas mentionnés dans ce tableau, par exemple **<security support-ssl="on|off">**, sont toujours pris en charge et pourront donc être migrés. La seule différence est que leurs valeurs vont passer de **on/off** à **true/false**.

**Tableau 4.5. Attributs à désactiver ou à supprimer**

| Élément | Attributs à définir sur « Off » |
| --- | --- |
| **<orb>** | cache-poa-names<br>cache-typecodes<br>print-version<br>use-bom<br>use-imr |

| Élément | Attributs à définir sur « Off » |
| --- | --- |
| **<interop>** | (tout sauf **sun**)<br>- comet<br>- iona<br>- chunk-custom-rmi-valuetypes<br>- indirection-encoding-disable<br>- lax-boolean-encoding<br>- strict-check-on-tc-creation |
| **<poa/>** | - monitoring<br>- queue-wait |

## 4.9. MIGRER LA CONFIGURATION DU SOUS-SYSTÈME DE THREADS

La configuration de serveur de JBoss EAP 6 incluse dans le sous-système de **threads** qui était utilisée pour gérer les pools de threads à travers les divers sous-systèmes du serveur.

Le sous-système de **threads** n'est plus disponible dans JBoss EAP 7. À la place, chaque sous-système est chargé de gérer ses propres pools de threads.

Pour obtenir plus d'informations sur la façon de configurer les pools de threads du sous-système **infinispan**, consulter Configurer les pools de threads d'Infinispan dans le *Guide de configuration* de JBoss EAP.

Pour obtenir plus d'informations sur la façon de configurer les pools de threads du sous-système **jgroups**, voir Configurer les pools de threads de JGroups dans le *Guide de configuration* de JBoss EAP.

Dans JBoss EAP 6, vous avez configuré des pools de threads pour connecteurs et listeners du sous-système de **web** en référençant un **exécuteur** défini dans le sous-système **threads**. Dans JBoss EAP 7, vous pouvez maintenant configurer les pools de threads pour le sous-système **undertow** en référençant un **worker** défini dans le sous-système **e/s**. Pour plus d'informations, consultez Configurer le sous-système e/s dans le *Guide de configuration* de JBoss EAP.

Pour obtenir plus d'informations sur les changements apportés à la configuration du pool de threads dans le sous-système **distant** , voir Migrer la configuration du sous-système distant de ce guide, et Configurer le point de terminaison du *Guide de configuration* de JBoss EAP.

## 4.10. MIGRER LA CONFIGURATION DU SOUS-SYSTÈME DISTANT

Dans JBoss EAP 6, vous pouviez configurer le pool de threads du sous-système **distant** en définissant des attributs de **worker-***. Le pool de threads de workers n'est plus configuré dans le sous-système **distant** de JBoss EAP 7 et si vous tentez de modifier la configuration existante, vous verrez le message suivant.

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

Dans JBoss EAP 7, le pool de threads de workers est remplacé par une configuration de point de terminaison qui référence un **`worker`** défini dans le sous-système **`e/s`**.

Pour plus d'informations sur la façon de configurer les points de terminaison, voir Configurer les points de terminaison du *Guide de configuration* de JBoss EAP 7.

## 4.11. CHANGEMENTS DE CONFIGURATION DU SERVEUR WEBSOCKET

Pour utiliser WebSockets sur JBoss EAP 6, vous deviez activer le protocole du connecteur Java NIO2 non bloquant pour le connecteur **`http`** dans le sous-système **`web`** du fichier de configuration du serveur JBoss EAP en utilisant une commande similaire à la suivante.

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

Pour utiliser les WebSockets dans une application, vous deviez également créer un élément **`<enable-websockets>`** dans le fichier de l'application **`WEB-INF/jboss-web.xml`** et le définir sur **`true`**.

Dans JBoss EAP 7, il n'est plus nécessaire de configurer le serveur pour la prise en charge par défaut de WebSocket, ni de configurer l'appliquer pour qu'elle puisse l'utiliser . Les WebSockets sont une condition préalable de la spécification Java EE 7 et les protocoles requis sont configurés par défaut. Une configuration WebSocket plus complexe est effectuée dans le **`servlet-container`** du sous-système **`Undertow`** du fichier de configuration du serveur JBoss EAP. Vous pouvez afficher les paramètres disponibles en utilisant la commande suivante.

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

Des exemples de code WebSocket se trouvent également dans les guides de mise en route rapide (« quickstarts ») fournis avec JBoss EAP.

## 4.12. CHANGEMENTS DU SERVEUR « SINGLE SIGN-ON »

Le sous-système Infinispan fournit toujours une prise en charge de cache distribué pour les services HA sous la forme de caches Infinispan dans JBoss EAP 7 ; cependant, la mise en cache et la distribution des informations d'authentification sont gérées de manière différente par rapport aux versions précédentes.

Dans JBoss EAP 6, si SSO (« Single Sign-on ») n'a pas reçu de cache Infinispan ; le cache n'a pas été distribué.

Dans JBoss EAP 7, SSO est distribué automatiquement lorsque vous sélectionnez le profil HA. Lors de l'exécution du profil HA, chaque hôte possède son propre cache Infinispan, qui est basé sur le cache par défaut du conteneur du cache web. Ce cache stocke la session pertinente et les informations du cookie SSO pour l'hôte. JBoss EAP gère la propagation de chaque cache individuel vers tous les hôtes. Il n'y a pas de manière spécifique d'assigner un cache Infinispan à SSO dans JBoss EAP 7.

Dans JBoss EAP 7, SSO est configuré dans le sous-système **`Undertow`** du fichier de configuration du système.

Aucun changement du code d'application n'est requis pour SSO lors de la migration vers JBoss EAP 7.

## 4.13. CHANGEMENTS DANS LA CONFIGURATION DE LA SOURCE DE DONNÉES

### 4.13.1. Le nom du pilote de la source de données JBDC

Lorsque vous configuriez une source de données dans la version précédente de JBoss EAP, la valeur spécifiée de nom du pilote dépendait du nombre de classes répertoriées dans le fichier **`META-INF/services/java.sql.Driver`** contenu dans la JAR du pilote JDBC.

#### Pilote contenant une classe unique

Si le fichier **`META-INF/services/java.sql.Driver`** spécifiait une seule classe uniquement, la valeur du nom du pilote correspondait tout simplement au nom du JAR du pilote JDBC. Cela n'a pas été modifié dans JBoss EAP 7.

#### Pilote contenant plusieurs classes

Dans JBoss EAP 6, s'il y avait plus d'une classe listée dans le fichier **`META-INF/services/java.sql.Driver`**, vous pouviez spécifier laquelle était la classe de pilote en ajoutant son nom au nom du JAR, avec la version mineure ou majeure, dans le format suivant :

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

Dans JBoss EAP7, cela a changé. Vous devez maintenant spécifier le nom du pilote en utilisant le format suivant :

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

**Note**

Un trait de soulignement a été ajouté entre le JAR_NAME et le DRIVER_CLASS_NAME.

Le pilote JDBC MySQL 5.1.31 est un exemple de pilote qui contient deux classes. Le nom de classe du pilote est **`com.mysql.jdbc.Driver`**. Les exemples suivants illustrent les différences entre la manière dont vous spécifiez un nom du pilote dans la version précédente de JBoss EAP et celle dont vous le spécifiez dans la version actuelle.

### Exemple de nom de pilote JBoss EAP 6

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

### Exemple de nom de pilote JBoss EAP 7

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

## 4.14. CHANGEMENTS DE CONFIGURATION DU SERVEUR DE SÉCURITÉ

Pour obtenir des informations sur les changements de configuration au serveur du Gestionnaire de sécurité Java, voir Considérations quand on passe d'une ancienne à une nouvelle version dans *Comment configurer la sécurité dans les serveurs* de JBoss EAP.

## 4.15. MODIFICATIONS APPORTÉES AU SOUS-SYSTÈME DE TRANSACTIONS

Certains attributs de configuration du Gestionnaire de transactions, qui étaient disponibles dans le sous-système **`transactions`** de JBoss EAP 6 ont changé dans JBoss EAP 7.

### Attributs du sous-système de transactions supprimés

Le tableau suivant répertorie les attributs de JBoss EAP 6 qui ont été retirés du sous-système **`transactions`** dans JBoss EAP 7 et les attributs de remplacement équivalents.

| Attribut de JBoss EAP 6 | Remplacement dans JBoss EAP 7 |
|---|---|
| path | object-store-path |
| relative-to | object-store-relative-to |

### Attributs de sous-systèmes de transactions dépréciés

Les attributs suivants qui étaient disponibles dans le sous-système de **`transactions`** de JBoss EAP 6 sont dépréciés dans JBoss EAP 7. Les attributs dépréciés risquent d'être supprimés dans une version ultérieure du produit. Le tableau suivant répertorie les attributs de remplacement équivalents.

| Attribut de JBoss EAP 6 | Remplacement dans JBoss EAP 7 |
|---|---|
| use-hornetq-store | use-journal-store |
| hornetq-store-enable-async-io-to | journal-store-enable-async-io |
| enable-statistics | statistics-enabled |

## 4.16. CHANGEMENTS À LA CONFIGURATION DE MOD_CLUSTER

La configuration des listes de proxy statiques de mod_cluster a été modifiée dans JBoss EAP 7.

Dans JBoss EAP 6, vous configuriez l'attribut **`proxy-list`**, qui correspondait à une liste des adresses de proxy httpd séparées par des virgules, sous le format **`hostname:port`**.

L'attribut **`proxy-list`** est déprécié dans JBoss EAP 7. Il a été remplacé par l'attribut **`proxies`** , qui correspond à une liste de noms de liaisons de sockets de sortie.

Ce changement a un impact sur la façon de définir une liste de proxys statiques, par exemple, quand on désactive les annonces dans mod_cluster. Pour obtenir des informations sur la façon de désactiver les annonces dans mod_cluster, voir Désactiver les annonces dans mod_cluster dans le *Guide de configuration* de JBoss EAP.

Pour obtenir des informations sur les attributs de mod_cluster, voir mod_cluster Subsystem Attributes dans le *Guide de configuration* de JBoss EAP.

# CHAPITRE 5. CHANGEMENTS APPORTÉS À LA MIGRATION DES APPLICATIONS

## 5.1. CHANGEMENTS APPORTÉS AUX APPLICATIONS DE SERVICES WEB

JBossWS 5 apporte de nouvelles fonctionnalités et améliorations des performances aux services web JBoss EAP 7, principalement à travers la mise à niveau des composants Apache CXF, Apache WSS4J, et Apache Santuario.

### 5.1.1. Changements de prise en charge JAX-RPC

L'API Java pour RPC basée XML (JAX-RPC) a été déconseillée pour Java EE 6 et est optionnelle sur Java EE 7. Cette API n'est plus disponible ou prise en charge sur JBoss EAP 7. Les applications qui utilisent JAX-RPC doivent être migrées pour utiliser JAX-WS, qui est le framework actuel des services web standard Java EE.

L'utilisation des services JAX-RPC peut être identifiée de l'une des manières suivantes :

- La présence d'un fichier de mappage JAX-RPC, qui est un fichier XML avec l'élément root **`<java-wsdl-mapping>`**.
- La présence d'un fichier descripteur XML **`webservices.xml`** qui contient un élément **`<webservice-description>`** incluant un élément enfant **`<jaxrpc-mapping-file>`**. Ci-dessous figure un exemple de fichier descripteur **`webservices.xml`** qui définit un service 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">
  <webservice-description>
    <webservice-description-name>HelloService</webservice-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>
  </webservice-description>
</webservices>
```

- La présence d'un fichier **`ejb-jar.xml`**, qui contient un **`<service-ref>`** référençant un fichier de mappage JAX-RPC.

### 5.1.2. Changements aux services web Apache CXF Spring

Dans les versions précédentes de JBoss EAP, vous pouviez personnaliser l'intégration JBossWS et Apache CXF en incluant un fichier de configuration **`jbossws-cxf.xml`** avec l'archive du déploiement du point de terminaison (« endpoint deployment archive »). Un exemple de cas d'utilisation consiste à configurer des threads d'intercepteurs pour les points de terminaison du serveur et du client du service web sur le bus Apache CXF. Cette intégration nécessitait que Spring soit déployé dans le serveur JBoss EAP.

L'intégration Spring n'est plus prise en charge dans JBoss EAP 7. Toute application qui contient un fichier de configuration de descripteur **`jbossws-cxf.xml`** doit être modifié pour remplacer la configuration personnalisée définie dans ce fichier. Alors qu'il est toujours possible d'accéder directement à l'API Apache CXF, veuillez prendre en compte le fait que l'application ne sera pas portable.

L'approche suggérée consiste à remplacer les configurations personnalisées Spring par les nouvelles options de configuration du descripteur JBossWS lorsque c'est possible. L'approche basée descripteur JBossWS fournit une fonctionnalité similaire sans nécessiter de modifications du code du point de terminaison du client. Dans certains cas, vous pouvez remplacer Spring par CDI (« Context Dependency Injection »).

#### Intercepteurs Apache CXF

Le descripteur JBossWS fournit de nouvelles options de configuration vous permettant de déclarer les intercepteurs sans modifier le code du point de terminaison du client. Au lieu de cela, vous pouvez déclarer les intercepteurs dans des configurations de clients et de points de terminaison prédéfinis en spécifiant une liste de noms de classes d'intercepteurs pour les propriétés **`cxf.interceptors.in`** et **`cxf.interceptors.out`**.

Ci-dessous figure un exemple de fichier **`jaxws-endpoint-config.xml`** qui déclare des intercepteurs en utilisant ces propriétés.

```
<?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.jb
oss.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>
  </endpoint-config>
</jaxws-config>
```

### Fonctionnalités Apache CXF

Le descripteur JBossWS vous permet de déclarer des fonctionnalités dans des configurations de clients et de points de terminaison prédéfinis en spécifiant une liste de noms de classes de fonctionnalités pour la propriété **`cxf.features`**.

Ci-dessous figure un exemple de fichier **`jaxws-endpoint-config.xml`** qui déclare une fonctionnalité en utilisant cette propriété.

