HP Operations Orchestration

Windows および Linux向け

ソフトウェアバージョン: 10.10

システム構成とハードニングガイド

ドキュメントリリース日: 2014 年 5 月 (英語版)

ソフトウェアリリース日: 2014 年 5 月

ご注意

保証

HP製品、またはサービスの保証は、当該製品、およびサービスに付随する明示的な保証文によってのみ規定されるものとします。ここでの記載は、追加保証を提供するものではありません。ここに含まれる技術的、編集上の誤り、または欠如について、HPはいかなる責任も負いません。

ここに記載する情報は、予告なしに変更されることがあります。

権利の制限

機密性のあるコンピューターソフトウェアです。これらを所有、使用、または複製するには、HPからの有効な使用許諾が必要です。商用コンピューターソフトウェア、コンピューターソフトウェアに関する文書類、および商用アイテムの技術データは、FAR12.211および12.212の規定に従い、ベンダーの標準商用ライセンスに基づいて米国政府に使用許諾が付与されます。

著作権について

商標について

Adobe™は、Adobe Systems Incorporated (アドビシステムズ社) の登録商標です。

本製品には、'zlib' (汎用圧縮ライブラリ) のインタフェースが含まれています。'zlib': Copyright © 1995-2002 Jean-loup Gailly and Mark Adler.

AMDおよびAMD Arrowのシンボルは、Advanced Micro Devices, Inc.の登録商標です。

Google™およびGoogle Maps™は、Google Inc.の登録商標です。

Intel®、Itanium®、Pentium®、Intel®およびXeon®は、Intel Coporationの米国およびその他の国における登録商標です。

Javaは、Oracle Corporationおよびその関連会社の登録商標です。

Microsoft®, Windows®、Windows NT®、Windows® XP、およびWindows Vista®は、米国におけるMicrosoft Corporationの登録商標です。

Oracleは、Oracle Corporationおよびその関連会社の登録商標です。

UNIX® は、The Open Group の登録商標です。

ドキュメントの更新情報

このマニュアルの表紙には、以下の識別情報が記載されています。

- ソフトウェアバージョンの番号は、ソフトウェアのバージョンを示します。
- ドキュメントリリース日は、ドキュメントが更新されるたびに変更されます。
- ソフトウェアリリース日は、このバージョンのソフトウェアのリリース期日を表します。

更新状況、およびご使用のドキュメントが最新版かどうかは、次のサイトで確認できます。**http://h20230.www2.hp.com/selfsolve/manuals**

このサイトを利用するには、HP Passportへの登録とサインインが必要です。HP Passport IDの登録は、次のWebサイトから行なうことができます。**http://h20229.www2.hp.com/passport-registration.html**

または、HP Passport のログインページの **[New users - please register]** リンクをクリックします。

適切な製品サポートサービスをお申し込みいただいたお客様は、更新版または最新版をご入手いただけます。詳細は、HPの営業担当にお問い合わせください。

サポート

HPソフトウェアサポートオンラインWebサイトを参照してください。**http://www.hp.com/go/hpsoftwaresupport**

このサイトでは、HPのお客様窓口のほか、HPソフトウェアが提供する製品、サービス、およびサポートに関する詳細情報をご覧いただけます。

HPソフトウェアオンラインではセルフソルブ機能を提供しています。お客様のビジネスを管理するのに必要な対話型の技術サポートツールに、素早く効率的にアクセスできます。HPソフトウェアサポートのWebサイトでは、次のようなことができます。

- 関心のあるナレッジドキュメントの検索
- サポートケースの登録とエンハンスメント要求のトラッキング
- ソフトウェアパッチのダウンロード
- サポート契約の管理
- HPサポート窓口の検索
- 利用可能なサービスに関する情報の閲覧
- 他のソフトウェアカスタマーとの意見交換
- ソフトウェアトレーニングの検索と登録

一部のサポートを除き、サポートのご利用には、HP Passportユーザーとしてご登録の上、サインインしていただく必要があります。また、多くのサポートのご利用には、サポート契約が必要です。HP Passport IDを登録するには、次のWebサイトにアクセスしてください。

http://h20229.www2.hp.com/passport-registration.html

アクセスレベルの詳細については、次のWebサイトをご覧ください。

http://h20230.www2.hp.com/new_access_levels.jsp

HP Software Solutions Nowは、HPSWのソリューションと統 合 に関 するポータルWebサイト です。このサイト では、お客 様 のビジネスニーズを満 たすHP製 品 ソリューション を検 索 したり、HP製 品 間 の統 合 に関 する詳 細 なリスト やITILプロセスのリストを閲 覧 することができます。このサイト のURL は**http://h20230.www2.hp.com/sc/solutions/index.jsp** です。

目次

システム構成とハードニング

このドキュメントでは、HP Operations Orchestration の構成方法とハードニング方法について説明します。

サーバーおよびクライアント証明書の認証

Secure Sockets Layer (SSL)/トランスポートレイヤーセキュリティ (TLS) 証明書は、暗号キーを組織の詳細にデジタル的に結び付けます。これにより、Web サーバーからブラウザーへのセキュアな接続が可能になります。

HP OO では、Keytool ユーティリティを使用して暗号キーと信頼された証明書を管理します。このユーティリティは、HP OO のインストールフォルダー (**<インストールディレクトリ>/java/bin/keytool**) に含まれています。Keytool ユーティリティの詳細については、http://docs.oracle.com/javase/7/docs/technotes/tools/solaris/keytool.html を参照してください。

HP OO Central のインストールには、次の 2 つの証明書管理用ファイルが含まれています。

- **<インストールディレクトリ>/central/var/security/client.truststore**: 信頼される証明書のリストが含まれています。
- **<インストールディレクトリ>/central/var/security/key.store**: HP OO 証明書が含まれています。

HP OO を新規にインストールした場合や現在の証明書の有効期限が切れた場合は、HP OO 証明書を置き換えることをお勧めします。

サーバー証明書の認証

Central SSL/TLS サーバー証明書の置き換え

知名度の高い企業によって署名された証明書またはカスタムサーバー証明書を使用することができます。

key.store ファイルやコンピューターの設定に合わせて、<黄色>でハイライトされているパラメーターを置換します。

> **注:** 次の手順は、Keytool ユーティリティ (**<インストールディレクトリ>/java/bin/keytool**) で実行されます。

1. Central を停止し、**<インストールディレクトリ>/central/var/security/key.store** にある **key.store** ファイルをバックアップします。
2. **<インストールディレクトリ>/central/var/security** でコマンドラインを開きます。

3. 次のコマンドを使用して、Central の **key.store** ファイルから既存のサーバー証明書を削除します。

   ```
   keytool -delete -alias tomcat -keystore key.store -storepass changeit
   ```

4. 拡張子が **.pfx** または **.p12** の証明書がすでに存在する場合は、次の手順に進みます。存在しない場合は、秘密キー付きの証明書を PKCS12 形式 (.pfx,.p12) にエクスポートします。たとえば、証明書の形式が PM の場合、次のようになります。

