ネットワークおよび通信のセキュリティ

『HPE OOアーキテクチャーガイド』では、基本的なOOトポロジ、高可用性、ロードバランサーのセキュリティについて説明しています。

『HPE OO Network Architecture White Paper』では、必要なファイアウォール構成を説明し、ポリシー制限によって必要なファイアウォール構成を実装できない場合に適用可能な2つの推奨される回避方法を提示しています。

- SSHリバーストンネリング
- リバースプロキシ

通信チャネルのセキュリティ

サポートされるプロトコルおよび構成

HPE OOはTLSプロトコルをサポートしています。

詳細については、「Central TLSサーバー証明書の置き換え」(37ページ)を参照してください。

Centralのポートは、インストール中に管理者によって定義されます。

チャネルのセキュリティ

HPE OOは、次のセキュアなチャネルをサポートしています。

チャネル (ダイレクト)	サポートされるセキュアプロトコル
OOSH、ブラウザー、Studioリモートデバッガー、またはRAS → Central	セキュアなチャネルでは、暗号化にはTLS通信を、認証にはクライアント証明書を使用します。
Central → LDAPサーバー	CentralとLDAPの間の通信の暗号化には、TLSプロトコルを使用するセキュアLDAPを使用します。

RASのセキュリティ

リバースRAS (Centralが接続を開始するのを待機する) でのトポロジでは、RASのセキュリティは次のメカニズムによって保護されます。

- 接続試行が複数回、連続して失敗すると(共有シークレットを間違えて入力したことが原因)、遅延が発生します。

リバースRASの詳細については、『HPE OO Centralユーザーガイド』の「トポロジのセットアップ – ワーカーとRAS」を参照してください。

ユーザーの管理および認証

認証モデル

HPE OOで認証メカニズムのブートストラッピングを容易にするため、製品の認証は最初は無効になっています。

インストール後直ちに認証を有効にすることを強くお勧めします。

認証を有効にする方法については、『HPE 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。

Centralへのアクセスを認証するにはいくつかの方法があります。

ユーザーの識別方法を選択します。

- ユーザー名とパスワード
- クライアント証明書
- SAMLトークン
- シングルサインオン (HPE LWSSO)

次のいずれかのユーザー管理方法を選択します。

- LDAPユーザー: Active DirectoryとしてLDAPサーバーに保存 (推奨)
- 内部ユーザーおよびパスワード: Centralサーバーにローカルに保存 (非推奨)

ユーザーのタイプ

ユーザーのタイプごとに異なるアクセス許可を割り当てることができます。たとえば、フロー作成者、管理者、システム管理者など。

異なるアクセス許可を必要とするこの他のタイプのユーザーの例については、『OOコンセプトガイド』の「主要なペルソナ」を参照してください。

認証の管理と構成

内部ユーザーまたはLDAPユーザー

Central UIで内部ユーザーとパスワードを設定するか、LDAPサーバーでユーザーを定義してLDAPグループをCentralの役割にマッピングすることができます。

デフォルト役割に関する属性を使用して、役割の1つを設定することができます。その場合は、最小限の権限を持つ役割にしてください。この役割にアクセス許可を付与する場合は、この役割に明示的に関連付けられているユーザーだけでなく、すべてのLDAPユーザーに影響することに留意してください。

詳細については、『HPE OO Centralユーザーガイド』の「セキュリティのセットアップ - 役割」の「デフォルトの役割としての役割の割り当て」を参照してください。

以下も参照:

- 『HPE OO Centralユーザーガイド』の「システムアカウントへのアクセス許可の割り当て」
- 『HPE OO Centralユーザーガイド』の「コンテンツアクセス許可の設定」

Studioのワークスペースへのアクセス

Studioで複数のワークスペースを作成する場合は、ユーザーが読み取りと書き込みのアクセス許可を持っているフォルダーの下にのみワークスペースを作成することをお勧めします。

ワークスペースをパブリックフォルダーの下に作成すると、すべてのユーザーがアクセスできるため、改ざんや機密情報の漏洩が発生しやすくなります。

バックアップ

データの損失を防ぐため、セキュアなメディアにサーバーのデータを定期的にバックアップすることを強くお勧めします。これは、ディザスターリカバリやビジネスの継続にも役立ちます。

OOをインストールしたら、**central\var\security**フォルダーと**central\conf\database.properties**ファイルを必ずバックアップしてください。

データベーススキーマでは、一部のデータが暗号化され、復号化キーはOO Centralサーバーにローカルに保存されています。システムファイルが破損または削除されるとデータの復号化が不可能になるので、スキーマは使用できなくなります。

注: キーは暗号化されているので、キーをバックアップに含めることが重要です。上記のスクリプトは**bin**フォルダーにあります。

参照:

- 『OO Administration Guide』の「Backing Up OO」
- 『OO Administration Guide』の「Setting up Disaster Recovery」
- 『OOインストールガイド』の「Centralセキュリティファイルのバックアップと復元」
- 『OOアーキテクチャーガイド』の「OOデプロイメントでのロードバランサーの使用」

暗号化

暗号化モデル

HPE OOは、機密データを保護するために、暗号化アルゴリズムとハッシュアルゴリズムをサポートしています。暗号化は、HPE OOシステムのパスワードや定義などの機密データの漏洩および変更を防ぐように設計されています。