```
<?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>
```

### Transport HTTP Apache CXF

Dans Apache CXF, la configuration du transport HTTP est effectuée en spécifiant les options **`org.apache.cxf.transport.http.HTTPConduit`**. L'intégration JBossWS permet aux conduits d'être modifiés par moyen de programme en utilisant l'API Apache CXF comme suit.

```
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);
...
```

Vous pouvez également contrôler et outrepasser les valeurs par défaut Apache CXF **`HTTPConduit`** en paramétrant les propriétés système.

| Propriété | Type | Description |
|---|---|---|
| cxf.client.allowChunking | Boolean | Spécifie s'il faut envoyer des requêtes en utilisant la segmentation. |

| Propriété | Type | Description |
|---|---|---|
| cxf.client.chunkingThreshold | Integer | Définit la limite à laquelle basculer du mode sans segmentation au mode de segmentation. |
| cxf.client.connectionTimeout | Long | Définit le nombre de millisecondes avant l'expiration de la connexion. |
| cxf.client.receiveTimeout | Long | Définit le nombre de millisecondes avant l'expiration de la réception. |
| cxf.client.connection | String | Spécifie s'il faut utiliser le type de connexion **`Keep-Alive`** ou **`close`**. |
| cxf.tls-client.disableCNCheck | Boolean | Spécifie s'il faut désactiver la vérification de nom d'hôte CN. |

### 5.1.3. Changements WS-Security

- Si votre application contient un gestionnaire de rappels (« callback ») personnalisé qui accède à la classe **`org.apache.ws.security.WSPasswordCallback`**, prenez en compte que cette classe a été déplacée sur le paquet **`org.apache.wss4j.common.ext`**.
- La plupart des objets bean SAML du paquet **`org.apache.ws.security.saml.ext`** ont été déplacés sur **`org.apache.wss4j.common.saml package`**.
- Veuillez utiliser le transport de clé RSA v1.5 et tous les algorithmes connexes seront rejetés par défaut.
- Auparavant, le service de jetons de sécurité (STS, ou « Security Token Service ») ne validait que les jetons **`onBehalfOf`**. Désormais, il valide également les jetons **`ActAs`**. Par conséquent, un nom d'utilisateur et un mot de passe valides doivent être spécifiés dans le **`UsernameToken`** fourni pour le jeton **`ActAs`**.
- Les jetons Bearer SAML doivent désormais avoir une signature interne. La classe **`org.apache.wss4j.dom.validate.SamlAssertionValidator`** contient maintenant une méthode **`setRequireBearerSignature()`** pour activer ou désactiver la vérification de signatures.

### 5.1.4. Changements dans la structure des modules JBoss

Les JAR **`cxf-api`** et **`cxf-rt-core`** ont été fusionnés en une seule JAR **`cxf-core`**. Par conséquent, le module **`org.apache.cxf`** de JBoss EAP contient désormais la JAR **`cxf-core`** et expose davantage de classes par rapport à la version précédente.

### 5.1.5. Conditions préalable et changements de Bouncy Castle

Si vous souhaitez utiliser le chiffrement AES avec GCM (« Galois/Counter Mode ») pour le chiffrement symétrique en XML/WS-Security, vous aurez besoin du fournisseur de sécurité BouncyCastle.

JBoss EAP 7 est fourni avec le module **`org.bouncycastle`** et JBossWS est désormais capable de reposer sur son chargeur de classes pour obtenir et utiliser le fournisseur de sécurité BouncyCastle. Ainsi, il n'est plus nécessaire d'installer BouncyCastle statiquement dans la JVM actuelle. Pour les applications exécutées hors du conteneur, le fournisseur de sécurité peut être mis à disponibilité de JBossWS en ajoutant une bibliothèque BouncyCastle au chemin de classe.

Vous pouvez désactiver ce comportement en paramétrant la valeur de la propriété **`org.jboss.ws.cxf.noLocalBC`** sur **`true`** dans le fichier du descripteur de déploiement **`jaxws-endpoint-config.xml`** pour le serveur ou dans le fichier du descripteur **`jaxws-client-config.xml`** pour les clients.

Si vous souhaitez utiliser une version différente de celle fournie dans JBoss EAP, vous pouvez installer statiquement BouncyCastle sur la JVM. Dans ce cas, le fournisseur de sécurité installé statiquement BouncyCastle sera choisi plutôt à la place du fournisseur présent dans le classpath. Pour éviter tout problème, vous devez utiliser BouncyCastle 1.49, 1.51, ou une version supérieure.

### 5.1.6. Stratégie de sélection de Bus Apache CXF

La stratégie de sélection de bus par défaut pour des clients exécutés dans des conteneurs a été modifiée de **`THREAD_BUS`** à **`TCCL_BUS`**. Pour les clients exécutés hors conteneurs, la stratégie par défaut reste **`THREAD_BUS`**. Vous pouvez restaurer le comportement de la version précédente en utilisant l'une de méthodes suivantes.

- Lancez le serveur JBoss EAP avec la valeur de la propriété système **`org.jboss.ws.cxf.jaxws-client.bus.strategy`** définie sur **`THREAD_BUS`**.
- Définissez explicitement la stratégie de sélection dans le code client.

### 5.1.7. Conditions préalables JAX-WS 2.2 pour WebServiceRef

Les conteneurs doivent utiliser des constructeurs de style JAX-WS 2.2, en utilisant la classe WebServiceFeature, pour créer des clients injectés dans les références du service web. Cela signifie que les classes de services fournies par l'utilisateur et injectées par le conteneur doivent implémenter JAX-WS 2.2 ou une version supérieure.

### 5.1.8. Changements de vérification CN IgnoreHttpsHost

Dans les versions précédentes, vous pouviez désactiver la vérification du nom d'hôte de l'URL HTTPS avec le nom commun (« Common Name », ou CN) donné dans son certificat en paramétrant la propriété système **`org.jboss.security.ignoreHttpsHost`** sur **`true`**. Ce nom de propriété système a été remplacé par **`cxf.tls-client.disableCNCheck`**.

### 5.1.9. Configuration côté client et chargement de classe

En conséquence de l'activation des injections dans les points de terminaison de services et les gestionnaires de clients de services, il n'est plus possible de charger automatiquement les classes de gestionnaires du module JBoss **`org.jboss.as.webservices.server.integration`**. Si votre application dépend d'une configuration prédéfinie, vous devrez définir de nouvelles dépendances de modules pour votre déploiement de manière explicite. Pour obtenir davantage d'informations, veuillez consulter Migrer les dépendances de modules explicites

### 5.1.10. Le mécanisme de remplacement des standards approuvés Java est déconseillé

Le Mécanisme de remplacement des standards approuvés Java a été déconseillé sur JDK 1.8_40 dans le but de le supprimer dans JDK 9. Ce mécanisme permettait aux développeurs de rendre des bibliothèques disponibles à toutes les applications déployées en plaçant les JAR dans un répertoire approuvé dans JRE.

Si votre application utilise l'implémentation JBossWS d'Apache CXF, JBoss EAP 7 s'assure que les dépendances requises soient ajoutées dans l'ordre correct et que vous ne soyez pas affecté par ce changement. Si votre application accède directement à Apache CXF, vous devrez désormais fournir les dépendances Apache CXF après avoir fourni les dépendances JBossWS, car cela fait partie de votre déploiement d'application.

### 5.1.11. Spécification du descripteur dans l'archive EAR

Dans les versions précédentes de JBoss EAP, vous pourriez configurer le fichier descripteur de déploiement **`jboss-webservices.xml`** pour les déploiements de service web EJB dans le répertoire **`META-INF/`** des archives JAR ou dans le répertoire **`WEB-INF/`** pour les déploiements de services web POJO et les points de terminaison de services web EJB regroupés en archives WAR.

Avec JBoss EAP 7, vous pouvez désormais configurer le fichier du descripteur de déploiements **`jboss-webservices.xml`** dans le répertoire **`META-INF/`** d'une archive EAR. Si un fichier **`jboss-webservices.xml`** est trouvé dans l'archive EAR et dans l'archive JAR ou WAR, les données de configuration du fichier **`jboss-webservices.xml`** JAR ou WAR remplaceront les données correspondantes du fichier descripteur EAR.

## 5.2. METTEZ À JOUR LE PORT ET LE CONNECTEUR D'URL DISTANTS

Dans JBoss EAP 7, le connecteur par défaut a changé de **`remote`** à **`http-remoting`** et le port de connexion à distance par défaut est passé de **`4447`** à **`8080`**. L'URL du fournisseur JNDI pour la configuration par défaut est passé de **`remote://localhost:4447`** à **`http-remoting://localhost:8080`**.

Si vous utilisez l'opération **`migrate`** de JBoss EAP 7 pour mettre à jour votre configuration, vous n'aurez pas besoin de modifier le connecteur distant, ni le port distant ou l'URL de fournisseur JNDI parce que l'opération de migration préservera les paramètres de configuration du connecteur **`distant`** et du port**`4447`** dans la configuration du sous-système. Pour plus d'informations sur l'opération **`migrate`**, voir la Gestion des opération de migration par lignes de commande.

Si vous n'utilisez pas l'opération de **`migrate`**, et qu'à la place, vous exécutez avec la nouvelle configuration par défaut de JBoss EAP 7, vous devez modifier le connecteur distant, le port distant et l'URL de fournisseur JNDI pour utiliser les nouveaux paramètres, comme décrit ci-dessus.

## 5.3. CHANGEMENTS DANS L'APPLICATION DE MESSAGERIE

### 5.3.1. Remplacer ou mettre à jour des descripteurs de déploiement JMS

Les fichiers de descripteurs de déploiement des ressources JMS propriétaires identifiées par le schéma de dénomination **`-jms.xml`** sont déconseillés dans JBoss EAP 7. Vous verrez ci-dessous un exemple de fichier descripteur de déploiement de ressources JMS dans 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>
```

Si vous avez utilisé des descripteurs JMS **`-jms.xml`** dans votre application dans la version précédente, vous pouvez convertir votre application pour utiliser le descripteur de déploiement Java EE 7 standard comme indiqué dans la section EE.5.18 de la spécification Java EE 7, ou vous pouvez mettre à jour le descripteur de déploiement pour utiliser le sous-système **`messaging-activemq`** dans JBoss EAP 7.

Si vous choisissez de mettre à jour le descripteur, vous devrez effectuer les modifications suivantes.

- Remplacez l'espace de noms « urn:jboss:messaging-deployment:1.0 » par « urn:jboss:messaging-activemq-deployment:1.0 ».
- Remplacez le nom de l'élément **`<hornetq-server>`** par **`<server>`**.

Le fichier modifié devrait ressembler à l'exemple suivant.

```
<?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>
```

Pour obtenir des informations sur les changements de la configuration du serveur liés à la messagerie, veuillez consulter Changements de la configuration du serveur de messagerie.

### 5.3.2. Mise à jour des clients JMS externes

JBoss EAP 7 continue de prendre en charge l'API JMS 1.1. Ainsi, il n'est pas nécessaire de modifier votre code.

Le connecteur distant et le port distant par défaut ont été modifiés dans JBoss EAP 7. Pour obtenir plus d'informations sur ce changement, voir Mise à jour du port et du connecteur d'URL distants.

Si vous migrez votre configuration de serveur à l'aide de l'opération **`migrate`**, les anciens paramètres seront conservés et vous n'aurez pas besoin de mettre à jour votre **`PROVIDER_URL`**. Toutefois, si vous exécutez avec la nouvelle configuration par défaut de JBoss EAP 7, vous devrez modifier le **`PROVIDER_URL`** dans le code client pour utiliser le nouveau paramètre de configuration **`http-remoting://localhost:8080`**. Pour plus d'informations, consultez Migrer la désignation de code client à distance.

Si vous planifiez de migrer votre code pour utiliser l'API JMS 2.0, veuillez voir le quickstart **`helloworld-jms`** pour un exemple.

### 5.3.3. Remplacement de l'API HornetQ

JBoss EAP 6 comprenait le module **`org.hornetq`**, qui vous permettait d'utiliser l'API HornetQ dans le code source de votre application.

Apache ActiveMQ Artemis remplace HornetQ dans JBoss EAP 7, donc vous devez migrer tout code qui a utilisé l'API HornetQ pour pouvoir utiliser l' API de Artemis Apache ActiveMQ. Les bibliothèques de cette API se trouvent dans le module **`org.apache.activemq.artemis`**.

ActiveMQ Artemis est une évolution d'HornetQ, donc de nombreux concepts s'appliquent toujours.

## 5.4. CHANGEMENTS DANS LES APPLICATIONSJAX-RS ET RESTEASY

La version précédente de JBoss EAP fournissait RESTEasy 2, qui est une implémentation de JAX-RS 1.x. JBoss EAP 7 comprend maintenant 3 RESTEasy, qui est une implémentation de JAX-RS 2.0 définie par la spécification JSR 339 : JAX-RS 2.0 : The Java API for RESTful Web Services. Pour plus d'informations sur l'API de Java des Services Web RESTful, consultez la Spécification de l'API JAX-RS 2.0.

La version de Jackson comprise dans JBoss EAP a changé. L'ancienne version de JBoss EAP comprenait Jackson 1.9.9. JBoss EAP 7 inclut maintenant Jackson 2.6.3 ou version supérieure.

Cette section décrit comment ces changements peuvent avoir un impact sur les applications qui utilisent RESTEasy ou JAX-RS.

### 5.4.1. Classes dépréciées de RESTEasy

#### Classes de corps de texte ou d'intercepteurs

JSR 311 : JAX-RS: l'API Java™ des Services Web RESTful ne comprenait pas de framework d'intercepteur, donc RESTEasy 2 en a fourni un. JSR 339 : JAX-RS 2.0 : l'API Java des Services Web RESTful a introduit un framework d'intercepteur et de filtre officiels, donc le framework de l'intercepteur inclus dans RESTEasy 2 est maintenant obsolète et est remplacé par l'intercepteur conforme JAX-RS 2.0 dans RESTEasy 3.x.. Les interfaces pertinentes sont définies dans le paquet **`javax.ws.rs.ext`** du module **`jaxrs-api`**.

