HP Operations Orchestration

ソフトウェアバージョン: 10.20
Windows および Linux オペレーティングシステム

ハードニングガイド

ドキュメントリリース日 : 2014 年 11 月 (英語版)
ソフトウェアリリース日 : 2014 年 11 月 (英語版)

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。

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

UNIX® は、The Open Group 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.

謝辞

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

HP Operations Orchestration のハードニング

このドキュメントでは、HP Operations Orchestration のセキュリティハードニングの構成方法について説明します。

管理作業については、『HP OO Administration Guide』を参照してください。

お断り

このガイドでは、HP OO デプロイメントをセキュリティのリスクや脅威から保護するための推奨事項を提供しています。アプリケーションをセキュリティ保護する理由として最も重要なのは、組織の重要情報の機密性、整合性、可用性の保護です。ただし、HP OO データを保護するには、HP OO のセキュリティ保護と、アプリケーションが実行されるコンピューティング環境 (たとえば、インフラストラクチャー) のセキュリティ保護の両方が必要です。

このガイドで扱うのは、HP OO をアプリケーションレベルでセキュリティ保護するための推奨事項までです。ユーザー環境内のインフラストラクチャーをセキュリティ保護する方法はカバーしていません。使用するインフラストラクチャー/環境について理解し、それぞれのハードニングポリシーを適用するのは、もっぱらユーザーの責任です。

セキュリティハードニングの推奨事項

1. 最新バージョンの HP OO をインストールします。詳細については、『HP OO インストールガイド』を参照してください。
2. (オプション) HP OO に FIPS 140-2 互換を構成します。これを行う場合は、Central サーバーを起動する前に構成する必要があります。「HP OO での FIPS 140-2 レベル 1 互換の構成」(21ページ)を参照してください。
3. Central サーバー証明書で TLS 暗号化を構成し、クライアント証明書で強い認証 (相互) を構成します。

   注: これは、インストール時に実行できます。

   RAS、デバッガー、および OOSH について、(サーバー証明書に) 必要であれば、証明書認証を提供し、Central に対する認証でクライアント証明書を使用します。「サーバーおよびクライアント証明書の認証」(6ページ)を参照してください。
4. HTTP ポートを削除し、キーストアと信頼ストアのパスワードを強いパスワードに置き換えて、HP OO Central サーバーをハードニングします。「HTTP/HTTPS ポートの変更と無効化」(14ページ)および「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。
5. キーストアと信頼ストアのパスワードを強いパスワードに置き換えて、HP OO Studio をハードニングします。「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。
6. SSL サポート対象サイファーから RC4 サイファーを削除します。「SSL サポート対象サイファーからの RC4 サイファーの削除」(13ページ)を参照してください。
7. (オプション) TLS プロトコルのバージョンを設定します。「TLS プロトコルの構成」(27ページ)を参照してください。
8. Central での認証を有効にします。『HP OO Central ユーザーガイド』の「認証の有効化」を参照し

てください。

内部ユーザーはセキュリティで保護されていないため、セキュアな LDAP と強いパスワードポリシーを使用してください。『HP OO Central ユーザーガイド』の「セキュリティのセットアップ - LDAP 認証」を参照してください。

9. オペレーティングシステムとデータベースのハードニング/セキュリティ保護を行います。
10. わかりやすいメッセージのセキュリティバナーを追加します。たとえば、「実稼働環境にログオンしようとしています。当システムの管理ルールを理解していないユーザーはログオンする前に必要なトレーニングを受けてください」というバナーを作成することができます。『HP OO Central ユーザーガイド』の「セキュリティバナーのセットアップ」を参照してください。
11. Windows および SQL サーバーの環境で、HP OO が Windows 認証と連携するように構成します。『HP OO データベースガイド』の「Windows 認証で稼働する HP OO の構成」を参照してください。
12. Central で監査が有効なことを確認します。詳細については、『HP OO Central ユーザーガイド』の「監査の有効化」を参照してください。

サーバーおよびクライアント証明書の認証

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暗号化された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。