認証されていないユーザーによる復号化を防ぐためには、既知の脆弱性がないよく知られている標準的なアルゴリズムを使用することが重要です。

注: たとえば、SSLプロトコルには既知の脆弱性があるため、SSLは使用されません。

静的データ

保存されているすべてのパスワードがよく知られているアルゴリズムを使用して保護されており、クリアテキストのままのパスワードはありません。

例:

- システムアカウントのパスワードは暗号化されています。
- 内部ユーザーのパスワードはハッシュされています。
- データベースパスワードは暗号化されています。

転送中のデータ

OOは、トランスポートレイヤーセキュリティ(TLS)プロトコルを使用して、コンポーネント(CentralやRASなど)間のデータを暗号化します。

HTTPポートの無効化

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することをお勧めします。詳細については、「HTTP/HTTPSポートの変更またはHTTPポートの無効化」(47ページ)を参照してください。

暗号化の管理

推奨される暗号化のベストプラクティス

セキュリティレベルおよび暗 号 化 レベルを高 めるためには、OOをFederal Information Processing Standards (FIPS) 140-2互 換 に構 成 することをお勧 めします。OOをFIPS 140-2レベル1互 換 に設 定 できます。

デフォルト の構 成 セット

- 対 称 キーアルゴリズム: AES (キー長 : 128)
- ハッシュアルゴリズム: SHA1

詳細設定

OOでFIPS 140-2互換の構成を行うと、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

「HPE OOでのFIPS 140-2互換の構成」(58ページ)を参照してください。

デジタル証明書

デジタル証明書は、ユーザー、サーバー、ステーションなどの電子「パスポート」です。

- ブラウザーとCentralサーバーの間で暗号化を使用するには、サーバー側にデジタル証明書をインストールする必要があります。
- Centralサーバーの認証にクライアント証明書を使用するには、クライアント側 (たとえば、ブラウザー上のRAS、OOSH、Studioなど) にクライアント証明書をインストールする必要があります。

OOでは、Java 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、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。

証明書の場所

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**: OOプライベート証明書 (秘密キーを含む) が含まれています。

KeyStoreおよびTrustStoreへのアクセス制御

TrustStoreおよびKeyStoreの保存では、Centralサービスを実行するユーザーに対してのみ読み取りアクセス許可を付与することをお勧めします。

OO自己署名証明書の置き換え

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、OO自己署名証明書を置き換えることをお勧めします。

証明書の置き換えプロセスの一環で、PKCS12形式の証明書がCAを使用して作成されます。証明書プロセスの詳細についてはCAにお問い合わせください。または、コーポレートポリシーを参照してください。

詳細については、「Central TLSサーバー証明書の置き換え」(37ページ)を参照してください。

デジタル署名のコンテンツパックへの追加

コンテンツパックに信頼されたCAのデジタル署名が付いている場合は、コンテンツは信頼できます。

デジタル署名の追加は必須ではありません。

- OO設定済みのコンテンツパックには、Verisignのデジタル署名が含まれています。
- OOの作成者には、カスタムコンテンツパックにデジタル署名を追加することをお勧めします。
- 署名済みのコンテンツパックが破壊されている場合は、デプロイできません。
- 署名の有効期限が切れた場合は、デプロイ前に警告が表示されるので、期限切れの署名を無視することを確認するチェックボックスを選択する必要があります。

署名されていないコンテンツパックに注意してください。未署名のコンテンツパックは信頼できず、悪意のあるコンテンツが含まれている可能性があります。未署名のコンテンツパックは破壊され、署名が削除されている可能性があることにも注意してください。

コンテンツパックのデジタル証明書の詳細については、『OO Centralユーザーガイド』の「コンテンツパックのデプロイと管理」を参照してください。

コンテンツパックの機 密 情 報

システムアカウントのパスワード

コンテンツパックの作 成 時 にパスワードを含 めないでください。パスワードはコンテンツパック内 部 で暗 号 化 されるので、セキュアなオプションではありません。

OOのセキュリティに関 するベストプラクティスは、Centralでシステムアカウントのパスワードを設 定 することです。詳 細 については、『OO Centralユーザーガイド』の「コンテンツパックのシステムアカウントのセットアップ」を参 照してください。

監査とログファイル

監査

監査を行うと、Centralサーバーで行われるアクション(ログイン、フローの起動、スケジュールの作成、構成の編集など)を追跡できます。監査データによって、Centralシステム上のユーザー操作を追跡して、誰が何の操作をいつ行ったか追跡することができます。たとえば、監査によって、ユーザーによるフローの実行、構成の更新、スケジュールの削除、または認証の失敗を確認できます。

監査データはデータベースに保存されます。詳細については、『HPE OO API Guide』の「Auditing」を参照してください。

ログ

ログによって、エラー、警告、情報、デバッグメッセージをトレースできます。

ログはファイルサーバーの次の場所に保存されます。

- Central - **<OOインストール>/central/var/logs**
- Studio - **<ユーザー>/.oo/logs**
- RAS - **<OOインストール>/ras/var/logs**

監査レコードやログファイルに保存される機密データなし

HPE OOシステムでは、機密データは監査レコードやログファイルに保存されません。

監査レコードの取得

監査レコードは、API経由で、またはOO_AUDITテーブルへのクエリによって取得できます。詳細については、『HPE OO API Guide』の「Auditing」を参照してください。