- Les interfaces d'intercepteur suivantes sont dépréciées dans 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**

- L'interface **org.jboss.resteasy.spi.interception.PreProcessInterceptor** est remplacée par l'interface **javax.ws.rs.container.ContainerRequestFilter** de RESTEasy 3.x.
- Les classes et interfaces suivantes sont dépréciées dans RESTEasy 3.x.
  - **org.jboss.resteasy.spi.interception.MessageBodyReaderInterceptor**
  - **org.jboss.resteasy.spi.interception.MessageBodyWriterInterceptor**
  - **org.jboss.resteasy.spi.interception.MessageBodyWriterContext**
  - **org.jboss.resteasy.spi.interception.MessageBodyReaderContext**
  - **org.jboss.resteasy.core.interception.InterceptorRegistry**
  - **org.jboss.resteasy.core.interception.InterceptorRegistryListener**
  - **org.jboss.resteasy.core.interception.ClientExecutionContextImpl**
- L'interface **org.jboss.resteasy.spi.interception.MessageBodyWriterInterceptor** est remplacée par l'interface **javax.ws.rs.ext.WriterInterceptor**.
- Certains changements apportés à l'interface **javax.ws.rs.ext.MessageBodyWriter** risquent de ne pas être rétro-compatibles quant à JAX-RS 1.x. Si votre application utilisait JAX-RS 1.x, examinez votre code d'application pour vous assurer de définir **@Produces** ou **@Consumes** pour vos points de terminaison. Manquer à cela pourrait résulter avec une erreur similaire à la suivante.

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

Voici un exemple de point de terminaison de REST qui peut entraîner une erreur.

```
@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.getLocalizedMess
age()).type(MediaType.TEXT_PLAIN)
                    .build());
        }
        return days;
    }
}
```

Pour régler ce problème, ajouter l'importation de **`javax.ws.rs.Produces`** et de l'annotation **`@Produces`** comme suit.

```
...
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.getLocalizedMess
age()).type(MediaType.TEXT_PLAIN)
                    .build());
        }
        return days;
    }
}
```

**Note**

Tous les intercepteurs de l'ancienne version de RESTasy peuvent exécuter en parrallèle avec le nouveau filtre JAX-RS 2.0 et les interfaces de l'intercepteur.

Pour obtenir plus d'informations sur les intercepteurs, voir Intercepteurs RESTEasy dans *Développer des applications de services Web* dans JBoss EAP.

Pour obtenir plus d'informations sur la nouvelle API de remplacement, voir RESTEasy JAX-RS 3.0.13.Final API ou version supérieure.

### API Client

Le framework du client RESTEasy de **`resteasy-jaxrs`** a été remplacé par le module **`resteasy-client`** conforme à JAX-RS 2.0. De ce fait, certaines classes et méthodes de l'API du clientRESTEasy sont dépréciées.

- Les classes suivantes sont dépréciées.
  - **`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`**
- L'exception **`org.jboss.resteasy.client.ClientResponseFailure`** et les interfaces **`org.jboss.resteasy.client.ClientExecutor`** et **`org.jboss.resteasy.client.EntityTypeFactory`** sont également dépréciées.
- Vous devez remplacer les classes **`org.jboss.resteasy.client.ClientRequest`** et **`org.jboss.resteasy.client.ClientResponse`** par **`org.jboss.resteasy.client.jaxrs.ResteasyClient`** et **`javax.ws.rs.core.Response`** respectivement.

  Voici un exemple de la façon d'envoyer un en-tête de lien avec le client RESTEasy dans RESTEasy 2.3x.

  ```
  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();
  ```

  Voici un exemple sur la façon d'accomplir la même tâche avec le client RESTEasy dans 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");
  ```

  Voir le quickstart **`resteasy-jaxrs-client`** pour obtenir un exemple de clientJAX-RS RESTEasy qui interagit avec un service JAX-RS Web.
- Les classes et interfaces du paquet **`org.jboss.resteasy.client.cache`** sont également dépréciées. Elles sont remplacées par les classes et interfaces équivalentes qui se trouvent dans le paquet **`org.jboss.resteasy.client.jaxrs.cache`**.

**Note**

Pour obtenir davantage d'informations sur les classes d'API **`org.jboss.resteasy.client.jaxrs`**, veuillez consulter le RESTEasy JAX-RS JavaDoc.

#### StringConverter

La classe **`org.jboss.resteasy.spi.StringConverter`** est obsolète dans RESTEasy 3.x. Cette fonctionnalité peut être remplacée par la classe JAX-RS 2.0 jax.ws.rs.ext.ParamConverterProvider.

### 5.4.2. Classes de RESTEasy supprimées ou protégées

#### Méthodes d'ajout de ResteasyProviderFactory

La plupart des méthodes **`org.jboss.resteasy.spi.ResteasyProviderFactory`** **`add()`** ont été supprimées ou sont maintenant protégées dans RESTEasy 3.0. Ainsi, les méthodes **`addBuiltInMessageBodyReader()`** et **`addBuiltInMessageBodyWriter()`** ont été supprimées et les méthodes **`addMessageBodyReader()`** et **`addMessageBodyWriter()`** sont maintenant protégées.

Vous devriez utiliser les méthodes **`registerProvider()`** et **`registerProviderInstance()`**.

#### Classes supplémentaires supprimées de RESTEasy 3

L'annotation **`@org.jboss.resteasy.annotations.cache.ServerCached`**, qui spécifiait que la réponse à la méthode JAX-RS devait être mise en cache sur le serveur, a été supprimée de RESTEasy 3 et doit être supprimée du code d'application.

### 5.4.3. Autres changements dans RESTEasy

#### SignedInput et SignedOuput

- **`SignedInput`** et **`SignedOutput`** de **`resteasy-crypto`** doivent avoir le **`Content-Type`** défini sur **`multipart/signed`** dans l'objet **`Request`** ou **`Response`**, ou en utilisant les annotations **`@Consumes`** ou **`@Produces`**.
- **`SignedOutput`** et **`SignedInput`** peuvent être utilisés pour transformer le format MIME de la **`application/pkcs7-signature`** en forme binaire, en définissant ce type dans les annotations **`@Produces`** ou **`@Consumes`**.
- Si **`@Produces`** ou **`@Consumes`** sont de type MIME **`text/plain`**, **`SignedOutput`** sera encodé en base64 et envoyé en tant que chaîne.

#### Filtres de sécurité

Les filtres de sécurité **`@RolesAllowed`**, **`@PermitAll`**, et **`@DenyAll`** retournent maintenant l'erreur "403 Forbidden" à la place de "401 Unauthorized".

#### Filtres côté client

Les nouveaux filtres JAX-RS 2.0 côté client ne seront pas reliés et exécuteront quand vous utilisez l'API de client RESTEasy d'une ancienne version.

#### Support HTTP asynchrone

Parce que la spécification JAX-RS 2.0 ajoute la prise en charge HTTP asynchrone par l'annotation **@Suspended** et l'interface **AsynResponse**, l'API RESTEasy propriétaire d'HTTP asynchrone a été dépréciée et risque d'être supprimée dès RESTEasy 3.1. Le Tomcat asynchrone et les modules de JBoss Web asynchrones ont également été supprimés de l'installation du serveur. Si vous n'utilisez pas le conteneur de Servlet 3.0 ou ultérieur, le traitement HTTP asynchrone côté serveur HTTP sera simulé et exécuté en simultané dans le même thread de demande.

#### Cache côté serveur

La configuration du cache côté serveur a changé. Veuillez consulter RESTEasy Documentation pour plus d'informations.

### 5.4.4. Changements apportés à RESTEasy

#### Exceptions SPI

Toutes les exceptions d'échecs SPI ont été dépréciées et ne sont plus utilisées en interne. Elles ont été remplacées par l'exception JAX-RS 2.0 correspondante.

| Exception dépréciée | Exception de remplacement dans le module jaxrs-api |
|---|---|
| org.jboss.resteasy.spi.ForbiddenException | javax.ws.rs.ForbiddenException |
| org.jboss.resteasy.spi.MethodNotAllowedException | javax.ws.rs.NotAllowedException |
| org.jboss.resteasy.spi.NotAcceptableException | javax.ws.rs.NotAcceptableException |
| org.jboss.resteasy.spi.NotFoundException | javax.ws.rs.NotFoundException |
| org.jboss.resteasy.spi.UnauthorizedException | javax.ws.rs.NotAuthorizedException |
| org.jboss.resteasy.spi.UnsupportedMediaTypeException | javax.ws.rs.NotSupportedException |

#### InjectorFactory et Registre

Les SPI **`InjectorFactory`** et **`Registry`** ont changé. Cela ne devrait pas être un problème si vous utilisez le RESTEasy documenté et pris en charge.

### 5.4.5. Changements dans le fournisseur Jackson

La version de Jackson comprise dans JBoss EAP a changé. L'ancienne version de JBoss EAP comprenait Jackson 1.9.9. JBoss EAP 7 inclut maintenant Jackson 2.6.3 ou version supérieure. De ce fait, le fournisseur Jackson est passé de **`resteasy-jackson-provider`** à **`resteasy-jackson2-provider`**.

La mise à niveau à **`resteasy-jackson2-provider`** requiert certains changements de paquets. Par exemple, le package d'annotations Jackson est passé de **`org.codehaus.jackson.annotate`** à **`com.fasterxml.jackson.annotation`**.

Pour que votre application puisse utiliser le fournisseur par défaut qui était inclus dans la dernière version de JBoss EAP, voir Activer le fournisseur Jackson par défaut dans *Développer des applications de services web* de JBoss EAP.

### 5.4.6. Changements à l'intégration de Spring RESTEasy

Le framework de Spring 4.0 introduisait un support pour Java 8. Si vous avez l'intention d'utiliser l'intégration RESTEasy 3.x avec Spring, veillez à spécifier 4.2.x comme version Spring minimum dans votre déploiement car c'est l'ancienne version la plus stable supportée par JBoss EAP 7.

### 5.4.7. Changements au fournisseur Jettison RESTEasy

Le fournisseur Jettison RESTEasy est déprécié dans JBoss EAP 7 et n'est plus ajouté aux déploiements par défaut. Nous vous encourageons à passer au fournisseur Jackson RESTEasy recommandé. Si vous préférez continuer à utiliser le fournisseur Jettison, vous devez lui définir une dépendance explicite dans le fichier **`jboss-deployment-descriptor.xml`** comme illustré dans l'exemple suivant.

```
<?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>
```

Pour plus d'informations sur la façon de définir les dépendances explicites, voir Ajouter une dépendance de module explicite à un déploiement dans le *Guide de développement* de JBoss EAP.

## 5.5. CHANGEMENTS À L'APPLICATION CDI 1.2

JBoss EAP 7 inclut un support à CDI 1.2. De ce fait, les applications écrites avec CDI 1.0 risquent de voir des changements de comportement quand elles sont migrées dans JBoss EAP 7. Cette section résume quelques uns de ces changements uniquement.

Vous allez trouver des informations sur Weld et CDI 1.2 dans les références suivantes :

- Injection de dépendances et de contextes dans Java EE
- CDI 1.2 Javadoc
- Weld 2.3.3.Final - Implementation Référence CDI

### Archives Bean

Les classes de bean des beans activés doivent être déployées dans les archives de bean pour s'assurer qu'elles soient scannées par CDI afin de trouver et traiter les classes de bean.

Dans CDI 1.0, une archive était définie comme archive de bean*explicite* si elle contenait un fichier **`beans.xml`** dans le répertoire **`META-INF/`** pour une application cliente, un EJB, ou un JAR de bibliothèque, ou si elle contenait un fichier **`beans.xml`** dans le répertoire **`WEB-INF/`** pour un WAR.

CDI 1.1 introduit des archives bean *implicites*, qui sont des archives qui contiennent une ou plusieurs classes de bean avec une annotation de définition de bean, ou une ou plusieurs session beans. Les archives bean implicites sont scannées par CDI et, au cours de la découverte du type, seules les classes ayant des annotations de définition de bean sont découvertes. Pour plus d'informations, consultez Type et Découverte de bean dans *Injection de dépendances et contextes dans la plate-forme Java EE*.

Une archive bean possède un mode de découverte de bean coorespondant à **`all`**, **`annoted`** ou **`none`**. Une archive bean qui contient un fichier**`beans.xml`** sans version dispose d'un mode de découverte de bean par défaut de **`all`**. Une archive de bean qui contient un fichier**`beans.xml`** avec la version **`1.1`** ou version ultérieure doit spécifier l'attribut **`bean-discovery-mode`**. La valeur par défaut de l'attribut est **`annoted`**.

Une archive n'est pas une archive bean dans les cas suivants :

- Elle contient un fichier **`beans.xml`** avec un mode **`bean-discovery-mode`** de **`none`**.
- Elle contient une extension de CDI sans fichier **`beans.xml`**.

Une archive est une archive bean *explicite* dans les cas suivants :

- L'archive contient un fichier **`beans.xml`** avec un numéro de version 1.1 ou ultérieur et un**`bean-discovery-mode`** correspondant à **`all`**.
- L'archive contient un fichier **`beans.xml`** sans numéro de version.
- L'archive contient un fichier **`beans.xml`** vide.

L'archive est une archive bean *implicite* dans les cas suivants :

- L'archive contient une ou plusieurs classes de beans avec une annotation de définition de bean, une ou plusieurs sessions de bean, même si elle ne contient-elle pas de fichier **`beans.xml`**.
- L'archive contient un fichier **`beans.xml`** avec un **`bean-discovery-mode`** correspondant à **`annotated`**.

CDI 1.2 limitait les annotations de définition de bean aux suivantes :

- Annotations **`@ApplicationScoped`**, **`@SessionScoped`**, **`@ConversationScoped`**, et **`@RequestScoped`**
- Tous les autres types de scopes habituels
- Annotations **`@Interceptor`** et **`@Decorator`**

- Toutes les annotations de stéréotype, qui sont des annotations annotées par **`@Stereotype`**
- Annotations de scope **`@Dependent`**

Pour plus d'informations sur les archives de bean, voir les Archives Bean dans *Injection de dépendance et contextes dans la plateforme Java EE*.

### Clarification et Résolution de conversations

Le cycle de vie de contexte de conversation a été changé pour éviter tout conflit avec la spécification du Servlet comme décrit dans Problème de spécification CDI-411. Le scope de conversation est actif durant toutes les requêtes du servlet et ne devrait pas empêcher les autres servlets ou filtres de servlets de fixer le corps de la requête ou le codage de caractères.

### Résolution par observation

La résolution de l'événement a été en partie réécrite dans CDI 1.2. Dans CDI 1.0, un événement est remis à une méthode d'observation si la méthode d'observation possède tous les qualificateurs d'événement. Dans CDI 1.2, un événement est remis à une méthode d'observation si la méthode d'observation n'a aucun qualificateur d'événement ou a un sous-ensemble de qualificateurs événement.

## 5.6. MIGRER LES DÉPENDANCES DE MODULE EXPLICITES

L'introduction du système de chargement de classe modulaire et des Modules JBoss dans la version précédente de JBoss EAP permet un contrôle très précis des classes disponibles aux applications. Cette fonctionnalité vous permet de configurer des dépendances de module explicites en utilisant le fichier **`MANIFEST.MF`** de l'application ou le fichier descripteur de déploiement **`jboss-deployment-structure.xml`**.

Si vous avez définit des dépendances de module explicites dans votre application, veuillez prendre en compte les changements suivants apportés à JBoss EAP 7.

### Examiner les dépendances pour la disponibilité

Les modules inclus dans JBoss EAP ont changé. Lorsque vous migrez votre application vers JBoss EAP 7, veuillez examiner les entrées des fichiers **`MANIFEST.MF`** et **`jboss-deployment-structure.xml`** afin de vous assurer qu'elles ne font pas référence à des modules supprimés dans cette version du produit.

### Dépendances requérant un scan d'annotations

Dans la version précédente de JBoss EAP, si votre dépendance contenait des annotations nécessitant d'être traitées pendant le scan des annotations, comme lors de la déclaration d'intercepteurs EJB, il est requis de générer et d'inclure un index Jandex dans un nouveau fichier JAR et de paramétrer un indicateur dans le fichier descripteur de déploiement **`MANIFEST.MF`** ou **`jboss-deployment-structure.xml`**.

JBoss EAP 7 permet désormais de générer un runtime automatique des index d'annotations pour les modules statiques, il n'est ainsi plus nécessaire de les générer manuellement. Cependant, vous devrez toujours ajouter l'indicateur **`annotations`** au fichier de l'application **`MANIFEST.MF`** ou au fichier descripteur de déploiement **`jboss-deployment-structure.xml`** comme indiqué ci-dessous.

**Exemple d'indicateur d'annotation dans le fichier MANIFEST.MF**

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

**Exemple d'indicateur d'annotation dans le fichier jboss-deployment-structure.xml**