   ```
   >openssl pkcs12 -export -in <cert.pem> -inkey <key.key> -out <証明書名>.p12 -name <名前>
   ```

   証明書の形式が DER の場合、次のように、`-inform DER` パラメーターを pkcs12 の後に追加します。

   ```
   >openssl pkcs12 -inform DER -export -in <cert.pem> -inkey <.key> -out <証明書名>.p12 -name <名前>
   ```

   > **注:** パスワードを記録しておいてください。この秘密キーのパスワードは、後の手順でキーストアのパスフレーズ入力で使用します。

5. 次のコマンドを使用して証明書のエイリアスを抽出します。

   ```
   keytool -list -keystore <証明書名> -v -storetype PKCS12
   ```

   エイリアスが表示されます。次の例では、下から 4 番目の行です。

   ```
   c:\Program Files\Hewlett-Packard\oo-sam\central\var\security>keytool -list -keystore server.pfx -v -storetype PKCS12
   Enter keystore password:

   Keystore type: PKCS12
   Keystore provider: SunJSSE

   Your keystore contains 1 entry

   Alias name: le-775fb32c-269c-499b-bae8-fe7077479ec6
   Creation date: 24/04/2014
   Entry type: PrivateKeyEntry
   Certificate chain length: 2
   ```

6. PKCS12 形式のサーバー証明書を **key.store** ファイルにインポートします。

   keytool -importkeystore -srckeystore <PKCS12 形式の証明書のパス> -destkeystore key.store -srcstoretype pkcs12 -deststoretype JKS -alias <証明書のエイリアス> -destalias tomcat

7. Central サーバーの自動生成されたキーストア内のデフォルトの "changeit" パスワードを変更することをお勧めします。「キーストア/信頼ストアのパスワードの変更」(11ページ) を参照してください。

8. Central を起動します。

Central SSL/TLS サーバー証明書の自己署名証明書での置き換え

Keytool ユーティリティを使用すると自己署名証明書を生成することができます。

注: HP OO 10.10 へアップグレード後に、次を実行します。

- 以前のインストールと同じマシンに Central を新規にインストールする場合は、既存の自己署名証明書を使用できます。
- 別のマシンに Central を新規にインストールする場合は、以前のバージョン用の証明書がある場合でも、インストールごとに新規の自己署名証明書を生成する必要があります。

注: 次の手順は、Keytool ユーティリティ (**<インストールディレクトリ>/java/bin/keytool**) で実行されます。

key.store ファイルやコンピューターの設定に合わせて、<黄色>でハイライトされているパラメーターを置換します。

1. Central を停止し、**<インストールディレクトリ>/central/var/security/key.store** にある **key.store** ファイルをバックアップします。
2. **<インストールディレクトリ>/central/var/security** でコマンドラインを開きます。
3. 次のコマンドを使用して、Central の **key.store** ファイルから既存のサーバー証明書を削除します。

   ```
   keytool -delete -alias tomcat -keystore key.store -storepass <changeit>
   ```

4. 自己署名証明書を生成します。

   ```
   keytool -genkey -alias tomcat -keyalg RSA -keypass <changeit >-keystore <path/for/new/Keystore> -storepass <changeit>-storetype pkcs12 -dname "CN=<CENTRAL_FQDN>, OU=<ORGANIZATION_UNIT>, O=<ORGANIZATION>, L=<LOCALITY>, C=<COUNTRY>"
   ```

   注: 新しいキーストアの生成に使用するパスを入力しないと、コマンドを実行したフォルダー (**<インストールディレクトリ>/central/var/security** など) で作成されます。

5. 自己署名証明書を Central の **key.store** ファイルにインポートします。

   ```
   keytool -v -importkeystore -srckeystore <new/path/created/Keystore> -srcstoretype PKCS12 -srcstorepass <changeit> -destkeystore key.store -deststoretype JKS -deststorepass <changeit>
   ```

6. Central を起動します。

証明書の RAS 信頼ストアへのインポート

RAS のインストール後、Central でカスタムルート証明書を使用し、RAS のインストール時にこのルート証明書を提示しなかった場合、信頼されたルート証明機関 (CA) を RAS **client.truststore** にイン

ポートする必要があります。標準の署名済みルート証明書を使用する場合は、証明書はすでに **client.truststore** ファイルに登録されているので、次の手順を実行する必要はありません。

デフォルトで、HP OO はすべての自己署名証明書をサポートします。ただし、実稼働環境では、セキュリティ上の理由から、このデフォルト設定を変更することをお勧めします。

<黄色>でマークされているパラメーターを置き換えます。

> **注:** 次の手順は、Keytool ユーティリティ (**<インストールディレクトリ>/java/bin/keytool**) で実行されます。

1. RAS を停止し、**<インストールディレクトリ>/ras/var/security/client.truststore** にある元の **client.truststore** ファイルをバックアップします。
2. **<インストールディレクトリ>/ras/var/security** でコマンドラインを開きます。
3. **<インストールディレクトリ> ras/conf/ras-wrapper.conf** ファイルを開き、`-Dssl.support-self-signed` の値を **false** に設定します。これにより、信頼されたルート証明機関 (CA) が有効になります。

   例:

   ```
   wrapper.java.additional.<x>=-Dssl.support-self-signed=false
   ```

4. **<インストールディレクトリ> ras/conf/ras-wrapper.conf** ファイルを開き、`-Dssl.verifyHostName` の値を **true** に設定します。これにより、ホスト名が検証されます。

   例:

   ```
   wrapper.java.additional.<x>=-Dssl.verifyHostName=true
   ```

5. 信頼されたルート証明機関 (CA) を RAS **client.truststore** ファイルにインポートします。

   ```
   keytool -importcert -alias <任意のエイリアス> -keystore client.truststore -file <証明書名.cer> -storepass <changeit>
   ```

6. RAS を起動します。

証明書の OOSH 信頼ストアへのインポート

Central でカスタムルート証明書を使用する場合、信頼されたルート証明機関 (CA) を OOSH **client.truststore** にインポートする必要があります。標準の署名済みルート証明書を使用する場合は、証明書はすでに **client.truststore** ファイルに登録されているので、次の手順を実行する必要はありません。

デフォルトで、HP OO はすべての自己署名証明書をサポートします。ただし、実稼働環境では、セキュリティ上の理由から、このデフォルト設定を変更することをお勧めします。

<黄色>でマークされているパラメーターを置き換えます。

注: 次の手順は、Keytool ユーティリティ (**<インストールディレクトリ>/java/bin/keytool**) で実行されます。

1. Central を停止し、**<インストールディレクトリ>/central/var/security/client.truststore** にある **client.truststore** ファイルをバックアップします。
2. **<インストールディレクトリ>/central/bin** にある **oosh.bat** を編集します。
3. `-Dssl.support-self-signed` の値を **false** に設定します。これにより、信頼されたルート証明機関 (CA) が有効になります。

   例:

   ```
   -Dssl.support-self-signed=false
   ```

4. `-Dssl.verifyHostName` を **true** に設定します。これにより、ホスト名が検証されます。

   例:

   ```
   -Dssl.verifyHostName=true
   ```

5. 信頼されたルート証明機関 (CA) を Central **client.truststore** ファイルにインポートします。

   ```
   keytool -importcert -alias <任意のエイリアス> -keystore client.truststore -file <証明書名.cer> -storepass <changeit>
   ```

6. OOSH を実行します。

証明書の Studio Debugger 信頼ストアへのインポート

Studio のインストール後、Studio でカスタムルート証明書を使用する場合、信頼されたルート証明機関 (CA) を Studio **client.truststore** にインポートする必要があります。標準の署名済みルート証明書を使用する場合は、証明書はすでに **client.truststore** ファイルに登録されているので、次の手順を実行する必要はありません。

デフォルトで、HP OO はすべての自己署名証明書をサポートします。ただし、実稼働環境では、セキュリティ上の理由から、このデフォルト設定を変更することをお勧めします。

<黄色>でマークされているパラメーターを置き換えます。

注: 次の手順は、Keytool ユーティリティ (**<インストールディレクトリ>/java/bin/keytool**) で実行されます。

1. Studio を停止し、**<インストールディレクトリ>/studio/var/security/client.truststore** にある **client.truststore** ファイルをバックアップします。
2. **<インストールディレクトリ>/studio** にある **Studio.l4j.ini** ファイルを編集します。

3. -Dssl.support-self-signed の値を **false** に設定します。これにより、信頼されたルート証明機関 (CA) が有効になります。

   例:

   ```
   -Dssl.support-self-signed=false
   ```

4. -Dssl.verifyHostName を **true** に設定します。これにより、ホスト名が検証されます。

   例:

   ```
   -Dssl.verifyHostName=true
   ```

5. 信頼されたルート証明機関 (CA) を Studio **client.truststore** ファイルにインポートします。

   ```
   keytool -importcert -alias <任意のエイリアス> -keystore client.truststore -file <証明書名.cer> -storepass <changeit>
   ```

6. Studio を起動します。

詳細については、『Studio オーサリングガイド』の「リモート Central の Studio でのデバッグ」を参照してください。

キーストア/信頼ストアのパスワードの変更

- **Central のパスワードを変更するには、次の手順を実行します。**

  a. **<インストールディレクトリ>/central/tomcat/conf/server.xml** にある **server.xml** ファイルを編集します。

  b. HTTPS コネクターを検索します。例:

  ```
  keyPass="changeit" keystoreFile="C:/Program Files/Hewlett-Packard/HP Operations Orchestration/central/var/security/key.store" keystorePass="changeit" keystoreType="JKS" maxThreads="200" port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol" scheme="https" secure="true" sslProtocol="TLSv1.2" truststoreFile="C:/Program Files/Hewlett-Packard/HP Operations Orchestration/central/var/security/client.truststore" truststorePass="changeit" truststoreType="JKS"/>
  ```

  c. パスワードを変更します。

  - keyPass - キーストアファイルのサーバー証明書にアクセスする際に使用するパスワード。デフォルト値は "changeit" です。
  - keystorePass - キーストアファイルへのアクセスに使用するパスワード。デフォルト値は keyPass 属性の値です。

- `truststorePass` - 信頼ストアへのアクセスに使用するパスワード。デフォルト値は **javax.net.ssl.trustStorePassword** システムプロパティの値です。プロパティの値が null の場合、信頼ストアのパスワードは設定されません。信頼ストアのパスワードに無効な値が指定されると、警告がログに記録され、パスワードなしで信頼ストアにアクセスします。信頼ストアの内容の検証は省略されます。

d. ファイルを保存します。

e. **central/conf** にある **central-wrapper.conf** を開き、次のように変更します。

```
wrapper.java.additional.<x>=-Djavax.net.ssl.trustStorePassword=changeit
```

f. Central を再起動します。

- **RAS 信頼ストアのパスワードを変更するには、次の手順を実行します。ras-wrapper.conf** ファイルを編集し、信頼ストアの `changeit` パラメーターを変更します。
- **OOSH 信頼ストアのパスワードを変更するには、次の手順を実行します。oosh.bat** ファイルを編集し、信頼ストアの `changeit` パラメーターを変更します。
- **Studio 信頼ストアのパスワードを変更するには、次の手順を実行します。<インストールディレクトリ>/studio/Studio.l4j.ini** ファイルを編集し、信頼ストアの `changeit` パラメーターを変更します。

SSL サポート対象サイファーからの RC4 サイファーの削除

リモートホストは、RC4 暗号の使用をサポートしています。この暗号は、バイトの擬似乱数ストリームの生成処理に欠陥があるため、ストリームに多様で軽微な偏りが生じ、そのランダム性が低下します。

プレーンテキストを繰り返し暗号化するときに (たとえば、HTTP Cookie など)、攻撃者が数多く(数千万) の暗号化テキストを入手できる場合、攻撃者はプレーンテキストを推測できることがあります。

JRE レベルでRC4 暗号を無効にします (Java 7 以降)。

1. **$JRE_HOME/lib/security/java.security** ファイルを開きます。
2. **jdk.tls.disabledAlgorithms** プロパティを編集して、RC4 暗号を無効にします。

詳細については、http://stackoverflow.com/questions/18589761/restict-cipher-suites-on-jre-level を参照してください。

HTTP/HTTPS ポートの変更とクローズ

[OO_HOME]\central\Tomcat\conf の下の **server.xml** ファイルには、**<Service>** 要素の下に **<Connector>** という名前の要素が 2 つあります。これらのコネクターでは、サーバーがリッスンしているポートを定義または有効にします。

各コネクターの構成は、それぞれの属性を使用して定義します。最初のコネクターでは通常の HTTP コネクターを定義し、2 番目のコネクターでは HTTPS コネクターを定義します。

デフォルトで、これらのコネクターは次のようになります。

HTTP コネクター:

```
<Connector URIEncoding="UTF-8" compression="on" connectionTimeout="20000" po
rt="8080" protocol="org.apache.coyote.http11.Http11NioProtocol" redirectPort
="8443"/>
```

HTTPS コネクター:

```
<Connector SSLEnabled="true" URIEncoding="UTF-8" clientAuth="false" compress
ion="on" keyAlias="tomcat" keyPass="changeit" keystoreFile="C:/Program Files
/Hewlett-Packard/HP Operations Orchestration/central/var/security/key.store"
keystorePass="changeit" keystoreType="JKS" maxThreads="200" port="8443" prot
ocol="org.apache.coyote.http11.Http11NioProtocol" scheme="https" secure="tru
e" sslProtocol="TLSv1.2" truststoreFile="C:/Program Files/Hewlett-Packard/HP
Operations Orchestration/central/var/security/client.truststore" truststoreP
ass="changeit" truststoreType="JKS"/>
```