監査データの例:

```
[
{
“time”:1412312016740, “type”:”AuditConfigurationChange”,
“group”:”AuditManagement”, “subject”:” mydomain\myuser2”, “outcome”:”Success”,
“data”:”{"enabled":false}”
},
{
```

```
“time”:1412312016722, “type”:”InternalUserDelete”, “group”:”Authentication-
Authorization”, “subject”:”mydomain\myuser2”, “outcome”:”Success”, “data”:
{"usersNames":["admin"]}"

}

]
```

APIとインタフェース

APIモデルとインタフェースモデル

HPE OO CentralのUIではなく、HPE Operations OrchestrationのパブリックApplication Programming Interfaces (API) を使用して作業して、同じ操作を実行することができます。削除や監査などの一部の操作は、API経由でのみ実行できます。パブリックAPIはHTTPベースです。すべてのAPIがRESTfulで、JavaScript Object Notationを使用します。

APIとインタフェースのセキュリティ構成の機能と管理

APIを使用してセキュアに作業することが重要です。APIを使用して作業している間は、このガイドに記載されているセキュリティメカニズム (認証、暗号化など) を使用してください。

APIインタフェースは、HTTPまたはHTTPS上で動作します。

注: APIを使用してHTMLを表示する場合は、XSS攻撃から保護する必要があります。

詳細については、『HPE OO API Guide』の以下の章を参照してください。

- 「LDAP Configuration」
- 「Users」
- 「LW SSO Configuration」
- 「Authentication」
- 「Roles」

セキュリティに関するQ&A

外部CAによる署名が可能な証明書要求の生成方法は?

証明書要求をエクスポートして外部CAに送って署名してもらいます。手順については、「Central TLSサーバー証明書の置き換え」(37ページ)を参照してください。

HPE OOを使用するのはどのTCP/UDPポートですか?方向、ユーザー、暗号化とは?

HPE OOをインストールする場合は、[HTTP/HTTPS] フィールドで、Centralサーバーに使用可能な少なくとも1つのポートを構成する必要があります。デフォルト値は8080および8443ですが、変更可能です。Centralと他のコンポーネントの間のセキュアなチャネルの詳細については、「ネットワークおよび通信のセキュリティ」(15ページ)を参照してください。

資格情報はどこにどのように保存されますか (管理者アカウント、統合ユーザー)?

「ユーザーの管理および認証」(17ページ)を参照してください。

Central/RAS/Studioの自己署名SSL証明書の構成方法は?

HPE OOのインストール中に証明書を提示しないと、自己証明書がデフォルトで作成されます。ただし、セキュリティ上の理由から、自己署名証明書は使用しないようにしてください。HPEは、カスタムルートCAまたは知名度の高いCAの証明書の使用をお勧めします。

HPE OOの証明書の構成の詳細については、「サーバー証明書を使用した通信の暗号化」(36ページ)を参照してください。

監査を有効または無効にする方法は?

デフォルトでは、監査は有効になっていません。監査を有効にする方法の詳細については、『HPE 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。監査の詳細については、「監査とログファイル」(27ページ)を参照してください。

どの程度詳細なログか? またログ量の変更方法は?

ログはさまざまなレベルの詳細度に設定できます。デフォルトレベルはINFOですが、調整可能です。詳細については、『HPE OO Administration Guide』の「Adjusting the Logging Levels」を参照してください。

ログファイルの詳細については、「監査とログファイル」(27ページ)を参照してください。

機密情報の暗号化方法は?

「暗号化」(22ページ)を参照してください。

CentralとRASの間の通信は暗号化されていますか?

HTTPSを使用している場合は、暗号化されています。

HPE OOと他の統合コンポーネント(HPNA、CSA、ADなど)の間の通信は暗号化されていますか?

これは、使用している統合に依存します。HTTPSを使用している場合は、暗号化されています。

フローライブラリへのアクセスをユーザーの役割に基づいて制限する方法は?

『HPE OO Centralユーザーガイド』の「セキュリティのセットアップ – 役割」を参照してください。

OOがサポートしている認証メカニズムは?

サポートされている認証メカニズムは、LDAP、SAML、内部ユーザーです。HPE OOは、クライアント証明書とLW SSOもサポートしています。「ユーザーの管理および認証」(17ページ)を参照してください。

HPE OOはFIPS 140-2互換ですか?

はい。詳細については、「HPE OOでのFIPS 140-2互換の構成」(58ページ)を参照してください。

CentralとRASの間の認証方法は?

ユーザーパスワードまたはクライアント証明書。

すべてのパスワードが暗号化されて保存またはハッシュされますか?

はい。保存されているすべてのパスワードがよく知られているアルゴリズムを使用して保護されており、クリアテキストのままのパスワードはありません。

CentralユーザーのIPアドレスを制限できますか?

いいえ。現時点ではサポートされていません。

HPE OOはコモンクライテリアの認証を受けていますか?

これは進行中です。現在、「評価段階」にあります。詳細については、https://www.cse-cst.gc.ca/en/canadian-common-criteria-scheme/publication/list/evaluation-productを参照してください。

OOSHを使用した場合、機密データをCentralに渡すことはできますか?