```
<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. CHANGEMENTS DE MIGRATION HIBERNATE ET JPA

### Hibernate ORM 5.0

Si votre application contient un fichier **`persistence.xml`** ou si le code utilise les annotations **`@PersistenceContext`** ou **`@PersistenceUnit`**, JBoss EAP 7 les détectera pendant le déploiement et assumera que l'application utilise JPA. Les bibliothèques Hibernate ORM 5.O sont implicitement ajoutées (ainsi que quelques autres dépendances) au chemin de classe de votre application et ces bibliothèques sont utilisées par défaut.

### Hibernate ORM 4.0 - 4.3

Si votre application nécessite que le cache du second niveau soit activé, vous devriez effectuer une migration vers Hibernate ORM 5.0, qui est intégré dans Infinispan 8.0.

Les applications écrites avec Hibernate ORM 4.x peuvent toujours utiliser Hibernate 4.x. Vous devez définir un module JBoss personnalisé pour les JAR Hibernate 4.x et exclure les classes Hibernate 5 de votre application. Cependant, il est fortement recommandé que vous ré-écriviez votre code d'application afin d'utiliser Hibernate 5.

Pour obtenir des informations sur les changements implémentés entre Hibernate 4 et Hibernate 5, veuillez consulter le Guide de migration Hibernate ORM 5.0.

### Hibernate ORM 3.0

Les classes d'intégration qui ont facilité l'utilisation d'Hibernate 3 dans la version précédente ont été supprimées de JBoss EAP 7. Si votre application utilise toujours des bibliothues Hibernate 3, il est fortement recommandé de migrer votre application pour utiliser Hibernate 5 car Hibernate 3 ne fonctionnera plus dans JBoss EAP sans fournir de gros efforts. Si vous ne pouvez pas effectuer la migration vers Hibernate 5, vous devrez définir un module JBoss personnalisé pour les JAR Hibernate 3 et exclure les classes Hibernate 5 de votre application.

## 5.8. CHANGEMENTS APPORTÉS À HIBERNATE SEARCH

La version d'Hibernate Search comprise dans JBoss EAP 7 a changé. L'ancienne version de JBoss EAP comprenait Hibernate Search 4.6.x. JBoss EAP 7 inclut maintenant Hibernate Search 5.5.x.

Hibernate Search 5.5 repose sur Apache Lucene 5.3.1. Si vous utilisez des APIs Lucene natif, assurez-vous de vous aligner avec cette version. L' Hibernate Search 5.5 API encapsule et masque la complexité des nombreux changements des API Lucene qui ont eu lieu entre les versions 3 et 5. Cependant, certaines classes sont désormais déconseillées, renommées ou reconditionnées. Cette section décrit comment ces changements peuvent avoir une incidence sur votre code d'application.

### Changements apportés au mappage dans Hibernate Search

#### Indexation des champs d'id dans les relations intégrées

Quand on utilise une annotation **`@IndexedEmbedded`** pour qu'elle puisse inclure des champs en provenance d'une entité liée, l'**`identifiant`** de l'entité liée n'est plus inclus. Vous pouvez activer l'inclusion de l'**`id`** en utilisant l'attribut **`includeEmbeddedObjectId`** de l'annotation **`@IndexedEmbedded`**.

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

#### Modifications apportées au format d'indexes de dates ou de nombres

Les nombres et des dates sont maintenant indexés sous forme de champs numériques par défaut. Les propriétés de type **`int`**, **`long`**, **`float`**, **`double`** et leurs classes wrapper correspondantes ne sont plus indexées sous forme de chaînes. Au lieu de cela, elles sont désormais indexées en utilisant le codage numérique approprié de Lucene. Les champs **`d'identification`** (id) sont une exception à cette règle. Même quand ils sont représentés par un type numérique, ils sont toujours indexés comme un mot clé de chaîne par défaut. L'utilisation de **`@NumericField`** est désormais obsolète, sauf si vous voulez spécifier une précision personnalisée pour le codage numérique. Vous pouvez garder l'ancien format d'index basés sur une chaîne, en spécifiant explicitement un pont de champ de codage de chaîne. Dans le cas des entiers, c'est org.hibernate.search.bridge.builtin.IntegerBridge. Vérifier le paquet org.hibernate.search.bridge.builtin pour les autres ponts de champs accessibles publiquement.

Les champs **`Date`** et **`Calendrier`** ne sont plus indexés sous forme de chaînes. Au lieu de cela, les instances sont codées comme valeurs longues représentant le nombre de millisecondes depuis le 1er janvier 1970, 00:00:00 GMT. Vous pouvez changer le format d'indexation en utilisant le nouvel enum EncodingType. Par exemple :

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

Le changement de codage des dates et des nombres est important, et peut avoir un grand impact sur le comportement de l'application. Si vous avez une requête qui cible un champ qui avait été codé précédemment en chaîne, mais qui est maintenant codé numériquement, vous devez mettre la requête à jour. Les champs numériques doivent être recherchés par une requête **`NumericRangeQuery`**. Vous devez également vous assurer que tous les champs ciblés par facettage sont en codifiés par chaîne. Si vous utilisez la requête de DSL Search, la requête correcte devra être créée automatiquement pour vous.

### Divers changements apportés à Hibernate Search

- Les options de tri sont améliorées et les champ de codage spécifiés de façon erronée dans les options de tri résultent maintenant en exceptions de runtime. Lucene offre également un système de triage plus performant si les champs utilisés pour le triage sont connus dès le départ. Hibernate Search 5.5 fournit la nouvelle annotation **`@SortableField`** et son compagnon multi-valeurs **`@SortableFields`**. Consultez le Guide de Migration d'Hibernate Search 5.4 à 5.5 pour plus d'informations.

- L'API **`SortField`** de Lucene exige le changement de code d'application suivant :

  Dans les anciennes versions de JBoss EAP, vous définissiez le type de champ de triage dans la demande comme suit :

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

  Voici un exemple sur la façon de le définir dans JBoss EAP 7.

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

- Comme **`SearchFactory`** ne doit être utilisé que par l'intégration ORM, elle a été déplacée du module de **`hibernate-search-engine`** dans le module **`hibernate-search-orm`**. D'autres intégrateurs doivent dépendre exclusivement du **`SearchIntegrator`**, qui remplace l'ancien **`SearchFactoryIntegrator`**.

- L'enum **`SpatialMode.GRID`** a été renommé **`SpatialMode.HASH`**.

- **`FullTextIndexEventListener`** est maintenant une classe finale. Si vous étendez cette classe actuellement, vous devez trouver une autres solution pour obtenir la même fonctionnalité.

- Le module **`hibernate-search-analyzers`** a été supprimé. L'approche conseillée est de rediriger l'utilisation de l'artefact de Lucène qui convient, par exemple **`org.apache.lucene:lucene-analyzers-common`**.

- L'API du contrôleur JMS a changé. La dépendance de backend de JMS sur l'ORM d'Hibernate a été supprimée afin qu'elle puisse être utilisée dans d'autres environnements non-ORM. Une des conséquence est que les implémenteurs de **`org.hibernate.search.backend.impl.jms.AbstractJMSHibernateSearchController`** doivent s'adapter à la nouvelle signature. Cette classe est une classe interne et il est recommandé de l'utiliser comme exemple au lieu de l'étendre.

- Le SPI **`org.hibernate.search.spi.ServiceProvider`** a été remanié. Si vous intégriez avec l'ancien contrat de service, reportez-vous à la Hibernate Search 5.5 Javadoc du **`ServiceManager`**, **`Service`**, **`Startable`** et **`Stoppable`** pour plus de détails sur le nouveau contrat.

- Si vous avez conservé les indexes générés par Lucene 3.x et que vous ne les avez pas reconstruit dans Hibernate Search 5.0 ou version ultérieure, vous obtiendrez une **`IndexFormatTooOldException`**. Il est conseillé de reconstruire les indexes avec l'indexeur en masse. Si vous n'êtes pas capable de le faire, essayez d'utiliser l'**`IndexUpgrader`** de Lucene. Vous devez soigneusement mettre à jour les mappages d'Hibernate Search au cas où le comportement par défaut aurait changé. Pour plus d'informations, consultez le Guide de Migration d'Apache Lucene.

- Apache Lucene est passé de 3.6 à 5.3 dans JBoss EAP 7. Si votre code importe directement des code de Lucene, voir le Guide de Migration d'Apache Lucene pour plus de détails des changements. On trouvera également des informations supplémentaires dans le Journal des changements dans Lucene.

- Lorsque vous utilisez **`@Field(indexNullAs=)`** pour encoder une valeur de marqueur null dans l'index, le type de marqueur doit être compatible avec toutes les autres valeurs qui sont indexées dans ce même champ. Par exemple, il était possible d'encoder une valeur null pour les champs numériques à l'aide d'une chaîne « null ». Ce n'est plus autorisé. Au lieu de cela, vous

devez choisir un nombre pour représenter la valeur **`null`**, comme **`-1`**.

- Des améliorations de taille ont été apportées au moteur de facettage. La plupart des modifications n'affecte pas l'API. La seule exception notable est que vous devez maintenant annoter tous les champs que vous voulez utiliser pour le facettage avec l'annotation **`@Facet`** ou **`@Facets`**.

### Classes d'Hibernate Search renommées ou re-paquetées

Voici un liste de classes d'Hibernate Search qui ont été renommées ou repaquetées.

| Ancien paquet et classe | Nouveau paquet et 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 renommées ou re-paquetées

Les analyseurs de demandes ont été déplacés dans un nouveau module, résultant en changement de packaging de **`org.apache.lucene.queryParser.QueryParser`** à **`org.apache.lucene.queryparser.classic.QueryParser`**.

Bon nombre des analyseurs Lucene ont été refactorisés, entraînant des modifications de packaging. Consultez la Documentation d'Apache Lucene pour trouver les paquets de remplacement.

Certaines classes d'utilitaire Apache Solr, par exemple **`TokenizerFactory`** ou **`TokenFilterFactory`**, ont été déplacés vers Apache Lucene. Si votre application utilise ces utilitaires ou des analyseurs personnalisés, vous devez trouver le nouveau nom du package dans Apache Lucene.

Voir Apache Lucene Migration Guide pour plus d'informations.

### Les API d'Hibernate Search dépréciés

Pour obtenir une liste complète des interfaces d'Hibernate Search obsolètes, des classes, des enums, des types d'annotations, des méthodes, des constructeurs, et des constantes d'enum, voir Hibernate Search Deprecated API.