注: Keytool はオープンソースのユーティリティです。

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 (秘密キー) 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 自己署名証明書を置き換えることをお勧めします。

サーバー証明書を使用した通信の暗号化

Central TLS サーバー証明書の置き換え

よく知られている証明機関によって署名された証明書か、ローカル証明機関の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> -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。

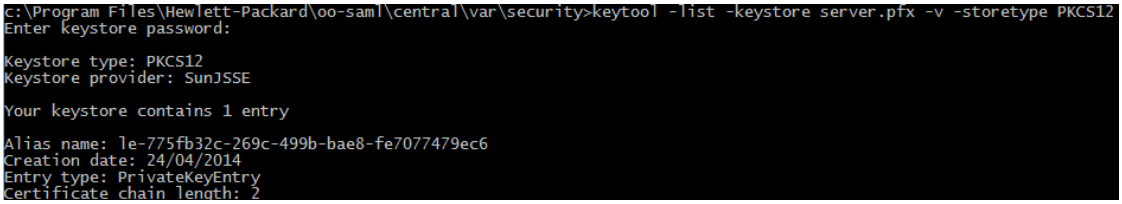
   必ず、強いパスワードを選択してください。

5. 次のコマンドを使用して証明書のエイリアスをリストします。

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

   証明書のエイリアスが表示されます。このエイリアスは、この次のコマンドで入力します。

次の例では、下から4番目の行です。



6. 次のコマンドを使用して、PKCS12形式のサーバー証明書をCentralの**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. インポートしたサーバー証明書のパスワードが元のサーバー証明書と異なる場合は、keyPassパスワードを変更することが重要です。「キーストア/信頼ストアのパスワードの変更」(11ページ)の手順を実行してください。

   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。

RAS信頼ストアへのCAルート証明書のインポート

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。よく知られているルートCA(Verisignなど)を使用する場合、証明書はすでに**client.truststore**ファイルに登録されているので、以下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、このデフォルトをカスタムCAまたはよく知られているCAに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、証明書内のFQDNが、要求のFQDNに一致することが検

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) が CA リスト内にまだない場合は、RAS の **client.truststore** ファイルにインポートします (デフォルトでは、よく知られているすべての CA がリストにあります)。

```
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。

OOSH 信頼ストアへの CA ルート証明書のインポート

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。よく知られているルート CA (Verisign など) を使用する場合、証明書はすでに **client.truststore** ファイルに登録されているので、以下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、このデフォルトをカスタム CA またはよく知られている CA に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、証明書内の FQDN が、要求の FQDN に一致することが検証されます。

   例:

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

5. 信頼されたルート証明機関 (CA) が CA リスト内にまだない場合は、Central の **client.truststore** ファイルにインポートします (デフォルトでは、よく知られているすべての CA がリストにあります)。

   ```
   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。
7. Central を起動します。

Studio デバッガー信頼ストアへの CA ルート証明書のインポート

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。よく知られているルート CA (Verisign など) を使用する場合、証明書はすでに **client.truststore** ファイルに登録されているので、以下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、このデフォルトをカスタム CA またはよく知られている CA に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、証明書内の FQDN が、要求の FQDN に一致することが検証されます。

   例:

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

5. 信頼されたルート証明機関 (CA) が CA リスト内にまだない場合は、Studio の **client.truststore** ファイルにインポートします (デフォルトでは、よく知られているすべての CA がリストにあります)。