Centralへの接続時にはセキュアなチャネルの使用をお勧めします。「ネットワークおよび通信のセキュリティ」(15ページ)を参照してください。

Operations Orchestrationのハードニング

このセクションでは、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。

注: 管理作業については、『OOインストール、アップグレード、構成ガイド』を参照してください。

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HPE OOをインストールします。詳細については、『OOインストール、アップグレード、構成ガイド』を参照してください。
2. (オプション) 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。「HPE OOでのFIPS 140-2レベル1互換の構成」(56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使用」(35ページ)を参照してください。
4. HTTPポートを削除し、KeyStoreとTrustStoreのパスワードを強いパスワードに置き換えて、HPE OO

Centralサーバーをハードニングします。「HTTP/HTTPSポートの変更またはHTTPポートの無効化」(47ページ)および「KeyStore/TrustStoreのパスワードの変更と暗号化/難読化」(42ページ)を参照してください。

5. KeyStoreとTrustStoreのパスワードを強いパスワードに置き換えて、HPE OO Studioをハードニングし、構成ファイルのパスワードを暗号化または難読化します。「KeyStore/TrustStoreのパスワードの変更と暗号化/難読化」(42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」(46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」(63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。『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。『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。『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、OOがWindows認証と連携するように構成します。『OOデータベースガイド』の「Windows認証で稼働する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、『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。

デフォルトのセキュリティ設定

多くの場合、構成済みで提供されるデフォルトのセキュリティ設定は修正することをお勧めします。

- **認証** – Centralで、認証はデフォルトでは有効になっていません。ユーザーのセットアップが完了したら、すぐに有効にすることをお勧めします。詳細については、『HPE 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。
- **監査** – Centralで、監査はデフォルトでは有効になっていません。有効にすることをお勧めします。詳細については、『HPE 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。

- **TLS暗号化** – HPE OOは、デフォルトで3つのTLSプロトコル(1.0、1.1、1.2)をサポートしています。最新バージョンの使用をお勧めします。詳細については、「TLSプロトコルの構成」(63ページ)を参照してください。
- **TLSサーバー証明書** – デフォルトでは、OOサーバーのインストール時に、CA証明書の提示がユーザーに求められます。
- **クライアント証明書** – クライアント証明書は、デフォルトでは有効になっていません。Centralへの認証には、クライアント証明書を使用することをお勧めします。詳細については、「クライアント証明書認証の構成 (Central)」(49ページ)を参照してください。
- **KeyStore、TrustStore、およびサーバー証明書のパスワード** – デフォルトでは、KeyStore、TrustStore、およびサーバー証明書用にJavaパスワードが提供されています。これらのパスワードは、暗号化されたパスワードに置き換えることをお勧めします。詳細については、「KeyStore/TrustStoreのパスワードの変更と暗号化/難読化」(42ページ)を参照してください。
- **RC4暗号** – RC4暗号はデフォルトで有効になっています。RC4暗号はJREレベルで無効にすることをお勧めします。詳細については、「SSLサポート対象サイファーからのRC4サイファーの削除」(46ページ)を参照してください。
- **セキュリティバナー** – Centralで、セキュリティバナーはデフォルトでは有効になっていません。これは、カスタムメッセージを指定して、有効にすることをお勧めします。詳細については、『HPE 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。
- **データベースのWindows認証** – Centralで、Windows認証はデフォルトでは有効になっていません。WindowsおよびSQLサーバーの環境を使用する場合は、Windows認証と連携するようにHPE OOを構成することをお勧めします。『HPE OOデータベースガイド』の「Windows認証で稼働するHPE OOの構成」を参照してください。
- **デフォルトのアルゴリズム** – **encryption.properties**ファイルにはデフォルトのアルゴリズムが含まれています。FIPSへの準拠が必要な場合は、「HPE OOでのFIPS 140-2レベル1互換の構成」(56ページ)を参照してください。FIPS 140-2 Level 1のデフォルトの詳細については、「暗号化」(22ページ)の「暗号化管理」を参照してください。
- **Javaポリシー** – **java.policy**ファイルは、デフォルトではハードニングされていません。**java.policy**ファイルの変更方法については、「フローがCentral/RASのローカルファイルシステムにアクセスできなくする」(63ページ)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使用

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。

HPE 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、HPE 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。

HPE 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**: HPE OO証明書 (秘密キー) が含まれています。

推奨事項:

- HPE 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、HPE OO自己署名証明書を置き換えることをお勧めします。
- TrustStoreとKeyStoreは、Centralサービスを実行するユーザーのみに対する読み取り権限で格納することをお勧めします。
- Keytoolの使用後はコンソールをクリアするか、パスワード入力のプロンプトを使用す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:

   PKCS12形式の証明書を生成するにはCAを使用する必要があります。その手順はCAベンダーとポリシーによって異なる可能性があるため、CAに連絡し、証明書の生成プロセスに関する詳細な説明を求める必要があります。

   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KeyStore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。

```
C:\Program Files\Hewlett-Packard\oo-saml\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. 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。「KeyStore/TrustStoreのパスワードの変更と暗号化/難読化」(42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KeyStore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。「KeyStore/TrustStoreのパスワードの変更と暗号化/難読化」(42ページ)を参照してください。

8. Centralを起動します。

CentralのTrustStore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**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、HPE 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. 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.truststoteへのパス>
   -file <証明書名.cer> -storepass <changeit>
   ```