#### Interfaces d'Hibernate Search dépréciées

| Interface | Description |
| --- | --- |
| org.hibernate.search.store.IndexShardingStrategy | Dépréciée dans Hibernate Search 4.4. Pourrait être supprimée dans Search 5. Utiliser ShardIdentifierProvider à la place. |
| org.hibernate.search.store.Workspace | Cette interface sera supprimée et ne sera plus considérée publique API [HSEARCH-1915]. |

### Classes d'Hibernate Search dépréciées

| Classe | Description |
| --- | --- |
| org.hibernate.search.filter.FilterKey | Les clés de filtrage personnalisées sont dépréciées et devront être supprimées dans Hibernate Search 6. À partir d'Hibernate Search 5.1, les clés pour cacher les filtres de Lucene sont calculées automatiquement sur la base des paramètres de filres donnés. |
| org.hibernate.search.filter.StandardFilterKey | Les clés de filtrage personnalisées sont dépréciées et devront être supprimées dans Hibernate Search 6. À partir d'Hibernate Search 5.1, les clés pour cacher les filtres de Lucene sont calculées automatiquement sur la base des paramètres de filres donnés. |

### Enums dépréciés d'Hibernate Search

| Enum | Description |
| --- | --- |
| org.hibernate.search.annotations.FieldCacheType | Supprimer l'annotation **`CacheFromIndex`** si elle est dépréciée. Voir Annotations d'Hibernate dépréciées. |

### Annotations d'Hibernate Search dépréciées

| Annotation | Description |
| --- | --- |

| Annotation | Description |
| --- | --- |
| org.hibernate.search.annotations.CacheFromIndex | Supprimer l'annotation. Aucun autre remplacement n'est nécessaire. |
| org.hibernate.search.annotations.Key | Les clés de cache de filtre personnalisé constituent une fonctionnalité obsolète et sont programmés pour être supprimées dans Hibernate Search 6. À partir de Hibernate Search 5.1, les clés du cache filtre sont déterminées automatiquement sur la base des paramètres de filtre, donc il n'est plus nécessaire de fournir un objet de clé. |

### Méthodes dépréciées d'Hibernate Search

| Méthode | Description |
| --- | --- |
| org.hibernate.search.FullTextSharedSessionBuilder.autoClose() | Aucun remplacement |
| org.hibernate.search.FullTextSharedSessionBuilder.autoClose(boolean) | Aucun remplacement |
| org.hibernate.search.cfg.IndexedMapping.cacheFromIndex(FieldCacheType…) | Cela sera supprimé sans remplacement. |
| org.hibernate.search.cfg.EntityDescriptor.getCacheInMemory() | Cela sera supprimé sans remplacement. |
| org.hibernate.search.cfg.ContainedInMapping.numericField() | Invoquer **`field().numericField()`** à la place. |
| org.hibernate.search.cfg.EntityDescriptor.setCacheInMemory(Map<String, Object>) | Cela sera supprimé sans remplacement. |
| org.hibernate.search.MassIndexer.threadsForSubsequentFetching(int) | Cette méthode sera remplacée. |

| Méthode | Description |
| --- | --- |
| org.hibernate.search.query.dsl.FuzzyContext.withThreshold(float) | Utiliser **`FuzzyContext.withEditDistanceUpTo(int)`**. |

#### Constructeurs d'Hibernate dépréciés

| Construteurs | Description |
| --- | --- |
| org.hibernate.search.cfg.NumericFieldMapping(PropertyDescriptor, EntityDescriptor, SearchMapping) | Utiliser **`NumericFieldMapping.NumericFieldMapping(String, PropertyDescriptor, EntityDescriptor, SearchMapping)`** à la place. |

### Changements ayant un impact sur les intégrateurs avancés

Cette section décrit les modifications qui ne font pas partie de l'API publique. Elles ne devraient pas influer le développeur moyen car ces artefacts devraient uniquement être accessibles par les intégrateurs qui étendent le framework d'Hibernate Search.

- Les constantes d'enum **`IndexWriterSetting.MAX_THREAD_STATES`** et **`IndexWriterSetting.TERM_INDEX_INTERVAL`** sont obsolètes. Elles affectent les propriétés qui sont lues dans la configuration, donc, le fait qu'elles soient manquantes signifie que les propriétés de configuration comme **`hibernate.search.Animals.2.indexwriter.term_index_interval = default`** sont ignorées. L'autre problème, c'est que la propriété n'est pas appliquée.
- L'interface **`SearchFactoryIntegrator`** est maintenant obsolète. Vous devriez migrer tous les codes immédiatement pour utiliser le **`SearchIntegrator`**.
- La classe **`SearchFactoryBuilder`** est maintenant dépréciée. Utilser **`SearchIntegrationBuilder`** à la place.
- La méthode **`HSQuery.getExtendedSearchIntegrator()`** a été dépréciée. Il est sans doute possible d'utiliser la méthode **`SearchIntegrator`**, mais il vaut mieux la supprimer.
- La méthode **`DocumentBuilderIndexedEntity.getFieldCacheOption()`** est dépréciée. Il n'y a pas de remplacement.
- La méthode **`BuildContext.getIndexingStrategy()`** est dépréciée. Utiliser **`BuildContext.getIndexingMode()`** à la place.
- La méthode **`DirectoryHelper.getVerifiedIndexDir(String, Properties, boolean)`** est dépréciée. Utiliser **`DirectoryHelper.getVerifiedIndexPath(java.lang.String, java.util.Properties, boolean)`** à la place.

- Voici un liste de classes d'Hibernate Search qui ont été renommées ou repaquetées.

| Ancien paquet et classe | Nouveau paquet et 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. MIGRER DES BEANS D'ENTITÉ CMP VERS JPA

Le support aux beans d'entité EJB est facultatif dans Java EE 7 et les beans d'entité ne sont pas supportés dans JBoss EAP 7. Cela signifie que la persistance gérée par conteneur (CMP) (container-managed persistence) et que les beans entity (BMP) (bean-managed persistence ) doivent être réécrits pour utiliser des entités Java Persistence API (JPA) lors de la migration vers JBoss EAP 7.

Dans les versions précédentes de JBoss EAP, des beans d'entité étaient créés dans le code source de l'application en étendant la classe **`javax.ejb.EntityBean`** et en implémentant les méthodes requises. Ils étaient ensuite configurés dans le fichier **`ejb-jar.xml`**. Un bean d'entité CMP était spécifié en utilisant un élément **`<entity>`** qui contenait un élément enfant **`<persistence-type>`** avec une valeur de **Conteneur**. Un entity bean BPM a été spécifié avec un élément**`<entity>`** contenant un élément enfant **`<persistence-type>`** ayant une valeur de **Bean**.

Dans JBoss EAP 7, vous devez remplacer tout bean d'entité CMP et BMP dans votre code par des entités JPA (« Java Persistence API »). Les entités JPA sont créées en utilisant les classes javax.persistence.* et sont définies dans le fichier **`persistence.xml`**.

Ci-dessous figure un exemple de classe d'entité 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;
    }
```

Ci-dessous figure un exemple de fichier **`persistence.xml`**.

```
<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>
```

Pour obtenir des exemples d'entités JPA, voir les quickstarts **`bmt`**, **`cmt`**, et **`hibernate5`** fournis dans JBoss EAP 7.

## 5.10. CHANGEMENTS DANS LES PROPRIÉTÉS DE PERSISTENCE JPA

Une nouvelles propriété de persistance, **`jboss.as.jpa.deferdetach`**, a été ajoutée pour permettre la compatibilité avec le comportement de persistance présent dans les anciennes versions de JBoss EAP.

La propriété **`jboss.as.jpa.deferdetach`** contrôle si le contexte de persistance de transaction-scoped utilisé dans un thread de transaction non-JTA détache des entités chargées après chaque appel de l'**`EntityManager`** ou si il attend que le contexte de persistance soit fermé, par exemple, lors de l'invocation de bean de session se termine. La valeur de la propriété est par défaut sur **`false`**, ce qui signifie que des entités sont détachées ou effacées après chaque appel de l'**`EntityManager`**. C'est le comportement par défaut approprié ainsi défini dans la spécification JPA. Si la valeur de la propriété est définie sur **`true`**, les entités ne sont pas détachées tant que le contexte de persistance est fermé.

Dans JBoss EAP 5, la persistance se comportait comme si la propriété **`jboss.as.jpa.deferdetach`** était définie à la valeur **`true`**. Pour obtenir ce même comportement lors de la migration de votre application de JBoss EAP 5 à JBoss EAP 7, vous devez définir la valeur de la propriété **`jboss.as.jpa.deferdetach`** sur **`true`** dans votre **`persistence.xml`** comme illustré dans l'exemple suivant.

```
<?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>
```

Dans JBoss EAP 6, la persistance se comportait comme si la propriété **`jboss.as.jpa.deferdetach`** était définie à la valeur **`false`**. C'est le même comportement que dans JBoss EAP 7, donc aucun changement n'est nécessaire quand vous migrez votre application.

## 5.11. MIGREZ LE CODE DU CLIENT EJB

Le connecteur distant et le port distant par défaut ont été modifiés dans JBoss EAP 7. Pour obtenir plus d'informations sur ce changement, voir Mise à jour du port et du connecteur d'URL distants.

Si vous avez utilisé l'opération **`migrate`** pour migrer la configuration de votre serveur, les anciens paramètres sont conservés et vous n'avez pas besoin d'ajouter les modifications détaillées ci-dessous. Toutefois, si vous exécutez avec la nouvelle configuration par défaut de JBoss EAP 7, vous devez ajouter les modifications suivantes.

### 5.11.1. Mettez à jour le port de connexion distante par défaut

Modifier la valeur de port de la connexion distante **`4447`** à **`8080`** dans le fichier **`jboss-ejb-client.properties`**.

Ci-dessous figurent des exemples de fichier **`jboss-ejb-client.properties`** dans la version précédente et la version actuelle.

**Exemple : Fichier JBoss EAP 6 jboss-ejb-client.properties**

```
remote.connectionprovider.create.options.org.xnio.Options.SSL_ENABLED=f
alse
remote.connections=default
remote.connection.default.host=localhost
remote.connection.default.port=4447
remote.connection.default.connect.options.org.xnio.Options.SASL_POLICY_
NOANONYMOUS=false
```

**Exemple : Fichier JBoss EAP 7 jboss-ejb-client.properties**

```
remote.connectionprovider.create.options.org.xnio.Options.SSL_ENABLED=f
alse
remote.connections=default
remote.connection.default.host=localhost
remote.connection.default.port=8080
remote.connection.default.connect.options.org.xnio.Options.SASL_POLICY_
NOANONYMOUS=false
```

### 5.11.2. Mettez à jour le connecteur par défaut

Dans la nouvelle configuration de JBoss EAP 7, le connecteur par défaut est passé de **`remote`** à **`http-remoting`**. Ce changement a un impact sur les clients qui utilisent les bibliothèques d'une version de JBoss EAP et pour se connecter au serveur dans une version différente.

- Si une application cliente utilise la bibliothèque EJB à partir de JBoss EAP 6 et souhaite se connecter au serveur JBoss EAP 7, le serveur doit être configuré pour exposer un connecteur distant « **`remote`** » sur un autre port que le port **`8080`**. Le client doit ensuite se connecter en utilisant le connecteur nouvellement configuré.
- Une application cliente qui utilise la bibliothèque cliente EJB de JBoss EAP 7 et souhaite se connecter au serveur JBoss EAP 6 doit veiller à ce que l'instance du serveur n'utilise pas le connecteur **`http-remoting`**, mais le connecteur **`remote`**. Ceci est effectué en définissant une nouvelle propriété de connexion côté client.

  ```
  remote.connection.default.protocol=remote
  ```

### 5.11.3. Migrez le code du client de nommage distant

Si vous exécutez dans la nouvelle configuration de JBoss EAP 7 par défaut, vous devez modifier votre code client pour utiliser le nouveau connecteur et port distant par défaut.

Voici un exemple sur la façon dont les propriétés d'affectation de noms ont été spécifiées dans le code client de JBoss EAP 6.

```
java.naming.factory.initial=org.jboss.naming.remote.client.InitialConte
xtFactory
java.naming.provider.url=remote://localhost:4447
```

Voici un exemple sur la façon de spécifier les propriétés d'affectation de noms dans le code client de JBoss EAP 7.

```
java.naming.factory.initial=org.jboss.naming.remote.client.InitialConte
xtFactory
```

```
java.naming.provider.url=http-remoting://localhost:8080
```

## 5.12. MIGRER LES CONFIGURATIONS DES PLANS DE DÉPLOIEMENT

La spécification du déploiement d'applications Java EE (JSR-88) visait à définir un contrat type pour activer les outils provenant de plusieurs fournisseurs pour configurer et déployer des applications sur n'importe quel produit de plateforme Java EE. Le contrat exigeait que les fournisseurs de produits Java EE implémentant les interfaces **`DeploymentManager`** et autres interfaces de **`javax.enterprise.deploy.spi`** puissent être accédées par les fournisseurs de l'outil. Dans le cas de JBoss EAP 6, un plan de déploiement est identifié par un descripteur XML nommé **`deployment-plan.xml`** qui est livré dans une archive ZIP ou JAR.

Cette spécification n'a été que faiblement adoptée car la plupart des produits de serveur d'application fournissent leurs propre solution de déploiement « riche en fonctionnalités ». Pour cette raison, la prise en charge de JSR-88 a été abandonnée sur Java EE 7, et par conséquent par JBoss EAP 7 aussi.

Si vous avez utilisé la JSR-88 pour déployer votre application, vous devez maintenant utiliser une autre méthode pour déployer l'application. La commande CLI de JBoss EAP **`deploy`** fournit un moyen standard de déployer des archives dans des serveurs autonomes ou dans des groupes de serveurs dans un domaine géré. Pour plus d'informations sur la gestion par interface en ligne de commandes (CLI), consultez le Guide de gestion en ligne de commandes.

## 5.13. MIGRER DES VALVES D'APPLICATIONS PERSONNALISÉES

Vous devez migrer manuellement les valves personnalisés ou des valves qui sont définies dans le fichier XML **`jboss-Web.xml`**. Cela inclut les valves créées en étendant la classe **`org.apache.catalina.valves.ValveBase`** et configurées dans l'élément **`<valve>`** du fichier descripteur **`jboss-Web.xml`**.

**Important**

Les valves personnalisées et celles qui sont définies dans le fichier **`jboss-web.xml`** ne sont pas automatiquement réécrites ou remplacées par le handler d'Undertow correspondant intégré. Pour obtenir des informations sur le mappage de valves dans les handlers d'Undertow, voir Migrate JBoss Web Valves.

Les valves d'authentification doivent être remplacées manuellement en utilisant les mécanismes d'authentification intégrés d'Undertow.

### Migrer des valves configurées dans les déploiements

Dans JBoss EAP 6, vous pouvez définir des valves personnalisées au niveau de l'application en les configurant dans le fichier descripteur d'application web **`jboss-Web.xml`**. Dans JBoss EAP 7, il est possible de le faire avec les handlers d'Undertow également.

Voici un exemple de valve configurée dans le fichier **`jboss-web.xml`** de 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>
```

Pour obtenir plus d'informations sur la façon de créer et de configurer des handlers personnalisés dans JBoss EAP, voir Créer des handlers personnalisés du *Guide de développement*.

#### Migrer des valves d'authentificateur personnalisées

Pour obtenir plus d'informations sur la façon de migrer des valves d'authentificateur, voir Changements dans les applications de sécurité

## 5.14. CHANGEMENTS DANS LES APPLICATIONS DE SÉCURITÉ

Le remplacement de JBoss Web par Undertow nécessite des modifications de configuration de la sécurité dans JBoss EAP 7.