   ```
   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 構成のキーストア、信頼ストア、およびサーバー証明書のパスワードの変更

1. Central が実行中であることを確認します。

   注: このステップを実行する前に、暗号化されたパスワードが存在することを確認します。パスワードを暗号化する方法については、『HP OO Administration Guide』の「Encrypting Passwords」を参照してください。

   OOSH から、次のコマンドを実行します。

   `set-sys-config --key <キー名> --value <暗号化されたパスワード>`

   ここで、<キー名> は、次の表のいずれかの値です。

構成アイテム	操作
`key.store.password`	**key.store** へのアクセスに使用するパスワードを設定します。デフォルト値は `"changeit"` です。 これは、下の手順で設定する `keystorePass` の値に対応している必要があります。
`key.store.private.key.alias.password`	**key.store** からサーバー証明書 (プライベートキー) にアクセスするために使用するパスワードを設定します。デフォルト値は `"changeit"` です。 これは、下の手順で設定する `keyPass` の値に対応している必要があります。

2. Central サービスを停止します。
3. Keytool を使用して、キーストア、信頼ストア、およびサーバー証明書のパスワードを変更します。
4. **<インストールディレクトリ>/central/tomcat/conf/** にある **server.xml** ファイルでもパスワードを編集します。
   a. 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"
sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2" truststoreFile="C:/Program
Files/Hewlett-Packard/HP Operations
```

```
Orchestration/central/var/security/client.truststore"
truststorePass="changeit" truststoreType="JKS"/>
```

b. パスワードを変更します。
   - keyPass - 指定する key.store ファイルのサーバー証明書の秘密キー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 - 指定する key.store 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。

     注: keyPass と同じパスワードを使用すること、および強いパスワードを使用することをお勧めします。

   - truststorePass - (信頼されているすべての CA を含む) 信頼ストアにアクセスするためのパスワード。デフォルト値は **javax.net.ssl.trustStorePassword** システムプロパティの値です。このプロパティが 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。

c. ファイルを保存します。

5. **<インストールディレクトリ>central\conf\central-wrapper.conf** にある **central-wrapper.conf** ファイルを編集します。

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

6. Central サービスを起動します。

RAS、OOSH、および Studio の信頼ストアのパスワードの変更

注: 以下の手順を実行する前に、Keytool を使用して、キーストア、信頼ストア、およびサーバー証明書のパスワードを変更してください。

- **スタンドアロンの 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** ファイルを編集して、信頼ストアの changeit パラメーターを、暗号化した形式の新しいパスワードで置き換えます。

  パスワードを暗号化する方法については、『HP OO Administration Guide』.の「Obfuscating Passwords」を参照してください。

Studio のキーストアおよび信頼ストアのパスワードの暗号化

HP 10.20 以降では、Studio のキーストアおよび信頼ストアのパスワードが暗号化されます。10.10 から 10.20 にアップグレードした後でこれらのパスワードが暗号化されるのは、パスワードが **<インストールディレ**

クトリ>/studio/Studio.l4j.ini ファイル内で変更されていないままの場合のみです。前のバージョンで変更された他のパスワードは、アップグレード中に自動的には暗号化されません。

信頼ストアやキーストアのパスワードを変更するには、暗号化形式またはプレーンテキストで、**Studio.l4j.ini** ファイルで変更します。変更したパスワードは、Studio プロセスのタスクマネージャーにプレーンテキストで表示されないように、手動で暗号化する必要があります。

1. Studio を閉じます。
2. **encrypt-password** スクリプトを **<インストールフォルダー>/central/bin** から探します。
3. 次のコマンドを発行して、カスタムパスワードを暗号化します。

```
encrypt-password.bat --obfuscate <パスワード>
```

4. 次のパラメーターについて、**<インストールディレクトリ>/studio/Studio.l4j.ini** ファイルでパスワードを変更します。

```
-Djavax.net.ssl.keyStorePassword={OBFUSCATED}<obfuscated_password>

-Djavax.net.ssl.trustStorePassword={OBFUSCATED}<obfuscated_password>
```

5. Studio のキーストアおよび信頼ストアのパスワードを、**<インストールディレクトリ>/studio/var/security/** フォルダーのキーツールで変更します。

   注: Studio リモートデバッガーに対してクライアント証明書が構成されていない場合、キーストアパス引数は無視されます。

6. Studio を起動します。

重要:encrypt-password スクリプトを使用した後で、コマンド履歴をクリアしてください。

これは、Linux OS の場合、パスワードパラメーターはクリアテキストで **/$USER/.bash_history** に保存され、`history` コマンドでアクセスできるためです。

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. 次の例に従ってコメントを削除し、パラメーターを変更して、RC4 暗号を無効にします。

```
jdk.certpath.disabledAlgorithms=RC4, MD2, RSA keySize < 1024
```

```
jdk.tls.disabledAlgorithms=RC4, MD5, DSA, RSA keySize < 1024
```