3. Centralを起動します。

RAS TrustStore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、HPE 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`の値が**false**に設定されていることを確認します。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`の値が**true**に設定されていることを確認します。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
```

注: このプロパティは、デフォルトで**true**に設定されています。

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へのパス>
   -file <証明書名.cer> -storepass <changeit>
   ```

6. RASを起動します。

OOSH TrustStore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、HPE 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`の値が**false**に設定されていることを確認します。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`が**true**に設定されていることを確認します。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
   ```

注: このプロパティは、デフォルトで**true**に設定されています。

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へのパス>
-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 TrustStoreへのCAルート証明書のインポート

Central、SVN、またはGITサーバーでカスタム証明書を使用する場合、これらと組み合わせてStudioを使用するには、Studioの**client.truststore**ファイルに、信頼されるルート証明機関 (CA) を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、次の手順を実行する必要はありません。

デフォルトで、HPE 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。

新規の**.oo**フォルダーの場合、Studioは**<インストールディレクトリ>/studio/var/security**の**client.truststore**ファイルを**<ユーザー>/.oo**フォルダーにコピーします。これは、Studioで (たとえば、Studioリモートデバッガーの) 証明書を自動的にインポートできるようにするために、一度だけ行われる操作です。このファイルが存在する場合は、それが**client.truststore**として使用され、存在しない場合はStudioインストールのファイル (**<インストールディレクトリ>/studio/var/security/client.truststore**) が使用されます。

10.5x以降にアップグレードした場合、Truststoreの場所は**<ユーザー>/.oo**フォルダーです。

証明書を手動でインポートする場合は、**.oo/client.truststore**またはStudioインストールフォルダーの**client.truststore**のいずれかにコピーできます。

複数のワークスペースを使用する場合、**.oo**フォルダーにある**client.trustore**ファイルへの変更は、特定のワークスペースに対してのみ適用されます。新規作成したワークスペースすべてに変更を適用するには、Studioのインストールフォルダーにある**client.trustore**ファイルを編集します。

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. Studioを閉じて、**<ユーザー>/.oo**にある元の**client.truststore**ファイルをバックアップします。

   たとえば、**C:/Users/<ユーザー名>/.oo**

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`の値が**false**に設定されていることを確認します。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`が**true**に設定されていることを確認します。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)。**<黄色>**でマークされているパラメーターを置き換えます。

   ```
   keytool -importcert -alias <任意のエイリアス> -keystore <client.truststoreへのパス>
   -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。

KeyStore/TrustStoreのパスワードの変更と暗号化/難読化

Central構成のKeyStore、TrustStore、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、『HPE 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、KeyStore、TrustStore、およびサーバー証明書のパスワードを変更します。

   KeyStoreのパスワードを変更するには、次のkeytoolコマンドを使用します。

   ```
   keytool -storepasswd -keystore <インストールフォルダー
   >/central/var/security/key.store
   ```

   サーバー証明書の秘密キーエントリパスワードを変更するには、次のkeytoolコマンドを使用します。

   ```
   keytool –keypasswd -alias tomcat -keystore <インストールフォルダー
   >/central/var/security/key.store
   ```

   TrustStoreのパスワードを変更するには、次のkeytoolコマンドを使用します。

   ```
   keytool -storepasswd -keystore <インストールフォルダー
   >/central/var/security/client.truststore
   ```

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"/>
```

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) TrustStore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、TrustStoreのパスワードは設定されません。TrustStore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TrustStoreにアクセスします。TrustStoreの内容の検証は省略されます。

b. 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**にある**central-wrapper.conf**ファイルを編集して、TrustStoreのパスワードを、暗号化または難読化した形式の新しいパスワードに置き換えます。例:

   ```
   wrapper.java.additional.<x>=-Djavax.net.ssl.trustStorePassword={ENCRYPTED}
   <encrypted_password>
   ```

   ```
   wrapper.java.additional.<x>=-Djavax.net.ssl.trustStorePassword={OBFUSCATED}
   <obfuscated_password>
   ```

   パスワードを暗号化する方法については、「パスワードの暗号化と難読化」(45ページ)を参照してください。

6. Centralサービスを起動します。

RAS、OOSH、およびStudioのTrustStoreのパスワードの変更

注: 次の手順を実行する前に、Keytoolを使用して、KeyStore、TrustStore、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 TrustStore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、TrustStoreのpasswordパラメーターを変更します。

- **OOSH TrustStore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、TrustStoreのpasswordパラメーターを変更します。
- **Studio TrustStoreのパスワードを変更するには、次の手順を実行します。**暗号化した形式のパスワードを指定したプロパティ**client.truststore.password**を**<ユーザー>/.oo**フォルダーの**Studio.properties**ファイルに追加します。