### 5.14.1. Migrer des valves d'authentificateur

Les valves d'authentification doivent être remplacées manuellement en utilisant des mécanismes d'authentification intégrés d'Undertow. Voir les sections suivantes pour obtenir des informations sur la façon de migrer les valves d'authentificateur.

### 5.14.2. Changements dans PicketLink

Pour obtenir des informations sur les changements requis dans SSO avec la configuration SAML v2, voir Changements par rapport aux anciennes versions de JBoss EAP dans *Comment installer SSO avec SAML v2* pour JBoss EAP.

### 5.14.3. Autres changements dans les applications de sécurité

Pour obtenir des informations sur les différences de configuration SSO avec Kerberos, voir Différences de configuration avec les anciennes versions de JBoss EAP dans *Comment installer SSO avec Kerberos* dans JBoss EAP.

## 5.15. CHANGEMENTS DE LA JOURNALISATION DANS JBOSS

Si votre application utilise JBoss Logging, sachez que les annotations dans le paquet **`org.jboss.logging`** sont désormais dépréciées dans JBoss EAP 7. Elles ont été mises dans le paquet **`org.jboss.logging.annotations`**, donc vous devez mettre à jour votre code source pour importer le nouveau package.

Les annotations sont également passées à un numéro d'identification distinct (ID) pour Maven **`groupId:artifactId:version`** (GAV), donc vous devrez ajouter une nouvelle dépendance de projet à **`org.jboss.logging:jboss-logging-annotations`** dans votre fichier **`pom.xml`** du projet.

**Note**

Seules les annotations de journalisation ont été déplacées. Le **`org.jboss.logging.BasicLogger`** et le **`org.jboss.logging.Logger`** existent toujours dans le paquet **`org.jboss.logging`**.

Le tableau suivant dresse la liste des classes d'annotations dépréciées et de leurs remplacements.

**Tableau 5.1. Remplacement d'annotations de journalisation dépréciées**

| Classe dépréciée | Classe de remplacement |
|---|---|
| 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 |

## 5.16. CHANGEMENTS AU CODE JAVASERVER FACES (JSF)

### Support JSF 1.2 abandonné

JBoss EAP 6 vous permet de continuer à utiliser JSF 1.2 avec vos déploiemens d'applications en créant un fichier **`jboss-deployment-structure.xml`**.

JBoss EAP 7 inclut JSF 2.2 et ne supporte plus l'API JSF 1.2. Si votre application utilise JSF 1.2, vous devrez la réécrire pour utiliser JSF 2.2.

### Problème de compatibilité entre JSF 2.1 et JSF 2.2

Les API JSF 2.1 et JSF 2.2 ne sont pas entièrement compatibles. La valeur de la constante **`FACELET_CONTEXT_KEY`** est passée de **`com.sun.faces.facelets.FACELET_CONTEXT`** à **`javax.faces.FACELET_CONTEXT`** entre les versions. Cette valeur est inlined par le compilateur et le code compilé pour une version ne fonctionnera pas avec l'autre.

Les applications qui contiennent du code similaire à l'exemple suivant et sont compilées avec l'API de JSF 2.1, mais qui sont exécutées dans JBoss EAP 7 en utilisant l'API JSF 2.2, engendrent une exception **`NullPointerException`**. Pour résoudre ce problème, vous devez recompiler l'application avec l'API JSF 2.2.

```
Object obj =
FacesContext.getCurrentInstance().getAttributes().get(FaceletContext.FACE
LET_CONTEXT_KEY);
```

Voir JAVASERVERFACES_SPEC_PUBLIC-1257 pour plus d'informations.

## 5.17. CHANGEMENTS AU NIVEAU DU CHARGEMENT DES CLASSES DE MODULES

Dans JBoss EAP 7, le comportement de chargement de classes a changé dans les cas où plusieurs modules contiennent les mêmes classes ou paquets.

Assumons qu'il y ait deux modules, **`MODULE_A`** et **`MODULE_B`**, dépendant l'un de l'autre et qui contiennent des paquets en commun. Dans JBoss EAP 6, les classes ou les paquets qui étaient chargés à partir des dépendances prévalaient sur ceux spécifiés dans **`ressource-root`** du fichier **`module.xml`**. Cela signifie que **`MODULE_A`** voyait les paquets pour **`MODULE_B`** et que **`MODULE_B`** voyait les paquets pour **`MODULE_A`**. Ce comportement a été source de confusion et pouvait entraîner des conflits. Il a changé dans JBoss EAP 7. Maintenant les classes ou les paquets spécifiés par **`ressource-root`** du fichier **`module.xml`** prévalent sur ceux qui sont spécifiés par la dépendance. Cela signifie que **`MODULE_A`** voit les paquets de **`MODULE_A`** et que **`MODULE_B`** voit les paquets de **`MODULE_B`**. Cela empêche les conflits et offre un comportement plus approprié.

Si vous avez défini des modules personnalisés avec des bibliothèques de **`ressources-root`** ou des paquets qui contiennent des classes dupliquées dans leurs dépendances de module, vous pouvez voir les exceptions **`ClassCastException`**, **`LinkageError`**, des erreurs de chargement de classes, ou autres changements dans le comportement de chargement lorsque vous migrez vers JBoss EAP 7. Pour résoudre ces problèmes, vous devez configurer votre fichier **`module.xml`** pour veiller à ce qu'une seule version de classe soit utilisée. Ceci peut être accompli en utilisant l'une des approches suivantes.

- Vous pouvez éviter de spécifier un **`resource-root`** qui multiplie les classes dans la dépendance de module.
- Vous pouvez utiliser les sous-éléments **`include`** et **`exclude`** des éléments **`imports`** et **`exports`** pour contrôler le chargement de classes dans le fichier **`module.xml`**. Ce qui suit est un élément d'exportation qui exclut les classes dans le paquet spécifié.

```
<exports>
  <exclude path="com/mycompany/duplicateclassespath/"/>
</exports>
```

Si vous préférez conserver votre comportement existant, vous devez filtrer les paquets de dépendance par la **`ressource-root`** dépendante dans le fichier **`module.xml`** à l'aide de l'élément **`filter`**. Cela vous permet de conserver le comportement existant sans le bouclage singulier qui vous voyez dans JBoss EAP 6. Voici un exemple d'une **`ressource-root`** qui filtre les classes dans un package spécifié.

```
<resource-root path="mycompany.jar">
  <filter>
    <exclude path="com/mycompany/duplicateclassespath"/>
  </filter>
</resource-root>
```

Pour obtenir davantage d'informations sur les chargements de classes et les modules, voir Chargements de classes et modules dans le *Guide de développement*.

## 5.18. CHANGEMENTS CLUSTERING D'APPLICATIONS

### 5.18.1. Aperçu sur les fonctionnalités du nouveau clustering

La liste suivante décrit certaines des nouvelles fonctionnalités de clustering avec lesquelles il faut être familiarisé avant la migration de votre application de JBoss EAP 6 à JBoss EAP 7.

- JBoss EAP 7 présente une nouvelle API publique pour construire des services singleton qui simplifient le processus considérablement. Pour plus d'informations, voir Implémenter un Singleton HA dans *Développer des Applications EJB* dans JBoss EAP.
- Un déploiement singleton peut être configuré pour déployer et démarrer sur un seul noeud à la fois dans le cluster. Pour plus d'informations, voir Déploiements Singleton HA du *Guide de développement* de JBoss EAP.
- Vous pouvez maintenant définir des MDB singleton clusterisés. Pour plus d'informations, voir MDB Singleton clusterisés dans *Développer des applications EJB* dans JBoss EAP.
- JBoss EAP 7 inclut l'implémentation mod_cluster d'Untertow. Il s'agit d'une solution d'équilibrage des charge pure Java, qui ne requiert aucun serveur web httpd. Pour plus d'informations, voir Configurer JBoss EAP en tant que Front-end Load Balancerdans le *Guide de configuration*.

Le reste de cette section décrit comment les changements niveau clustering peuvent impacter sur la migration de vos applications vers JBoss EAP 7.

### 5.18.2. Changements apportés au Web Session Clustering

JBoss EAP 7 introduit une nouvelle implémentation sur Web Session Clustering. Il remplace l'implémentation antérieure, qui a été étroitement couplée au code source du sous-système JBoss Web existant.

La nouvelle implémentation de Web Session Clustering impacte sur la façon dont l'application est configurée dans le fichier de descripteur XML de l'application web propiétaire **`jboss-web.xml`**. Ce qui suit représente les seuls éléments de configuration de clustering qui subsistent dans ce fichier.

```
<jboss-web>
  ...
  <max-active-sessions>...</max-active-sessions>
  ...
  <replication-config>
    <replication-granularity>...</replication-granularity>
    <cache-name>...</cache-name>
  </replication-config>
  ...
</jboss-web>
```

Le tableau suivant décrit la procédure pour obtenir un comportement similaire pour les éléments du fichier **`jboss-web.xml`** qui sont désormais obsolètes.

| Élément de configuration | Description du changement |
|---|---|
| <max-active-sessions/> | Auparavant, la création d'une session échouait si elle amenait le nombre de sessions actives à dépasser la valeur spécifiée dans **`<max-active-sessions/>`**.<br><br>Dans la nouvelle implémentation, **`<max-active-sessions/>`** est utilisé pour activer la passivation de la session. Si la création de session amène le nombre de sessions actives à dépasser la valeur de **`<max-active-sessions/>`**, alors, la session la plus ancienne connue du gestionnaire de session sera passivée pour faire place à la nouvelle session. |
| <passivation-config/> | L'élément de configuration et ses sous-éléments ne sont plus utilisés dans JBoss EAP 7. |
| <use-session-passivation/> | Auparavant, la passivation était activée en utilisant cet attribut.<br><br>Dans la nouvelle implémentation, la passivation est activée en spécifiant une valeur non-négative pour **`<max-active-sessions/>`**. |
| <passivation-min-idle-time/> | Auparavant, les séances devaient être actives pendant un minimum de temps avant de devenir candidat à la passivation. Cela pouvait entraîner l'échec de créations de sessions, même lorsque la passivation était activée.<br><br>Cette nouvelle implémentation ne prend pas en charge cette logique et évite ainsi ce DoS. |

| Élément de configuration | Description du changement |
|---|---|
| <passivation-max-idle-time/> | Auparavant, une session était passivée après avoir été inactive pendant un certain temps.<br><br>La nouvelle implémentation ne prend en charge que la passivation lazy, ni ne supporte la passivation eager. Les scéances sont passivées uniquement lorsque cela est nécessaire pour se conformer à **`<max-active-sessions/>`**. |
| <replication-config/> | Un certain nombre de sous-éléments sont dépréciés dans la nouvelle implémentation. |
| <replication-trigger/> | Auparavant, cet élément était utilisé pour déterminer quand la réplication de session était déclenchée. La nouvelle implémentation remplace cette option de configuration par une stratégie unique et robuste. Pour plus d'informations, consultez Attributs immuables de session dans le *Guide de développement* de JBoss EAP. |
| <use-jk/> | Auparavant, l'**`instance-id`** du nœud gèrant une demande donnée était annexée à **`jsessionid`**, pour les équilibreurs de charge comme mod_jk, mod_proxy_balancer, mod_cluster, selon la valeur spécifiée pour **`< use-jk/>`**.<br><br>Dans la nouvelle implementation, l'**`instance-id`**, si définie, est toujours annexée à **`jsessionid`**. |
| <max-unreplicated-interval/> | Auparavant, cette option de configuration a été conçue comme une optimisation pour empêcher la réplication de l'horodatage de la session si aucun attribut de session n'avait été modifié. Bien que cela semble une bonne chose, dans la pratique, cela n'empêche pas les RPC, car l'accès aux session nécessite les RPC de transaction cache indépendamment de savoir si tous les attributs de session ont changé.<br><br>Dans la nouvelle implémentation, l'horodatage d'une session est répliqué pour chaque requête. Cela empêche l'accumulation de métadonnées de sessions périmées après un basculement. |
| <snapshot-mode/> | Avant, on pouvait configurer **`<snapshot-mode/>`** en tant qu'**`INSTANT`** ou **`INTERVAL`**. Le file d'attente de réplication d'Infinispan rend cette option de configuration obsolète. |

| Élément de configuration | Description du changement |
|---|---|
| <snapshot-interval/> | Cela était utile uniquement pour **`<snapshot-mode>INTERVAL</snapshot-mode>`**. Comme le **`<snapshot-mode/>`** est obsolète, cette option est obsolète également. |
| <session-notification-policy/> | Auparavant, la valeur spécifiée par cet attribut définissait une politique de déclenchement d'événements de sessions.<br>Dans la nouvelle implémentation, ce comportement est basé sur la spécification et n'est pas configurable. |

### 5.18.3. Changements apportés au Stateful Session EJB Clustering (SFSB)

Dans JBoss EAP 6, vous deviez activer le comportement SFSB de l'une des manières suivantes.

- Vous pouviez ajouter l'annotation **`org.jboss.ejb3.annotation.Clustered`** au session bean.

```
@Stateful
@Clustered
public class MyBean implements MySessionInt {

   public void myMethod() {
      //
   }
}
```

- Vous pouviez ajouter l'élément **`<clustered>`** au fichier **`jboss-ejb3.xml`**.

```
<c:clustering>
  <ejb-name>DDBasedClusteredSFSB</ejb-name>
  <c:clustered>true</c:clustered>
</c:clustering>
```

Dans JBoss EAP 7, il n'est plus nécessaire d'activer le comportement de clustering. Par défaut, si le serveur est démarré à l'aide d'un profil HA, l'état des SFSB sera répliqué automatiquement.

Vous pouvez désactiver ce comportement par défaut d'une des manières suivantes.

- Vous pouvez désactiver le comportement par défaut d'un single stateful session bean avec **`@Stateful(passivationCapable=false)`**, ce qui est nouveau dans la spécification EJB 3.2.
- Vous pouvez désactiver ce comportement globalement dans la configuration du sous-système **`ejb3`** de la configuration du serveur.

**Note**

Si l'annotation **`@Clustered`** n'est pas supprimée de l'application, elle sera ignorée tout simplement et cela n'affectera pas le déploiement de l'application.