デフォルトでは、両方とも有効です。

ポートの値の変更

いずれかのポートの値を変更するには、次の手順を実行します。

1. **<インストールディレクトリ>/central/tomcat/conf/server.xml** にある **server.xml** ファイルを編集します。
2. HTTP または HTTPS コネクターを探し、**port** の値を変更します。

   注: HTTP とHTTPS を両方使用する場合にHTTPS ポートを変更するには、HTTP コネクターの **redirectPort** を変更します。

3. ファイルを保存します。
4. Central を再起動します。

ポートの無効化

いずれかのポートを無効にするには、次の手順を実行します。

1. **<インストールディレクトリ>/central/tomcat/conf/server.xml** にある **server.xml** ファイルを編集します。
2. HTTP または HTTPS コネクターを探し、その行を削除またはコメント行にします。
3. ファイルを保存します。
4. Central を再起動します。

クライアント証明書の認証 (相互認証)

X.509 証明書認証は、SSL/TLS を使用するサーバーの ID 検証によく使用され、特にブラウザーで HTTPS を使用する場合です。ブラウザーは、サーバーが提示する証明書が、信頼される証明機関リストに含まれる証明機関が発行したものかどうかを自動的にチェックします。

SSL/TLS を相互認証で使用することもできます。サーバーは、SSL/TLS ハンドシェイクにおいて、クライアントに有効な証明書を要求します。サーバーは、証明書が適切な証明機関によって署名されていることをチェックし、クライアントを認証します。有効な証明書が提供されている場合には、アプリケーション内のサーブレット API を使用して取得できます。

クライアント証明書認証の構成 (Central)

Central でクライアント証明書認証を構成する前に、「サーバーおよびクライアント証明書の認証」(6 ページ) の手順に従って SSL/TLS サーバー証明書を構成しておく必要があります。

接続を確立する前に、SSL スタックがクライアントに有効な証明書チェーンを要求する場合は、`clientAuth` 属性を `true` に設定します。SSL スタックはクライアント証明書を要求するが、提示されなくてもエラーにしない場合は、`want` に設定します。`false` (デフォルト) に設定すると、CLIENT-CERT 認証を使用するセキュリティ制限で保護されているリソースをクライアントが要求した場合を除き、証明書チェーンは要求されなくなります。詳細については、『Apache Tomcat Configuration Reference』を参照してください。

証明書失効リスト (CRL) ファイルを設定します。CRL は複数存在することがあります。暗号化システムでは一般的に公開キーインフラストラクチャー (PKI) が使用され、証明書失効リスト (CRL) には無効な証明書のリスト (具体的には、証明書のシリアル番号) が格納されています。したがって、ここに含まれる証明書を提示したエンティティは信頼できないエンティティということになります。

> **注:** 次の手順は、Keytool ユーティリティ (**<インストールディレクトリ>/java/bin/keytool**) で実行されます。

1. Central サーバーを停止します。
2. 適切なルート証明書 (CA) を Central の **client.truststore**: **<インストールディレクトリ>/central/var/security/client.truststore** にインポートします。次に例を示します。

   ```
   keytool -importcert -alias <任意のエイリアス> -keystore <パス>/client.truststore
   -file <証明書のパス> -storepass <changeit>
   ```

3. **<インストールディレクトリ>/central/tomcat/conf/server.xml** にある **server.xml** ファイルを編集します。
4. `Connector` タグの `clientAuth` 属性を `want` または `true` に変更します。デフォルトは `false` です。

   > **注:** この手順が終わってからサーバーを起動することをお勧めしますが、この時点でサーバー

を起動することもできます。

5. crlFile 属性を追加し、SSL/TLS 証明書の検証に使用する CRL を定義します。次に例を示します。

```
crlFile="<パス>/crlname.<crl/pem>"
```

ファイルの拡張子が .crl の場合は CRL が 1 つ、 .pem (PEM CRL 形式) の場合は CRL が複数含まれています。PEM CRL 形式では、次のようなヘッダー行とフッター行を使用します。

```
-----BEGIN X509 CRL-----
-----END X509 CRL-----
```

CRL を 1 つ含む .pem ファイルの例を示します (複数の場合、CRL ブロックを連結していきます)。

```
-----BEGIN X509 CRL-----
MIIBbzCB2QIBATANBgkqhkiG9w0BAQUFADBeMQswCQYDVQQGEwJVUzEYMBYGA1UE
ChMPVS5TLiBHb3Zlcm5tZW50MQwwCgYDVQQLEwNEb0QxEDAOBgNVBAsTB1Rlc3Rp
bmcxFTATBgNVBAMTDFRydXN0IEFuY2hvchcNOTkwMTAxMTIwMTAwWhcNNDgwMTAx
MTIwMTAwWjAiMCACAScXDTk5MDEwMTEyMDAwMFowDDAKBgNVHRUEAwoBAaAjMCEw
CgYDVR0UBAMCAQEwEwYDVR0jBAwwCoAIq5rr+cLnVI8wDQYJKoZIhvcNAQEFBQAD
gYEAC7lqZwejJRW7QvzH11/7cYcL3racgMxH3PSU/ufvyLk7ahR++RtHary/WeCv
RdyznLiIOA8ZBiguWtVPqsNysNn7WLofQIVa+/TD3T+lece4e1NwGQvj5Q+e2wRt
GXg+gCuTjTKUFfKRnWz7O7RyiJKKim0jtAF4RkCpLebNChY=
-----END X509 CRL-----
```

6. Central サーバーを起動します。

注: クライアント証明書ごとに、ユーザー (内部ユーザーまたは LDAP ユーザー) を定義します。ユーザー名は、証明書属性で定義する必要があります。デフォルトは、CN 属性の値です。詳細については、「証明書のプリンシパルの処理」を参照してください。

HP OO で LDAP 構成を複数設定しても、ユーザー認証に使用できるのは、デフォルト LDAP のクライアント証明書属性のみです。Central は、まずデフォルトの LDAP でユーザー認証を行い、失敗すると、HP OO 内部ドメインで認証を行います。

クライアント証明書の構成の更新 (RAS)

クライアント証明書は、RAS のインストール時に構成されます。ただし、クライアント証明書の更新が必要な場合は、**ras-wrapper.conf** ファイルを手動で編集します。

事前確認: Central の CA ルート証明書を RAS 信頼ストアにインポートする必要があります。「証明書の RAS 信頼ストアへのインポート」(8ページ) を参照してください。

外部 RAS でクライアント証明書を更新するには、次の手順を実行します。

1. RAS サーバーを停止します。
2. **<インストールディレクトリ>ras/var/conf/ras-wrapper.conf** にある **ras-wrapper.conf** ファイルを開きます。
3. クライアント証明書に基づいて次の変更を行います。

```
wrapper.java.additional.<x>=-Djavax.net.ssl.keyStore=<インストールディレクトリ>/var/security/certificate.p12"

wrapper.java.additional.<x>=-Djavax.net.ssl.keyStorePassword=changeit

wrapper.java.additional.<x>=-Djavax.net.ssl.keyStoreType=PKCS12
```