  ```
  client.truststore.password={OBFUSCATED}6L9+NqBjKYp5heuvMEzg0g==
  ```

  このプロパティが定義されていない場合、Studioはシステムプロパティ**javax.net.ssl.trustStorePassword**にフォールバックして、TrustStoreのパスワードを取得します。

パスワードを暗号化する方法については、「パスワードの暗号化と難読化」(45ページ)を参照してください。

パスワードの暗号化と難読化

パスワードはencrypt-passwordスクリプトを使用して暗号化または難読化できます。このスクリプトは**<インストールフォルダー>/central/bin**に保存されています。

暗号化を使用することを推奨します。

> **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。

パスワードの暗号化

1. 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。
2. -e -p <パスワード>オプションを指定して、スクリプトを実行します。ここでパスワードには暗号化するパスワードを指定します。

   > **注:** パスワードを暗号化するためのフラグとしての-p、または--passwordのいずれかを使用できます。

   暗号化したパスワードは次のように表示されます。

   ```
   {ENCRYPTED}<文字列>
   ```

パスワードの難解化

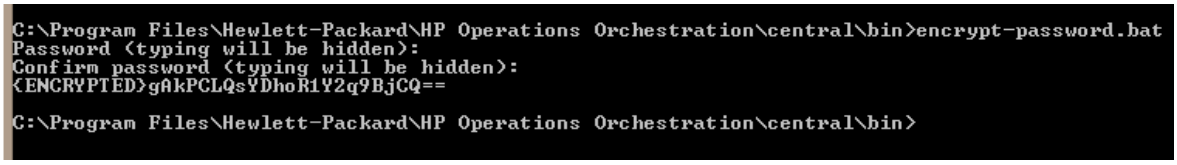
1. encrypt-passwordスクリプトを**<インストールフォルダー>/central/bin**から探します。
2. -o <パスワード>オプションを指定してスクリプトを実行します。ここでパスワードには難解化するパスワードを指定します。

   難解化したパスワードは次のように表示されます。

   ```
   {OBFUSCATED}<文字列>
   ```

パスワード入力のためのプロンプトの作成

-p引数を指定しないでencrypt-passwordスクリプトを実行することをお勧めします。例:



これにより、非表示パスワード入力のためのプロンプトが作成されます。

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るが、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する。

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. 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HPE 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 またはHTTPポート の無 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。

HTTPポートの無効化

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。

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へのパス>
   -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:\HPE\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:/HPE/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使用」(35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。

   例:

   ```
   <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" server="OO" sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"
   sslProtocol="TLSv1.2" truststoreFile="C:/Program Files/Hewlett-Packard/HP
   Operations Orchestration/central/var/security/client.truststore"
   truststorePass="changeit" truststoreType="JKS"/>
   ```

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

   以下のプロパティをコメント解除し、クライアント証明書の場所とパスワードを管理者ユーザーとともにクライアント証明書に設定します。

   ```
   #wrapper.java.additional.23=-Djavax.net.ssl.keyStore="%CENTRAL_
   HOME%/var/security/certificate.p12"

   #wrapper.java.additional.24.stripquotes=TRUE

   #wrapper.java.additional.25=-Djavax.net.ssl.keyStorePassword={OBFUSCATED}
   ZUoMreNLw6qIOyzX7g5YKw==

   #wrapper.java.additional.26=-Djavax.net.ssl.keyStoreType=PKCS12
   ```

   パスワードを暗号化する方法については、「パスワードの暗号化と難読化」(45ページ)を参照してください。

7. 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。

HPE 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。

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。

前提条件: CentralのCAルート証明書をRAS TrustStoreにインポートする必要があります。「RAS TrustStoreへのCAルート証明書のインポート」(39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={OBFUSCATED}<obfuscated_password>
   ```

   ```
   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。『OO Central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。

HPE 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、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。「パスワードの暗号化と難読化」(45ページ)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

前提条件: CentralのCAルート証明書をStudio Debugger TrustStoreにインポートする必要があります。「Studio TrustStoreへのCAルート証明書のインポート」(41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={OBFUSCATED}<obfuscated_password>
   ```

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

4. Studioを起動します。

注:

- HPE 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、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。「パスワードの暗号化と難読化」(45ページ)を参照してください。
- 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。
- HPE 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、HPE 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