### 5.18.4. Changements aux Services de clustering

Dans JBoss EAP 6, les API des services de clustering étaient des modules privés et n'étaient pas pris en charge.

JBoss EAP 7 présente une API de services de clustering publique utilisée par les applications. Les nouveaux services sont conçus pour être légers, injectables facilement, et ne pas nécessiter de dépendances externes.

- La nouvelle interface **`org.wildfly.clustering.group.Group`** donne accès au nouveau statut de cluster actuel et permet d'écouter les changements d'affiliation du cluster.
- La nouvelle interface **`org.wildfly.clustering.dispatcher.CommandDispatcher`** autorise le code existant dans le cluster, sur tous ou quelques sous-éléments des noeuds.

Ces services remplacent des API similaires qui étaient disponibles dans les versions précédentes, comme **`HAPartition`** dans JBoss EAP 5, **`GroupCommunicationService`**, **`GroupMembershipNotifier`**, et **`GroupRpcDispatcher`** dans JBoss EAP 6.

Pour obtenir davantage d'informations, voir l'API Public pour le Clustering de Services dans le *Guide de développement* de JBoss EAP.

### 5.18.5. Migrer le Clustering HA Singleton

Sur JBoss EAP 6, il n'existait pas d'API publique disponible au service global au cluster HA singleton. Si vous utilisiez des classes privées **`org.jboss.as.clustering.singleton.*`**, vous devez changer votre code pour utiliser les nouveaux paquets publics **`org.wildfly.clustering.singleton.*`** lorsque vous migrez votre application vers JBoss EAP 7.

Pour obtenir davantage d'informations sur la façon d'implémenter un singleton HA, voir Implémenter un Singleton HA dans *Développer des Applications EJB* de JBoss EAP.

# CHAPITRE 6. DIVERS CHANGEMENTS

## 6.1. CHANGEMENT DANS LE MODE DE LIVRAISON DE JBOSS EAP NATIVES ET APACHE HTTP SERVER

JBoss EAP 7 natifs sont livrés différemment dans cette version. Certains sont maintenant livrés avec le nouveau produit Red Hat JBoss Core Services, qui est un ensemble de logiciels complémentaires commun à beaucoup de produits Red Hat JBoss Middleware. Le nouveau produit permet une diffusion plus rapide des mises à jour et une expérience plus cohérente pour la mise à jour. Le produit JBoss Core Services est disponible en téléchargement à un emplacement différent sur le Portail Client de Red Hat.

- Le tableau suivant établit la liste des différences de mode de livraison entre les versions.

| Package | JBoss EAP 6 | JBoss EAP 7 |
|---|---|---|
| AIO Natives Messagerie | Livré avec le produit dans un paquet "Native Utilities" séparé | Inclut dans la distribution de JBoss EAP. Aucun téléchargement supplémentaire requis. |
| Serveur HTTP Apache | Livré avec le produit dans un paquet "Apache HTTP Server" séparé | Livré avec le produit JBoss Core Services |
| mod_cluster, mod_jk, isapi, et connecteurs nsapi | Livré avec le produit dans un paquet "Webserver Connector Natives" séparé | Livré avec le produit JBoss Core Services |
| JSVC | Livré avec le produit dans un paquet "Native Utilities" séparé | Livré avec le produit JBoss Core Services |
| OpenSSL | Livré avec le produit dans un paquet "Native Utilities" séparé | Abandonné dans JBoss EAP 7 |
| tcnatives | Livré avec le produit dans un téléchargement "Native Components" séparé | Abandonné dans JBoss EAP 7 |

- Nous vous informons des changements suivants :
  - Abandon du support aux connecteurs mod_cluster et mod_jk utilisés dans le serveur HTTP Apache à partir des canaux Red Hat Enterprise Linux RPM. Si vous exécutez le serveur HTTP Apache à partir des canaux Red Hat Enterprise Linux RPM et que vous devez configurer l'équilibrage des charge pour les serveurs JBoss EAP 7, vous pouvez effectuer

une des opérations suivantes :

- Utiliser le serveur HTTP Apache fourni par JBoss Core Services.
- Vous pouvez configurer JBoss EAP 7 pour qu'il agisse en tant qu'équilibreur des charges de front-end. Pour plus d'informations, voir Configurer JBoss EAP en tant qu'équibreur de charge de front-end dans le *Guide de configuration* de JBoss EAP.
- Vous pouvez déployer le serveur Apache HTTP sur une machine prise en charge et certifiée, puis exécuter l'équilibreur de charge sur cette machine. Pour obtenir une liste des configurations prises en charge, voir Aperçu général des connecteurs HTTP dans le *Guide de configuration* de JBoss EAP 7.

- Abandon du support aux connecteurs mod_cluster et mod_jk utilisés dans le serveur Apache HTTP à partir de HP-UX Web Server Suites. Si vous exécutez le serveur Apache HTTP à partir de HP-UX Web Server Suites et que vous devez configurer l'équilibrage des charges pour les serveurs JBoss EAP 7, vous pouvez effectuer une des opérations suivantes :
  - Vous pouvez configurer JBoss EAP 7 pour qu'il agisse en tant qu'équilibreur des charges de front-end. Pour plus d'informations, voir Configurer JBoss EAP en tant qu'équibreur de charge de front-end dans le *Guide de configuration* de JBoss EAP.
  - Vous pouvez déployer le serveur Apache HTTP sur une machine prise en charge et certifiée, puis exécuter l'équilibreur de charge sur cette machine. Pour obtenir une liste des configurations prises en charge, voir Aperçu général des connecteurs HTTP dans le *Guide de configuration* de JBoss EAP.

- Vous allez trouver des informations sur JBoss Core Services dans le *Guide d'installation du serveur Apache HTTP*.

## 6.2. CHANGEMENTS AUX DÉPLOIEMENTS DANS AMAZON EC2

Il y eu un certain nombre de changements à JBoss EAP 7.0 Amazon Machine Images (AMI) dans JBoss EAP 7. Cette section résume rapidement certains de ces changements.

- La façon de lancer des instances JBoss EAP clusterisées ou non-clusterisées, et des domaines dans Amazon EC2 a changé considérablement.
- JBoss EAP 6 a utilisé le champ **`User Data:`** pour la configuration JBoss EAP. Les scripts AMI qui analysaient la configuration dans le champ **`User Data:`** et démarraient les serveurs automatiquement au lancement de l'instance ont été retirés de JBoss EAP 7.
- L'agent Red Hat JBoss Operations Network a été installé dans une ancienne version deJBoss EAP. Dans JBoss EAP 7, vous devez l'installer séparemment.

Pour comprendre comment déployer JBoss EAP 7 sur Amazon EC2, voir *Déployer Red Hat JBoss Enterprise Application Platform dans Amazon EC2*.

# ANNEXE A. MATÉRIEL DE RÉFÉRENCE

## A.1. AVERTISSEMENTS POUR L'OPÉRATION DE MIGRATION DU SOUS-SYSTÈME JACORB

L'opération **`migrate`** n'est pas en mesure de traiter toutes les ressources et les attributs. Voici quelques-uns des éventuels avertissements qui peuvent s'afficher lorsque vous exécutez une opération **`migrate`** ou **`describe-migration`** dans le sous-système **`jacorb`**.

**Note**

Si vous voyez «Impossible de migrer » dans la sortie de l'opération de **`migration`**, cela indique que la migration de la configuration du serveur est terminée, mais qu'on n'a pas pu migrer automatiquement tous les éléments et attributs. Vous devez suivre les conseils donnés dans les «avertissements de migration» pour modifier ces configurations.

| Message d'avertissement | Ce que cela signifie / Comment les régler |
| --- | --- |
| La migration **`iiop`** peut être effectuée quand le serveur est en mode **`admin-only`** | L'opération **`migrate`** exige de démarrer le serveur en mode **`admin-only`**, ce qui est fait en ajoutant le paramètre **`--admin-only`** à la commande de démarrage du serveur :<br>`$ EAP_HOME/bin/standalone.sh --admin-only` |

| Message d'avertissement | Ce que cela signifie / Comment les régler |
|---|---|
| Les propriétés *X* ne peuvent pas être émulées avec OpenJDK ORB et ne sont pas prises en charge. | La configuration de la propriété spécifiée n'est pas prise en charge et n'est pas incluse dans la nouvelle configuration de sous-système **`iiop-openjdk`**. Le comportement de cette propriété dans la version précédente de JBoss EAP n'est pas migré et l'administrateur doit vérifier que le nouveau sous-système **`iiop-openjdk`** de JBoss EAP 7 est capable de fonctionner correctement sans ce comportement.<br><br>Propriétés non prises en charge : **`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`**. |
| Les propriétés *X* utilisent des expressions. Les propriétés de configuration qui sont utilisées pour résoudre ces expressions doivent être transformées manuellement au nouveau format du sous-système **`iiop-openjdk`** | Les propriétés qui utilisent des expressions doivent être configurées manuellement par l'administrateur.<br><br>Par exemple, le sous-système **`jacorb`** de JBoss EAP 6 définissait une propriété **`giop-minor-version`**. Le sous-système **`iiop-openjdk`** de JBoss EAP 7 définit une propriété **`giop-version`**. Supposons que l'attribut de la version mineure de sous-système **`jacorb`** soit défini sur **`${iiop-giop-minor-version}`** et que la propriété du système soit configurée dans le fichier **`standalone.conf`** à **`-Diiop-giop-minor-version=1`**. Après l'opération de **`migration`**, l'administrateur doit modifier la valeur système de la propriété **`1.1`** pour assurer que le nouveau sous-système soit correctement configuré. |
| Impossible de migrer : le nouveau sous-système **`iiop-openjdk`** est déjà défini | Le message contient une explication |

## A.2. AVERTISSEMENTS POUR L'OPÉRATION DE MIGRATION DU SOUS-SYSTÈME DE MESSAGERIE

L'opération **migrate** n'est pas en mesure de traiter toutes les ressources et attributs. Voici quelques-uns des éventuels avertissements qui peuvent s'afficher lorsque vous exécutez une opération **migrate** ou **describe-migration** dans le sous-système de **messagerie**.

**Note**

Si vous voyez «Impossible de migrer » dans la sortie de l'opération de **migration**, cela indique que la migration de la configuration du serveur est terminée, mais qu'on n'a pas pu migrer automatiquement tous les éléments et attributs. Vous devez suivre les conseils donnés dans les «avertissements de migration» pour modifier ces configurations.

| Message d'avertissement | Ce que cela signifie / Comment les régler |
|---|---|
| L'opération **migrate** ne peut pas être effectuée : le serveur doit être en mode **admin-only** | L'opération **migrate** exige de démarrer le serveur en mode **admin-only**, ce qui est fait en ajoutant le paramètre **--admin-only** à la commande de démarrage du serveur :<br>`$ EAP_HOME/bin/standalone.sh --admin-only` |
| Impossible de migrer l'attribut **local-bind-address** de la ressource *X*. Utiliser l'attribut**socket-binding** pour configurer **broadcast-group** à la place. | Le message contient une explication et indique la façon de régler ce problème. |
| Impossible de migrer l'attribut **local-bind-port** de la ressource *X*. Utiliser l'attribut**socket-binding** pour configurer **broadcast-group** à la place. | Le message contient une explication et indique la façon de régler ce problème. |
| Impossible de migrer l'attribut **group-address** de la ressource *X*. Utiliser l'attribut **socket-binding** pour configurer **broadcast-group** à la place. | Le message contient une explication et indique la façon de régler ce problème. |

| Message d'avertissement | Ce que cela signifie / Comment les régler |
|---|---|
| Impossible de migrer l'attribut **`group-port`** de la ressource *X*. Utiliser l'attribut **`socket-binding`** pour configurer **`broadcast-group`** à la place. | La ressource **`broadcast-group`** n'accepte plus les attributs **`local-bind-address`**, **`local-bind-port`**, **`groupe-address`** ou **`group-port`**. Elle accepte uniquement un attribut **`socket-binding`**. L'avertissement indique que la ressource *X* possède un attribut non pris en charge. Vous devez manuellement affecter cet attribut **`socket-binding`** à la ressource et vous assurer qu'il corresponde bien à une ressource **`socket-binding`** définie. |
| Les classes fournissant *X* sont ignorées pendant la migration. Pour les utiliser dans le nouveau sous-système **`messaging-activemq`**, vous devrez étendre l'**`intercepteur`** basé Artemis. | Le support aux intercepteurs de messagerie est sensiblement différent dans JBoss EAP 7. Les intercepteurs configurés dans la version précédente du sous-système sont ignorés lors de la migration. Pour plus d'informations, voir Migrer les intercepteurs de messagerie. |
| N'a pas pu migrer la configuration HA de *X*. Ses attributs **`shared-store`** et **`backup`** contiennent des expressions, et il n'est pas possible de déterminer sans ambiguïté comment créer la **`ha-policy`** correspondante pour le serveur de la messagerie-activemq. | Cela signifie que les attributs **`hornetq-server`** *X* : **`shared-store`** ou **`backup`** contiennent une expression, qui ressemble à ${xxx}, et que l'opération de migration n'a pas pu la résoudre en expression concrète. La valeur est ignorée et la **`ha-policy`** de **`messaging-activemq`** doit être mise à jour manuellement. |
| Impossible de migrer l'attribut **`local-bind-address`** de la ressource *X*. Utiliser l'attribut**`socket-binding`** pour configurer **`discovery-group`** à la place. | Le message contient une explication et indique la façon de régler ce problème. |
| Impossible de migrer l'attribut **`local-bind-port`** de la ressource *X*. Utiliser l'attribut**`socket-binding`** pour configurer **`discovery-group`** à la place. | Le message contient une explication et indique la façon de régler ce problème. |
| Impossible de migrer l'attribut **`group-address`** de la ressource *X*. Utiliser l'attribut **`socket-binding`** pour configurer **`discovery-group`** à la place. | Le message contient une explication et indique la façon de régler ce problème. |