3. HP OO Central サーバーを再起動します。

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。

注: 前のバージョンの HP OO 10.x からアップグレードしたら、この手順を繰り返します。

HTTP/HTTPS ポートの変更と無効化

[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"
port="8080" protocol="org.apache.coyote.http11.Http11NioProtocol"
redirectPort="8443"/>
```

HTTPS コネクター:

```
<Connector SSLEnabled="true" URIEncoding="UTF-8" clientAuth="false"
compression="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"
protocol="org.apache.coyote.http11.Http11NioProtocol" scheme="https"
secure="true" sslProtocol="TLSv1.2" sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"
truststoreFile="C:/Program Files/Hewlett-Packard/HP Operations
Orchestration/central/var/security/client.truststore" truststorePass="changeit"
truststoreType="JKS"/>
```

デフォルトでは、両方とも有効です。

重要: Central ポートのいずれかを **server.xml** ファイルで変更または無効化する場合は、**central-wrapper.conf** ファイルおよび各 **RAS-wrapper.conf** ファイル更新し、Central URL を更新したポートで指すようにする必要もあります。そうしない場合、Central から実行するすべてのフローが失敗します。さらに、ロードバランサーの構成も必ずチェックしてください。

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** 値および HTTPS コネクターの **port** 値を変更する必要があります。
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

たとえば、セキュリティ上の理由から、HTTP ポートを無効にして、TLS 上にある暗号化されたチャネルを唯一の通信チャネルにしなければならないことがあります。

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. 信頼されたルート証明機関 (CA) が CA リスト内にまだない場合は、Central の **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>
   ```

   注: よく知られているルート CA (Verisign など) を使用する場合、証明書はすでに **client.truststore** ファイルに登録されているので、この手順を実行する必要はありません。
4. ファイルを保存します。
5. Central を再起動します。

注: インストール時に HTTP ポートを無効にすることもできます。

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-password" keystoreType="PKCS12" maxThreads="200" port="8443"
protocol="org.apache.coyote.http11.Http11NioProtocol" scheme="https" secure="true"
sslProtocol="TLSv1.2" sslEnabledProtocols="TLSv1,TLSv1.1,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。

クライアント証明書の認証 (相互認証)

X.509 証明書認証は、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。

TLS を相互認証で使用することもできます。サーバーは、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 TLS サーバー証明書を構必要があります。

接続を確立する前に、TLS 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。TLS 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 認成しておく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) が CA リスト内にまだない場合は、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 (CA リストには、よく知られているすべての CA がデフォルトで登録されています)。例:

   ```
   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. (オプション) `crlFile` 属性を追加し、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。「RAS 信頼ストアへの CA ルート証明書のインポート」(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/conf/ras-wrapper.conf** の **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。「Studio デバッガー信頼ストアへの CA ルート証明書のインポート」(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** ファイルを編集します。
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:

- HP OO 10.20 以降では、パスワードがデフォルトのままだった場合に、**Studio.l4j.ini** 内の `keyStorePassword` パラメーターがデフォルトで暗号化されます。このパラメーターは変更し、クリアテキストまたは暗号化して保存できます。
- 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。「OOSH 信頼ストアへの CA ルート証明書のインポート」(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。

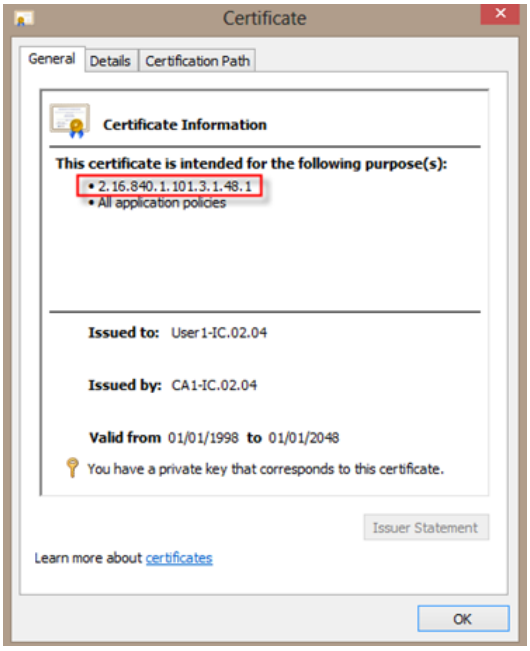
注: クライアント証明書で使用するユーザー (内部ユーザーまたは LDAP ユーザー) を定義します。ユーザー名は、証明書属性で定義する必要があります。デフォルトは、CN 属性の値です。詳細については、「証明書のプリンシパルの処理」を参照してください。

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)。



HP OO システムプロパティを変更する方法の詳細については、『HP OO Shell Guide』を参照してください。

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`

HP OO システムプロパティを変更する方法の詳細については、『HP OO Shell User Guide』を参照してください。

HP OO での FIPS 140-2 レベル 1 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 レベル 1 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:AES256
- ハッシュアルゴリズム:SHA256

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

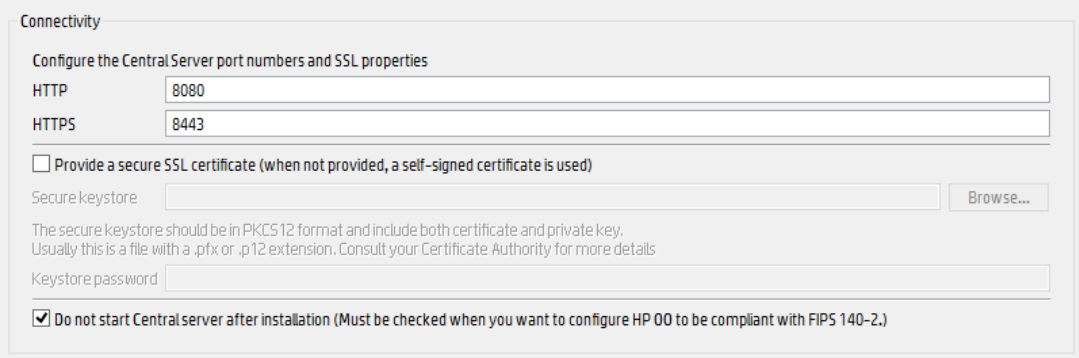
注: FIPS ですでに構成された HP OO 10.10 (以降) のインストールからアップグレードしている場合は、次の手順 4 と 5 を繰り返してから、「HP OO での FIPS 140-2 互換の構成」(22ページ)の「Java セキュリティファイルのプロパティ構成」の手順を繰り返す必要があります。

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。デフォルトディレクトリは **<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

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、「HP OO での FIPS 140-2 レベル 1 互換の構成」(21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」(25ページ)を参照してください。

注: AES では、NIST SP800-131A パブリケーションによる 128/192/256 の 3 つのキー長が認められています。

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: 使用していないデフォルトの CA ルート証明書は、HP OO 信頼ストアから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)。

新しい暗号化によるデータベースパスワードの再暗号化

データベースパスワードを、『HP OO Administration Guide』の「データベースパスワードの変更」の説明に従って、.再暗号化します。

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 暗号化キー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 暗号化キーの変更

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. 「HP OO での FIPS 140-2 レベル 1 互換の構成」(21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。

TLS プロトコルの構成

HP OO は、サポートされる TSL プロトコルバージョンを定義するように構成できます。HP OO は、デフォルトでは TLS v1、TLS v1.1、TLS v1.2 を使用できますが、これは制限することができます。

注: SSLv3 などの SSL バージョンはサポートされていません。

1. ファイル **<installation_folder>/central/tomcat/conf/server.xml** を開きます。
2. SSL コネクターを探します (ファイルの最後にあります)。
3. `sslEnabledProtocols` のデフォルト値を編集します。たとえば、次のように変更します。
   `sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"` to
   `sslEnabledProtocols="TLSv1.2"`
4. サーバーを再起動します。


Java™
COMPATIBLE