前提条件: CentralのCAルート証明書をOOSH TrustStoreにインポートする必要があります。「OOSH TrustStoreへのCAルート証明書のインポート」(40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={OBFUSCATED}<obfuscated_password>
   -Djavax.net.ssl.keyStoreType=PKCS12
   ```

4. OOSHを起動します。

注:

HPE 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、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。「パスワードの暗号化と難読化」(45ページ)を参照してください。

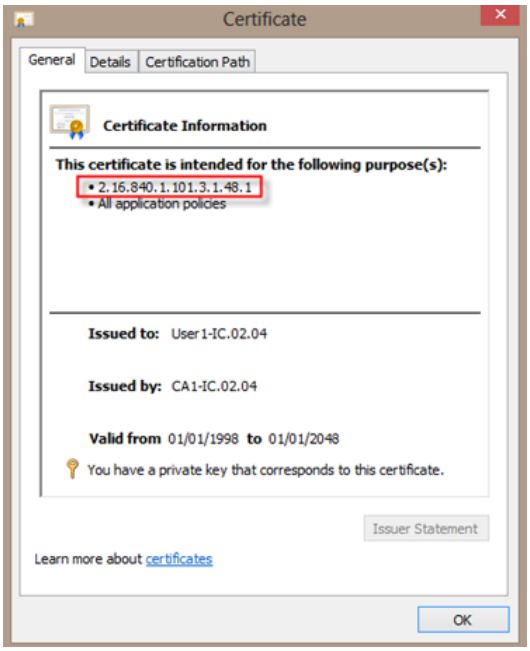
クライアント証明書で使用するユーザー(内部ユーザーまたはLDAPユーザー)を定義します。ユーザー名は、証明書属性で定義する必要があります。デフォルトは、CN属性の値です。詳細については、「証明書のプリンシパルの処理」を参照してください。

HPE 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、HPE 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

HPE 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。
- HPE 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)。



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、『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、HPE 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.

OOから証明書のSubject Alternative Nameフィールドの読み取りを可能にする

証明書のSubject Alternative Nameフィールドを OOから読み取れるようにするには、構成アイテム x509.principal.lookup.fieldを使用します。

この構成アイテムは、ユーザー名の抽出に使用する証明書のフィールドを指定します。

有効な値:

- subjectDN - 証明書のSubjectフィールドを表します。すなわち、OOはデフォルトの動作を実行し、**Subject**フィールドからユーザー名を抽出しようとします。これはデフォルト値です。
- subjectAltNames.otherName.principalName - Subject Alternative Names証明書拡張の Other Nameエントリに含まれているUser Principal Name (OID 1.3.6.1.4.1.311.20.2.3) を表します。CAC認証の場合は、User Principal Nameの値を使用することが要求される場合があるので、この値を使用します。

HPE OO構成アイテムを変更する方法の詳細については、『HPE OO Shell (OOSH) User Guide』を参照してください。

HPE OOでのFIPS 140-2レベル1互換の構成

以下のセクションでは、HPE 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.

HPE OOでFIPS 140-2互換の構成を行うと、HPE 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

HPE 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6.2.1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: HPE 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、HPE OOの再インストールが必要です。

前提条件

アップグレードプログラムのメモ:

FIPSですでに構成されたHPE OO 10.10 (以降) のインストールからアップグレードする場合は、「アップグレードプログラムの前提手順」を参照してください。

HPE 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、LW SSOを無効にする必要があります。

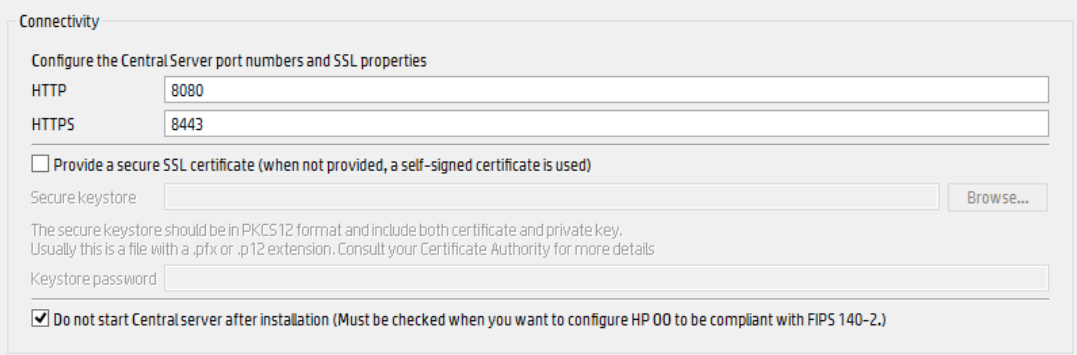
1. FIPS 140-2互換構成には、HPE OOバージョン10.10以降の新規インストールが必要です。

   インストール済みのHPE 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. HPE 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**
   - **<インストールディレクトリ>\java** (**java**フォルダー全体のバックアップが必要)
4. http://www.oracle.com/technetwork/java/javase/downloads/server-jre8-downloads-2133154.htmlから**Server Oracle JRE 8**をダウンロードし、**OpenJDK (Zulu) JRE**を**Server Oracle JRE**に置き換えます。
   a. **<インストールディレクトリ>\JAVA**フォルダーの内容をすべて削除します。
   b. ダウンロードしたアーカイブを展開します。
   c. **JRE**フォルダーの内容を**<インストールディレクトリ>\JAVA**にコピーします。
5. 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/server-jre8-downloads-2133154.html

   注: ファイルのデプロイとHPE 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。
6. RSA BSAFE Cryptoソフトウェアファイルをインストールします。HPE 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、HPE 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-6.2.1.jar**
   - **<インストールディレクトリ>\central\lib\cryptojcommon-6.2.1.jar**
   - **<インストールディレクトリ>\central\lib\jcmFIPS-6.2.1.jar**

アップグレードプログラムの前提手順

1. Server Oracle JRE 8をダウンロードし、OpenJDK (Zulu) JREをServer Oracle JREに置き換えます。
   a. **<アップグレードディレクトリ>\JAVA**フォルダーの内容をすべて削除します。
   b. ダウンロードしたアーカイブを展開します。
   c. **JRE**フォルダーの内容を**<アップグレードディレクトリ>\JAVA**にコピーします。

   http://www.oracle.com/technetwork/java/javase/downloads/server-jre8-downloads-2133154.html
2. 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/jce8-download-2133166.html

   ファイルのデプロイとHPE 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。
3. RSA BSAFE Cryptoソフトウェアファイルをインストールします。HPE 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**にコピーします。

   (ここで、**<oo_jre>**は、HPE OOアップグレードプログラムによって使用されるJREがインストールされているディレクトリです。これは、デフォルトでは**<アップグレードディレクトリ>\java**です)。
   - <インストールディレクトリ>\central\lib\cryptojce-6.2.1.jar
   - <インストールディレクトリ>\central\lib\cryptojcommon-6.2.1.jar
   - <インストールディレクトリ>\central\lib\jcmFIPS-6.2.1.jar

次に、「HPE OOでのFIPS 140-2互換の構成」(58ページ)の「Javaセキュリティファイルのプロパティの構成」セクションの手順を実行します。

HPE 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HPE OOで必要な構成手順を示します。

1. Javaセキュリティファイルのプロパティ構成。
2. encryption.propertiesファイルの構成とFIPSモードの有効化。