| Message d'avertissement | Ce que cela signifie / Comment les régler |
| --- | --- |
| Impossible de migrer l'attribut **`group-port`** de la ressource *X*. Utiliser l'attribut **`socket-binding`** pour configurer **`discovery-group`** à la place. | Les ressources **`discovery-group`** n'acceptent plus les attributs **`local-bind-address`**, **`local-bind-port`**, **`group-address`** ou **`group-port`**. Elles acceptent uniquement un **`socket-binding`**. L'avertissement indique que la ressource *X* possède un attribut non pris en charge. Vous devez manuellement affecter cet attribut **`socket-binding`** à la ressource et vous assurer que cela correspond à une ressource **`socket-binding`** définie. |
| Ne peut pas créer un **`legacy-connection-factory`** basé sur **`connexion-factory`** *X*. Utilise un connecteur **`in-vm`** HornetQ qui n'est pas compatible avec le connecteur **`in-vm`** d'Artemis | Les ressources distantes héritées de **`connection-factory`** d'HornetQ sont migrées dans les ressources **`legacy-connection-factory`** pour permettre aux clients de JBoss EAP 6 de se connecter à JBoss EAP 7. Cependant, les ressources **`legacy-connection-factory`** ne sont créés que lorsque **`connection-factory`** utilise des connecteurs distants. Toute **`connection-factory`** utilisant **`in-vm`** n'est pas migrée car les clients **`in-vm`** sont basés sur JBoss EAP 7, et non pas JBoss EAP 6. Cet avertissement est pour vous notifier que la **`in-vm`** **`connection-factory`** n'a pas migrée. |
| Impossible de migrer l'attribut *X* de la ressource *Y*. L'attribut utilise une expression qui peut être résolue différemment suivant les propriétés système. Après la migration, cet attribut doit être ajouté à nouveau avec la valeur réelle et non pas l'expression. | Cet avertissement apparaît lorsque la migration ne peut pas résoudre l'attribut *X* en une valeur réelle lors du processus de migration. La valeur est ignorée et l'attribut doit être migré manuellement. Cela se produit dans les cas suivants :<br>- **`cluster-connection`** **`forward-when-no-consumers`** :<br>Cet attribut de booléen a été remplacé par l'attribut **`message-load-balancing-type`**, qui est un enum avec une valeur correspondant à **`OFF`**, **`STRICT`**, ou **`ON_DEMAND`**.<br>- Pour **`broadcast-group`** et **`discovery-group`**, les attributs **`jgroups-stack`** et **`jgroups-channel`**<br>Ils référencent d'autres ressources et JBoss EAP 7 n'accepte plus ces expressions. |

| Message d'avertissement | Ce que cela signifie / Comment les régler |
| --- | --- |
| Impossible de migrer l'attribut *X* de la ressource *Y*. Cet attribut n'est pas pris en charge par le nouveau sous-système **`messaging-activemq`**. | Certains attributs ne sont plus pris en charge par le nouveau sous-système **`messaging-activemq`** et sont ignorés tout simplement :<br>- Attribut **`hornetq-server`**'s **`failback-delay`**<br>- Attribut **`http-connector`** **`use-nio`**<br>- Attribut **`http-acceptor`** **`use-nio`**<br>- Attribut **`remote-connector`** **`use-nio`**<br>- Attribut **`remote-acceptor`** **`use-nio`** |
| Impossible de migrer l'attribut **`failback-delay`** de la ressource *X*. Artemis détecte la restauration automatique de façon déterministe et il n'a plus besoin de spécifier un délai pour que la restauration automatique ait lieu. | Le message contient une explication |

### Remplacer les attributs broadcast-group et discovery-group

Si on vous conseille de remplacer les attributs **`broadcast-group`** et **`discovery-group`** par l'attribut **`socket-binding`**, vous pouvez ajouter le nouvel attribut en ligne de commande (CLI).

Cet exemple assume que vous migrez un serveur autonome qui contient la configuration **`discovery-group`** dans le sous-système de **`messagerie`**.

```
<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>
```

Quand vous exécutez l'opération **`migrate`** pour le sous-système **`messaging`**, vous verrez les messages d'avertissement et de sortie suivants :

```
[standalone@localhost:9999 /] /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."
    ]}
}
```

L'opération **`migrate`** crée un **`discovery-group`** nommé "my-discovery-group" dans le nouveau sous-système **`messaging-activemq`** qui est maintenant configuré ainsi :

```
<discovery-group name="my-discovery-group"/>
```

Vous devez maintenant utiliser la commande CLI pour créer un élément **`socket-binding`** dans le fichier de configuration de serveur nommé "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)
```

Ensuite, ajoutez la **`socket-binding`** nouvellement créée au **`discovery-group`** nommé "my-discovery-group" dans le sous-système **`messaging-activemq`** du fichier de configuration de serveur à l'aide de la commande CLI suivante :

```
/subsystem=messaging-activemq/server=default/discovery-group=my-
discovery-group:write-attribute(name=socket-binding,value=my-discovery-
group-socket-binding)
```

Ces commandes créent le XML suivant dans le fichier de configuration du serveur.

```
<subsystem xmlns="urn:jboss:domain:messaging-activemq:1.0">
    <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. AVERTISSEMENTS POUR L'OPÉRATION DE MIGRATION DU SOUS-SYSTÈME WEB

L'opération **`migrate`** n'est pas en mesure de traiter toutes les ressources et les attributs. Voici quelques-uns des éventuels avertissements qui peuvent s'afficher lorsque vous exécutez une opération **`migrate`** ou **`describe-migration`** dans le sous-système **`web`**.

**Note**

Si vous voyez «Impossible de migrer » dans la sortie de l'opération de **`migration`**, cela indique que la migration de la configuration du serveur est terminée, mais qu'on n'a pas pu migrer automatiquement tous les éléments et attributs. Vous devez suivre les conseils donnés dans les «avertissements de migration» pour modifier ces configurations.

| Message d'avertissement | Ce que cela signifie / Comment les régler |
|---|---|
| L'opération migrate n'est autorisée qu'en mode admin | L'opération **`migrate`** exige de démarrer le serveur en mode **`admin-only`**, ce qui est fait en ajoutant le paramètre **`--admin-only`** à la commande de démarrage du serveur :<br>`$ EAP_HOME/bin/standalone.sh --admin-only` |
| Impossible de migrer la ressource *X* | Le comportement de cette ressource dans la version précédente de JBoss EAP n'a pas migré. L'administrateur doit vérifier si le nouveau sous-système d'**`Undertow`** dans JBoss EAP 7 est capable de fonctionner correctement sans que ce comportement et si le comportement doit être migré manuellement. |
| Impossible de migrer l'attribut *X* de la ressource *Y* | Le comportement de cet attribut de ressource dans la version précédente de JBoss EAP n'a pas été migré. L'administrateur doit vérifier si le nouveau sous-système d'**`Undertow`** de JBoss EAP 7 est capable de fonctionner correctement sans ce comportement et si le comportement doit être migré manuellement. |
| N'a pas pu migrer le connecteur SSL car aucune config SSL n'est définie | Le message contient une explication |
| N'a pas pu migrer l'attribut **`verify-client`** *X* dans l'équivalent d'Undertow. | Le message contient une explication |
| Impossile de migrer l'expression **`verify-client`** *X* | Le message contient une explication |

| Message d'avertissement | Ce que cela signifie / Comment les régler |
| --- | --- |
| Impossible de migrer la valve *X* | Le comportement de cette valve dans la version précédente de JBoss EAP n'a pas été migré. L'administrateur doit vérifier si le nouveau sous-système d'**`Undertow`** dans JBoss EAP 7 est capable de fonctionner correctement sans ce comportement et si le comportement doit être migré manuellement.<br><br>Ce message d'avertissement peut s'afficher pour les valves suivantes :<br><br>- **`org.apache.catalina.valves.RemoteAddrValve`**<br><br>  Doit posséder une valeur autorisée ou non au moins<br>- **`org.apache.catalina.valves.RemoteHostValve`**<br><br>  Doit posséder une valeur autorisée ou non au moins<br>- **`org.apache.catalina.authenticator.BasicAuthenticator`**<br>- **`org.apache.catalina.authenticator.DigestAuthenticator`**<br>- **`org.apache.catalina.authenticator.FormAuthenticator`**<br>- **`org.apache.catalina.authenticator.SSLAuthenticator`**<br>- **`org.apache.catalina.authenticator.SpnegoAuthenticator`**<br>- valves personnalisées |

| Message d'avertissement | Ce que cela signifie / Comment les régler |
| --- | --- |
| Impossible de migrer l'attribut *X* de la valve *Y* | Le comportement de cet attribut de valve dans la version précédente de JBoss EAP n'a pas été migré. L'administrateur doit vérifier si le nouveau sous-système d'**`Undertow`** de JBoss EAP 7 est capable de fonctionner correctement sans ce comportement et si le comportement doit être migré manuellement. Ce message d'avertissement peut se produire pour les attributs de valve suivants :<br>- **`org.apache.catalina.valves.AccessLogValve`**<br>  - **`resolveHosts`**<br>  - **`fileDateFormat`**<br>  - **`renameOnRotate`**<br>  - **`encoding`**<br>  - **`locale`**<br>  - **`requestAttributesEnabled`**<br>  - **`buffered`**<br>- **`org.apache.catalina.valves.ExtendedAccessLogValve`**<br>  - **`resolveHosts`**<br>  - **`fileDateFormat`**<br>  - **`renameOnRotate`**<br>  - **`encoding`**<br>  - **`locale`**<br>  - **`requestAttributesEnabled`**<br>  - **`buffered`**<br>- **`org.apache.catalina.valves.RemoteIpValve`**<br>  - **`httpServerPort`**<br>  - **`httpsServerPort`**<br>  - **`remoteIpHeader`**<br>  Si défini, mais non configuré à "x-forwarded-for"<br>  - **`protocolHeader`**<br>  Si défini, mais non configuré à "x-forwarded-proto" |

## A.4. COMPATIBILITÉ ET INTEROPÉRABILITÉ ENTRE LES VERSIONS

Cette section décrit la compatibilité et l'interopérabilité Client/Serveur EJB et les composants de messagerie entre les versions de JBoss EAP 5, JBoss EAP 6, et JBoss EAP 7.

### EJB distant via IIOP

Vous ne devriez pas rencontrer de problèmes avec les configurations suivantes :

- Se connecter d'un client JBoss EAP 5 à un serveur JBoss EAP 7
- Se connecter d'un client JBoss EAP 6 à un serveur JBoss EAP 7
- Se connecter d'un client JBoss EAP 7 à un serveur JBoss EAP 6
- Se connecter d'un client JBoss EAP 7 à un serveur JBoss EAP 5

### EJB distant via JNDI

Vous ne devriez pas rencontrer de problèmes avec les configurations suivantes :

- Se connecter d'un client JBoss EAP 6 à un serveur JBoss EAP 7
- Se connecter d'un client JBoss EAP 7 à un serveur JBoss EAP 6

JBoss EAP 6 fournit un support pour la spécification EJB 3.1 et introduit l'usage des espace-noms JNDI globaux normalisés, encore utilisés dans JBoss EAP 7. En raison du changement dans les espace-noms JNDI, les configurations suivantes ne sont pas compatibles :

- Se connecter d'un client JBoss EAP 5 à un serveur JBoss EAP 7 ou à un serveur JBoss EAP 6
- Se connecter d'un client JBoss EAP 7 ou JBoss EAP 6 à un serveur JBoss EAP 5

Pour plus d'informations sur les changements d'espace-noms JNDI standardisés, voir Changements JNDI dans le *Guide de migration* de JBoss EAP 6.

### EJB distant utilisant @WebService

Vous ne devriez pas rencontrer de problèmes avec les configurations suivantes :

- Se connecter d'un client JBoss EAP 5 à un serveur JBoss EAP 7
- Se connecter d'un client JBoss EAP 6 à un serveur JBoss EAP 7
- Se connecter d'un client JBoss EAP 7 à un serveur JBoss EAP 6
- Se connecter d'un client JBoss EAP 7 à un serveur JBoss EAP 5

### Client autonome de messagerie

Vous ne devriez pas rencontrer de problèmes avec les configurations suivantes :

- Se connecter d'un client JBoss EAP 6 à un serveur JBoss EAP 7
- Se connecter d'un client JBoss EAP 7 à un serveur JBoss EAP 6

Dans la configuration suivante, si le client utilise l'API HornetQ d'une messagerie spécifique plutôt que l'API JMS générique, la connexion est possible. Cependant, les recherches JNDI doivent être adressées via l'extension de nommage JNDI de l'ancien JBoss EAP hérité fournie dans JBoss EAP 7.

- Se connecter d'un client JBoss EAP 5 à un serveur JBoss EAP 7

La messagerie intégrée de JBoss EAP 7 n'est pas en mesure de se connecter à HornetQ 2.2.x fourni dans JBoss EAP 5 en raison de problèmes de compatibilité de protocole. Pour cette raison, les configurations suivantes ne sont pas compatibles.

- Se connecter d'un client JBoss EAP 7 à un serveur JBoss EAP 5

### MDB Messagerie

Vous ne devriez pas rencontrer de problèmes avec les configurations suivantes :

- Se connecter d'un client JBoss EAP 6 à un serveur JBoss EAP 7
- Se connecter d'un client JBoss EAP 7 à un serveur JBoss EAP 6

Dans la configuration suivante, si le client utilise l'API HornetQ d'une messagerie spécifique plutôt que l'API JMS générique, la connexion est possible. Cependant, les recherches JNDI doivent être adressées via l'extension de nommage JNDI de l'ancien JBoss EAP hérité fournie dans JBoss EAP 7.

- Se connecter d'un client JBoss EAP 5 à un serveur JBoss EAP 7

La messagerie intégrée de JBoss EAP 7 n'est pas en mesure de se connecter à HornetQ 2.2.x fourni dans JBoss EAP 5 en raison de problèmes de compatibilité de protocole. Pour cette raison, les configurations suivantes ne sont pas compatibles.

- Se connecter d'un client JBoss EAP 7 à un serveur JBoss EAP 5

### Ponts JMS

Vous ne devriez pas rencontrer de problèmes avec les configurations suivantes :

- Se connecter d'un client JBoss EAP 5 à un serveur JBoss EAP 7
- Se connecter d'un client JBoss EAP 6 à un serveur JBoss EAP 7
- Se connecter d'un client JBoss EAP 7 à un serveur JBoss EAP 6
- Se connecter d'un client JBoss EAP 7 à un serveur JBoss EAP 5