4. RAS サーバーを起動します。

重要 X.509 クライアント証明書には、RAS のプリンシパル名が必要です。これは、RAS ID です(「証明書のプリンシパルの処理」を参照してください)。

RAS ID は、Central の **[トポロジ]** タブで確認できます。『HP OO ユーザーガイド』の「トポロジのセットアップ – ワーカー」を参照してください。

Studio リモートデバッガーでのクライアント証明書の構成

事前確認: Central の CA ルート証明書を Studio Debugger 信頼ストアにインポートする必要があります。「証明書の Studio Debugger 信頼ストアへのインポート」(10ページ) を参照してください。

Studio リモートデバッガーでクライアント証明書を構成するには、次の手順を実行します。

1. Studio を閉じます。
2. **<インストールディレクトリ>/studio** にある **Studio.l4j.ini** を編集します。
3. クライアント証明書に基づいて次の変更を行います。

```
-Djavax.net.ssl.keyStore="<インストールディレクトリ>/studio/var/security/certificate.p12"

-Djavax.net.ssl.keyStorePassword=changeit

-Djavax.net.ssl.keyStoreType=PKCS12
```

4. Studio を起動します。

注: クライアント証明書で使用するユーザー (内部ユーザーまたは LDAP ユーザー) を定義します。ユーザー名は、証明書属性で定義する必要があります。デフォルトは、CN 属性の値です。詳細については、「証明書のプリンシパルの処理」を参照してください。

HP OO で LDAP 構成を複数設定しても、ユーザー認証に使用できるのは、デフォルト LDAP のクライアント証明書属性のみです。Central は、まずデフォルトの LDAP でユーザー認証を行い、失敗すると、HP OO 内部ドメインで認証を行います。

OOSH でのクライアント証明書の構成

事前確認: Central の CA ルート証明書を OOSH 信頼ストアにインポートする必要があります。「証明書の OOSH 信頼ストアへのインポート」(9ページ) を参照してください。

1. OOSH を停止します。
2. **<インストールディレクトリ>/central/bin** にある **oosh.bat** を編集します。
3. クライアント証明書に基づいて次の変更を行います。

```
-Djavax.net.ssl.keyStore="<インストールディレクトリ>/var/security/certificate.p12"
-Djavax.net.ssl.keyStorePassword=changeit
-Djavax.net.ssl.keyStoreType=PKCS12
```

4. OOSH を起動します。

注: クライアント証明書で使用するユーザー (内部ユーザーまたは LDAP ユーザー) を定義します。ユーザー名は、証明書属性で定義する必要があります。デフォルトは、CN 属性の値です。詳細については、「証明書のプリンシパルの処理」を参照してください。

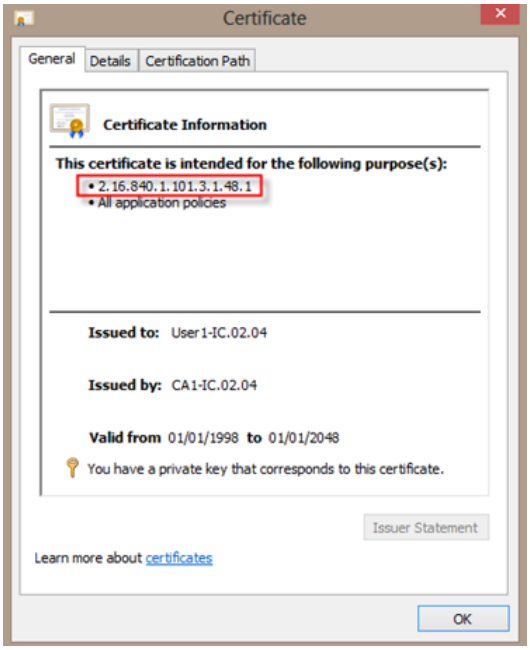
HP OO で LDAP 構成を複数設定しても、ユーザー認証に使用できるのは、デフォルト LDAP のクライアント証明書属性のみです。Central は、まずデフォルトの LDAP でユーザー認証を行い、失敗すると、HP OO 内部ドメインで認証を行います。

証明書ポリシーの処理

HP OO は、エンドポイントの証明書に適用する証明書ポリシーを処理します。

- 証明書では、使用目的を示す文字列を設定できます。
- HP OO では、ポリシー文字列を構成アイテムとして追加し、エンドポイントの証明書ごとにポリシー文字列をチェックすることができます。一致しないと、証明書は却下されます。
- 証明書ポリシーの検証を有効または無効にするには、次の構成アイテムを追加します。`x509.certificate.policy.enabled=true/false` (デフォルトは `false`)
- 次の構成アイテムを追加して、ポリシーリストを定義します。`x509.certificate.policy.list=<`

カンマ区切りのリスト> (デフォルトは空のリスト)。



証明書のプリンシパルの処理

Subject に対する正規表現を使用して、証明書からプリンシパルを取得する方法を定義できます。正規表現には、単一のグループを指定します。デフォルトの式は CN=(.?) であり、一般的な名前フィールドに一致します。たとえば CN=Jimi Hendrix, OU= は、Jimi Hendrix というユーザー名に一致します。

- 一致の比較では、大文字と小文字を区別します。
- 証明書のプリンシパルは、HP OO のユーザー名です (LDAP または内部ユーザー)。
- 正規表現を変更するには、次の構成アイテムを変更します。x509.subject.principal.regex

トラブルシューティング

サーバーが起動しない場合は、**wrapper.log** ファイルを開いて、ProtocolHandler ["http-nio-8443"] でエラーを確認します。

これは Tomcat でコネクターを初期化または起動する際に発生します。さまざまなバリエーションがありますが、エラーメッセージから情報を得ることができます。

HTTPS コネクターのパラメーターはすべて **C:\HP\oo\central\tomcat\conf\server.xml** にある Tomcat 構成ファイル内にあります。

ファイルを開いて下にスクロールし、HTTPS コネクターを確認します。

```
<Connector SSLEnabled="true" clientAuth="false" keyAlias="tomcat" keystoreFile="
C:/HP/oo/central/var/security/keystore.p12" keystorePass="tomcat-keystore-passwo
rd" keystoreType="PKCS12" maxThreads="200" port="8443" protocol="org.apache.coyo
te.http11.Http11NioProtocol" scheme="https" secure="true" sslProtocol="TLSv1.
2"/>
```

前 のステップで入 力したパラメーターと比 較して、一 致しないパラメーターがないかどうかを確 認しま す。

Federal Information Processing Standard (FIPS)

HP OO での FIPS 140-2 互換の構成

このセクションでは、HP Operations Orchestration を Federal Information Processing Standards (FIPS) 140-2 互換に構成する手順を説明します。

FIPS 140-2 は、暗号化モジュールに適用されるセキュリティ要件の標準であり、National Institute of Standards Technology (NIST) によって規定されています。標準の規定の内容は、次で参照できます。csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf.