3. FIPS互換のHPE OO暗号化の作成。
4. 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。
5. HPE 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: HPE OO 10.x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.xにアップグレードする場合は、次の手順を実行する必要があります。

注: FIPSで構成済みのHPE OO 10.10以降のインストールからアップグレードする場合は、「HPE OOでのFIPS 140-2レベル1互換の構成」(56ページ)の「アップグレードプログラムの前提手順」セクションを実行してから、ここの手順を実行する必要があります。ここで、**<oo_jre>** は (場所 **<アップグレードディレクトリ>\JAVA**にある) アップグレードに含まれるJREです。

抽出された「**upgrade**」フォルダー内の「**java**」フォルダーで、すべての変更を行ってください。

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

HPE 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OO暗号化の作成

FIPS互換の設定には、HPE 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」(61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TrustStoreのパスワードを変更することをお勧めします。「KeyStore/TrustStoreのパスワードの変更と暗号化/難読化」(42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、OO TrustStore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)。

注: クライアント証明書を使用する場合、その証明書は、FIPS準拠のRSA JCEプロバイダーと、上記リストに示すFIPSでサポートされるセキュアなハッシュアルゴリズムで生成されている必要があります。

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、『HPE 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。

HPE OOの起動

FIPS暗号化の置き換え

HPE 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。

HPE OO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

generate-keys.bat/shファイルを使用して、暗号化リポジトリのFIPS暗号化キーを置き換えます。

注: このプロセスでは**encryption_repository**ファイルがバックアップされます。そのため、適切な書き込み権限が必要です。

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、**<Centralインストールフォルダー>/var/security**フォルダーからそのファイルを削除します。
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。

5. **Y**キーを押して、続行します。

   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。

注: ユーザーが**Y**または**N**を入力するための一時停止を行わずに、**generate-keys**スクリプトを実行する場合は、スクリプトを実行するときにサイレントモードフラグ **–s**を使用します。

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. 「HPE OOでのFIPS 140-2レベル1互換の構成」(56ページ).の「前提条件」の手順をすべて実行します。
2. 「HPE OOでのFIPS 140-2互換の構成」(58ページ) の「Javaセキュリティファイルのプロパティの構成」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、「HPE OOでのFIPS 140-2互換の構成」(58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

HPE OOは、サポートされるTLSプロトコルバージョンを定 義 するように構 成 できます。HPE 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. **<インストールフォルダー>/central/tomcat/conf/server.xml**ファイルを開きます。
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、

   `sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"` を

   `sslEnabledProtocols="TLSv1.2"` に変 更します。
4. サーバーを再 起 動します。

フローがCentral/RASのローカルファイルシステムにアクセスできなくする

フローがCentralまたはRASのローカルファイルシステムにアクセスできなくしたり、機 密 リソースにアクセスできるようにしたりするためには、CentralまたはRASのラッパー構 成 ファイルとjava.policyファイルを変 更 する必 要 があります。

注 : このシナリオを利 用 するには、フローでの権 限 またはフローに権 限 を付 与 する権 限 に加 え、デプロイメントとトリガー権 限 の両 方 が必 要 です。このような権 限 を持 つユーザーは、信 頼 できるユーザーである可 能 性 が高いです。

このシナリオから保 護 するには、以 下 を実 行 します。

1. CentralまたはRASのラッパー構 成 ファイル (**<インストールフォルダー>/<ras/central>/conf/<central/ras>-wrapper.conf**) で、次 のように `wrapper.java.additional.<nn>` パラメーターを追 加します。

   `wrapper.java.additional.<nn>=-Djava.security.manager`

   <nn> は最 後 の番 号 の次 の番 号 で置 き換 えます。
2. **java.policy**ファイル (**<インストールフォルダー>/java/lib/security/java.policy**にある) に、以 下 を追 加します。これにより、HPE OOが必 要 とする最 小 リソースへのアクセスを可 能 にしたり、機 密 データ

が含 まれているCentral/RASのローカルファイルシステムにアクセスできなくしたりすることができます。

```
grant codebase "file:${oo.home}/bin/-" {
                permission java.security.AllPermission;
};

grant codebase "file:${oo.home}/lib/-" {
                permission java.security.AllPermission;
};

grant codebase "file:${oo.home}/tomcat/-" {
                permission java.security.AllPermission;
};

grant codebase "file:${oo.home}/var/cache/-" {
                permission java.io.FilePermission "${oo.home}/var/logs",
"read, write";
};
```

フローがCentral/RASのローカルファイルシステム内 のリソースにアクセスできるようにするには、上 記を java.policyに指 定します。例：

```
grant codebase "file:${oo.home}/var/cache/-" {
                permission java.io.FilePermission
"C:\\users\\cathy\\foo.bat", "read, write, execute, delete";
                permission java.io.FilePermission "C:\\users\\cathy\\-",
"read,write,execute,delete"; // Recursive Example
                permission java.io.FilePermission "C:\\users\\cathy\\*",
"read,write,execute,delete"; // Flat Example
                   .....
};
```

Hewlett Packard
Enterprise