HP OO で FIPS 140-2 互換の構成を行うと、HP OO は次のセキュリティアルゴリズムを使用します。

- 対称キーアルゴリズム: AES
- ハッシュアルゴリズム: SHA1

HP OO が使用するセキュリティプロバイダーは、RSA BSAFE Crypto ソフトウェアバージョン 6.1 です。これは、FIPS 140-2 でサポートされる唯一のセキュリティプロバイダーです。

注: HP OO で FIPS 140-2 互換構成が完了すると、標準構成に戻すことはできません。戻すには、HP OO の再インストールが必要です。

前提条件

注: FIPS ですでに構成された HP OO 10.10 (以降) のインストールからアップグレードしている場合は、次の手順 4 と 5 を繰り返してから、「HP OO での FIPS 140-2 互換の構成」(22ページ) の「Java セキュリティファイルのプロパティ構成」の手順を繰り返す必要があります。

HP OO で FIPS 140-2 互換構成を行う前は、次の手順を実行します。

注: FIPS140-2 互換の構成には、LWSSO を無効にする必要があります。

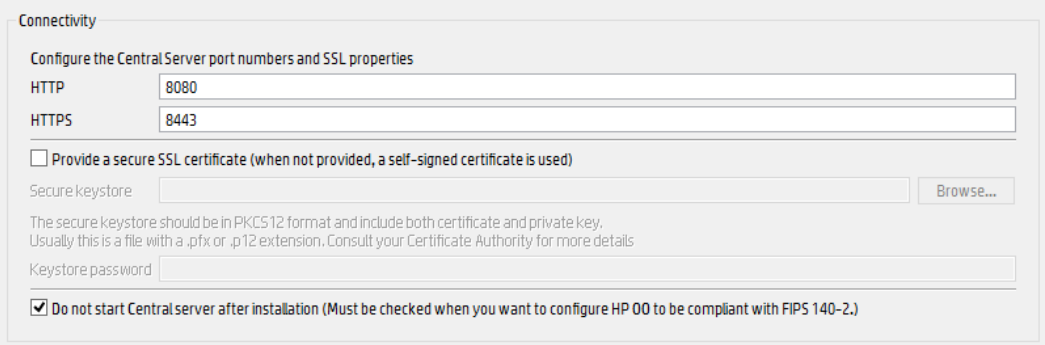
1. FIPS 140-2 互換構成には、HP OO バージョン 10.10 以降の新規インストールが必要です。

   インストール済みの HP OO (バージョン 9.x または 10.x を問わず) は使用できません。

2. HP OO のインストール時に、インストール後に Central サーバーを起動しないように設定されていることを確認します。

   - サイレントインストールでは、`should.start.central` パラメーターは [**No**] に設定されます。
   - ウィザードの [**Connectivity**] 手順で、[**Do not start Central server after installation**]

チェックボックスを選択します。



3. 次のディレクトリをバックアップします。
   - **<インストールディレクトリ>\central\tomcat\webapps\oo.war**
   - **<インストールディレクトリ>\central\tomcat\webapps\PAS.war**
   - **<インストールディレクトリ>\central\conf**
   - **<oo_jre>\lib\security** (**<oo_jre>** は、HP OO が使用する JRE のインストール先。デフォルトディレクトリは **<インストールディレクトリ>\java**)
4. Java Cryptographic Extension (JCE) 無制限強度管轄ポリシーファイルを次のサイトからダウンロードおよびインストールします。http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html.

   注: ファイルのデプロイと HP OO で使用する JRE のアップグレードの手順は、ダウンロードした **ReadMe.txt** ファイルを参照してください。
5. RSA BSAFE Crypto ソフトウェアファイルをインストールします。HP OO がインストールされているシステムで、次のファイルを **<oo_jre>\lib\ext** (**<oo_jre>** は、HP OO が使用する JRE のインストール先。デフォルトは **<インストールディレクトリ\java**) にコピーします。
   - **<インストールディレクトリ>\central\lib\cryptojce-6.1.jar**
   - **<インストールディレクトリ>\central\lib\cryptojcommon-6.1.jar**
   - **<インストールディレクトリ>\central\lib\jcmFIPS-6.1.jar**

注: FIPS ですでに構成された HP OO 10.10 (以降) のインストールからアップグレードしている場合は、前の「前提条件」の手順 4 と 5 を繰り返してから、「HP OO での FIPS 140-2 互換の構成」(22ページ) の「Java セキュリティファイルのプロパティ構成」の手順を繰り返す必要があります。

HP OO での FIPS 140-2 互換の構成

FIPS 140-2 との互換性を維持するために HP OO で必要な構成手順を示します。

- Java セキュリティファイルのプロパティ構成
- encryption.properties ファイルの構成と FIPS モードの有効化
- FIPS 互換の HP OO 暗号化の作成
- データベースパスワードの変更
- HP OO の起動

Java セキュリティファイルのプロパティ構成

JRE で使用する Java セキュリティファイルを編集してセキュリティプロバイダーを追加し、FIPS 140-2 互換のプロパティを構成します。

注: HP OO 10.10 にアップグレードすると、インストール済みの JRE ファイルは完全に置換されます。したがって、10.10 へのアップグレードが完了したら、次の手順を実行してください。

注: FIPS ですでに構成された HP OO 10.10 (以降) のインストールからアップグレードしている場合は、「Federal Information Processing Standard (FIPS)」(20ページ) の「前提条件」の手順 4 と 5 を繰り返してから、ここの手順を繰り返す必要があります。

エディターで **<oo_jre>\lib\security\java.security** ファイルを開き、次の手順を実行します。

1. プロバイダーごとに (**security.provider.<nn>=<プロバイダー名 >** という形式)、プリファレンス順序の数値 `<nn>` を 2 つずつ増やします。

   たとえば、次のようなプロバイダーエントリがある場合、次のように変更します。

   ```
   security.provider.1=sun.security.provider.Sun
   ```

   変更後

   ```
   security.provider.3=sun.security.provider.Sun
   ```

2. 新しいデフォルトプロバイダー (RSA JCE) を追加します。次のプロバイダーをリストの一番上に追加します。

   ```
   security.provider.1=com.rsa.jsafe.provider.JsafeJCE
   ```

3. RSA BSAFE SSL-J Java Secure Sockets Extension (JSSE) Provider を追加します。

   ```
   security.provider.2=com.rsa.jsse.JsseProvider
   ```

4. 次の行を **java.security** ファイルに貼り付けます。これにより、**RSA BSAFE** が FIPS 140-2 互換モードで使用されます。

   ```
   com.rsa.cryptoj.fips140initialmode=FIPS140_SSL_MODE
   ```

   この行は、**java.security** ファイル内の任意の場所に貼り付けることができます。

5. デフォルトの DRBG アルゴリズム ECDRBG128 は安全性が低いので (NIST の報告)、セキュリティプロパティ **com.rsa.crypto.default** を **HMACDRBG** に設定します。設定には、次の行を **java.security** ファイルにコピーしてください。

   ```
   com.rsa.crypto.default.random=HMACDRBG
   ```

   この行は、**java.security** ファイル内の任意の場所に貼り付けることができます。

6. **java.security** ファイルを保存してから閉じます。

encryption.properties ファイルの構成と FIPS モードの有効化

HP OO 暗号化プロパティファイルで、FIPS 140-2 互換の設定を行います。

1. **encryption.properties** ファイルをバックアップします。このファイルは **<インストールディレクトリ>\central\var\security** にあります。
2. **encryption.properties** ファイルをテキストエディターで開きます。たとえば、次の行を編集します。

   C:\Program Files\Hewlett-Packard\HP Operations Orchestration\central\var\security\encryption.properties.

3. `keySize=128` を探して、`keySize=256` に変更します。
4. `secureHashAlgorithm=SHA1` を探して、`secureHashAlgorithm=SHA256` に変更します。
5. `FIPS140ModeEnabled=false` を探して、`FIPS140ModeEnabled=true` に変更します。

   注: `FIPS140ModeEnabled=false` が存在しない場合、`FIPS140ModeEnabled=true` を新しくファイルの末尾に追加します。

6. ファイルを保存してから閉じます。

FIPS 互換の HP OO 暗号化の作成

FIPS 互換の設定には、HP OO 暗号化ストアファイルの作成または置換が必要です。手順は、「FIPS 暗号化の置き換え」(24ページ)を参照してください。

注: AES のキー長は、NIST SP800-131A によって 128/192/256 が承認されています。

FIPS では、安全なハッシュアルゴリズムとして、SHA1、SHA256、SHA384、SHA512 がサポートされています。

注: key.store (およびその秘密キーエントリ) と信頼ストアのパスワードを変更することをお勧めします。「キーストア/信頼ストアのパスワードの変更」(11ページ) を参照してください。

注: HP OO 信頼ストアからデフォルトの CA ルート証明書をすべて削除することをお勧めします (client.truststore は <インストール>/central/var/security にあります)。

データベースパスワードの変更

「データベースパスワードの変更」(28ページ) の手順に従って、データベースパスワードを変更します。

HP OO の起動

『HP OO インストールガイド』の説明に従って、HP OO を起動します。

FIPS 暗号化の置き換え

HP OO、Central、および RAS は、機密データや重要データを保護するための暗号ベースのセキュリティシステムを指定する際に、連邦機関で使用する技術要件を定めた Federal Information Processing Standard 140-2 (FIPS 140-2) に準拠しています。

HP OO 10.10 を新規にインストールした場合、FIPS 暗号化アルゴリズムを変更することができます。

注: この手順は、新規インストール専用です。アップグレードで実行することはできません。

Central での FIPS 暗号化アルゴリズムの変更

1. **<Central インストールフォルダー>/var/security** に移動します。
2. **encryption_repository** ファイルをバックアップして削除します。
3. **<Central インストールフォルダー>/bin** に移動します。
4. **generate-keys** スクリプトを実行します。

   新しいマスターキーが、**<Central インストールフォルダー>/var/security/encryption_repository** に生成されます。

RAS 暗号化プロパティの変更

RAS を新しい場所にインストールする場合、次の手順を実行します。

注: 以下の変更内容が有効になるのは、Central 暗号化プロパティの変更後に新しくRAS インストールを行う場合のみです。

RAS 暗号化プロパティを変更するには、次の手順を実行します。

1. 「Federal Information Processing Standard (FIPS)」(20ページ) の「前提条件」の手順をすべて実行します。
2. 「HP OO での FIPS 140-2 互換の構成」(22ページ) の「Java セキュリティファイルのプロパティの校正」の手順をすべて実行します。
3. 現在の **encryption.properties** ファイルを、<インストールディレクトリ>\ras\var\security フォルダーから <インストールディレクトリ>\ras\bin フォルダーにコピーします。
4. テキストエディターで **encryption.properties** ファイルを開き、必要な変更を行います。

   詳細は、「HP OO での FIPS 140-2 互換の構成」(22ページ) の「encryption.properties ファイルの構成とFIPS モードの有効化」を参照してください。
5. 変更内容を保存します。
6. <インストールディレクトリ>\ras\bin フォルダーでコマンドラインプロンプトを開きます。
7. **oosh.bat** を実行します。
8. 次の OOShell コマンドを実行します。`replace-encryption --file encryption.properties`

   注: encryption.properties ファイルを別のフォルダーにコピーした場合は、OOShell コマンドの場所を正しく指定してください。
9. RAS サービスを再起動します。

LW SSO 設定の構成

HP OO 10.10 をインストールする際に、LW SSO 設定を HP OO 9.x からアップグレードするように選択した場合、その LW SSO 設定は移行されますが、HP OO 10.10 では LW SSO が無効になります (HP OO 9.x で有効になっていた場合でも無効になります)。

その後 LW SSO を有効にすると、一定のシナリオで警告が生成されることがあります。ログの警告をクリアするには、次の手順に従い、完全修飾ドメイン名を使用して管理 URL プロパティを設定します。

- Central とRAS が同じマシン上にインストールされ、LW SSO 設定が有効になっている場合は、完全修飾ドメイン名を使用して、管理 URL プロパティを設定する必要があります。

a. RAS プロセスを停止します。

b. **ras/conf/ras-wrapper.conf** ファイルで次の行を変更します。

```
wrapper.java.additional.<x>=-Dmgmt.url=<プロトコル>://localhost:<ポート>/oo
```

変更後

```
wrapper.java.additional.<x>=-Dmgmt.url=<プロトコル>://<完全修飾ドメイン名>:<ポート>/oo
```

c. RAS プロセスを起動します。

- RAS が Central とは別のマシンにインストールされ、LW SSO 設定が有効になっている場合は、RAS のインストール中に、完全修飾ドメイン名を使用して Central の管理 URL を設定する必要があります (IP アドレスは使用できません)。
- LW SSO を介して別のアプリケーションを Central に接続している場合は、完全修飾ドメイン名を使用して Central の管理 URL を設定する必要があります

a. Central プロセスを停止します。

b. **central/conf/central-wrapper.conf** ファイルで次の行を変更します。

```
wrapper.java.additional.<x>=-Dmgmt.url=<プロトコル>://localhost:<ポート>/oo
```

変更後

```
wrapper.java.additional.<x>=-Dmgmt.url=<プロトコル>://<完全修飾ドメイン名>:<ポート>/oo
```

c. Central プロセスを起動します。

XSS ポリシーの構成

HP OO は、AntiSamy による XSS 保護機能を備えています。デフォルトの保護ポリシーは "antisamy" で、ほとんどの HTML 要素が許可され、ユーザーが HTML ページ全体を送信する場合に便利です。

このポリシーは、AntiSamy でサポートされるポリシーのいずれか 1 つに設定できます。詳細については、次を参照してください。

https://www.owasp.org/index.php/Category:OWASP_AntiSamy_Project#Stage_2_-_Choosing_a_base_policy_file

ポリシーは `xss.policy` というシステム構成プロパティを使用して設定できます。指定できる値には、antisamy (デフォルト)、antisamy-slashdot、antisamy-myspace、antisamy-ebay、antisamy-anythinggoes、antisamy-tinymc があります。

ポリシーの設定を確認するには、**https://host/oo/reports/sysinfo** にアクセスして、**システム構成** セクションでパラメーター `xss.policy` を確認します。

デフォルトの Slashdot ポリシーを変更する最も簡単な方法は、HP Operations Orchestration Shell ユーティリティを使用することです。

1. `oosh.bat` バッチファイルをダブルクリックし、OOSH ユーティリティを起動します。
2. コマンドラインで次のように入力します (例)。

```
ssc --url https://host/oo --key  xss.policy --value antisamy-anythinggoes
```

HP Operations Orchestration Shell ユーティリティの詳細については、『HP Operations Orchestration Shell User Guide』を参照してください。

ローカライズの構成

Central-wrapper.conf でのシステムロケールの設定

HP OO システムがローカライズされている場合、**central-wrapper.conf** ファイルで次のプロパティを設定してシステムロケールを反映する必要があります。

```
set.LANG=

set.LC_ALL=

set.LANGUAGE=

wrapper.java.additional.<x>=-Dssl.verifyHostName=true

wrapper.java.additional.<x>=-Dssl.verifyHostName=true
```

たとえば、日本語の場合: `set.LANG=ja_JP` および `set.LC_ALL=ja_JP`

システムの構 成

データベースパスワードの変 更

1. Central が実 行 中 である場 合 、Central サービスを停 止します。
2. -e -p <パスワード> オプションを指 定して、encrypt-password スクリプトを実 行します。ここでパス ワード はデータベースのパスワード です。
3. 次 のような結 果 が表 示されるので、コピーします。

   ```
   ${ENCRYPTED}<文 字 列>
   ```

4. **<Central インストールフォルダー>/conf** フォルダーに移 動して **database.properties** ファイルを開 きます。
5. db.password の値 を、コピーした内 容 に変 更します。

データベース IP の変 更

このセクションでは、他 のデータベースインスタンスを使 用 する場 合 に必 要 になる HP OO の構 成 につ いて説 明します。データベースパラメーター (データベース資 格 情 報 、スキーマ名 、テーブルなど) はすべ て一 意 である必 要 があります。

1. **\HP Operations Orchestration\central\conf\database.properties** ファイルを編 集します。
2. jdbc.url パラメーターを探します。例 :

   ```
   jdbc.url=jdbc\:jtds\:sqlserver\://16.60.185.109\:1433/schemaName;sendStringP
   arametersAsUnicode\=true
   ```

3. データベースサーバーの IP アドレス\FQDN を変 更します。
4. ファイルを保 存します。
5. Central を再 起 動します。

ログレベルの調 整

ログに記 録される情 報 の詳 細 度 は、通 常 のログ、デプロイメント、実 行 についてそれぞれ個 別 に調 整 できます。

詳 細 度 のオプションは、次 のとおりです。

- INFO - デフォルト のログ情 報
- DEBUG - より詳 細 なログ情 報

- ERROR/WARNING - より簡潔なログ情報

ログの詳細度を調整するには、次の手順を実行します。

1. **/<oo-installation>/central/conf/log4j.properties** の **log4j.properties** ファイルを開きます。
2. **log4j.properties** ファイルの次の場所で、INFO を DEBUG または ERROR/WARNING に置換します。

   例:

```
log.level=INFO
execution.log.level=DEBUG
deployment.log.level=DEBUG
```

Quartz ジョブのタイミング調整

HP OO システムでは、Quartz ジョブが定期的に実行され、システムのメンテナンスを行います。

ジョブは、設定された間隔で反復的に実行されます。次に、ジョブのトリガーの例を示します。

トリガー名	現在の反復間隔	処理
onRolling:OO_EXECUTION_STATES_Trigger	4.5 分	削除するステータステーブルの切り替え
queueCleanerTrigger	1 分	キューテーブルの削除
queueRecoveryTrigger	2 分	システムの復元が必要かどうかをチェック
recoveryVersionTrigger	0.5 分	復元に使用するバージョンカウンター
splitJoinTrigger	1 秒	終了したスプリットのジョイン
onRolling:OO_EXECUTION_EVENTS_Trigger **注:** このトリガーは非推奨となっています。	12 時間	削除するイベントテーブルの切り替え

パフォーマンスを向上する目的でジョブのタイミングを調整するんは、次の手順を実行します。

注: タイミングを変更するとシステムに大きな影響を及ぼすことがあります。HP サービス担当者に確認の上、変更してください。

1. Jminix ページを次の URL で入力します: **{OO_HOST}:{OO_PORT}/oo/jminix/**

   注: jminix の入力には、**システム設定の管理** アクセス許可が必要です。

2. [OO] タブを開きます。[**MBeans**] の下に、**jobTriggersMBean** というオペレーションがあります。
3. このオペレーションで、右のタブに値を入力します。変更したいトリガーの名前を指定してください。テーブルと同一の名前と、反復間隔を指定します。

これにより、ジョブがトリガーされるタイミングが変更されます。

注: イベントの持続性メカニズムは非推奨となっています (**onRolling:OO_EXECUTION_EVENTS_Trigger** を参照)。このジョブを構成できるのは、リモートデバッガーを使用するか、**events.persistency** フラグをオフにした場合です。「イベントログ機能の有効化」(30ページ) を参照してください。

RAS での Central/ロードバランサーの URL 変更

Central/ロードバランサーの URL はインストーラーを使用して構成することがベストプラクティスですが、RAS のインストール後に変更する場合は、**ras-wrapper.conf** ファイルを編集する方法もあります。

この手順が必要になるのは、たとえば RAS を Central/ロードバランサーにインストールし、Central/ロードバランサーの FQDN を変更した場合があります。RAS が Central/ロードバランサーと通信するには、RAS レベルで保存されている Central/ロードバランサーの URL の変更が必要です。

1. RAS 停止します。
2. **<インストールフォルダー>\ras\conf** にある **ras-wrapper.conf** ファイルを開きます。
3. 次の URL を編集します。

   ```
   wrapper.java.additional.<x>=-Dmgmt.url=http://localhost:8080/oo
   ```

4. RAS を再起動します。

イベントログ機能の有効化

イベントログ機能は HP OO 10.10 で非推奨となり、今後のリリースで削除される予定です。

HP OO はイベントログ機能なしでデプロイされていますが、このメカニズムを採用すると、**events.persistency** フラグを有効化できます。クラスター化されたシナリオやパフォーマンスの向上のために、このフラグを無効化することをお勧めします。

フラグを有効にする方法:

1. プロセスを停 止し、システムのすべてのノード (Central/RAS) の **wrapper.conf** ファイルを更 新しま す。

   Central で、**<インストールパス>\central\conf\central-wrapper.conf** に移 動します。

   RAS で、**<インストールパス>\ras\conf\ras-wrapper.conf** に移 動します。

2. **wrapper.conf** ファイル内 で**-Devents.persistency** パラメーターを探し、値 を **true** に変 更しま す。

3. サーバーを再 起 動します。


Java™
COMPATIBLE