Operations Orchestration

ソフト ウェアバージョン: 2018.05
WindowsおよびLinux向 け

管 理ガイド

ドキュメントリリース日 : 2018年 5月 (英 語 版)
ソフト ウェアリリース日 : 2018年 5月

ご注意

Micro Focusとその関連会社およびライセンサー (Micro Focus) の製品およびサービスに関する保証は、当該製品およびサービスに付属の保証規定に明示的に記載されているものに限られます。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、Micro Focus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。

機密情報が含まれています。特に明記されている場合を除き、所有、使用、またはコピーするためには有効なライセンスが必要です。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

© Copyright 2011-2018 Micro Focus or one of its affiliates.

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。

目次

Operations Orchestrationの管理 8
使用開始時の管理タスク 8
認証の構成と有効化 8
認証の構成 8
認証の有効化 8
関連トピック 8
central-wrapper.confでのシステムロケールの設定 8
OOライセンスの管理 9
ライセンス情報の表示 9
新しいライセンスのインストール 10
ライセンスの購入 10
MS SQL照合順序の言語の変更 11
ログレベルの調整 11
LW-SSO設定の構成 12
同じマシンにインストールされたCentralとRAS 12
別のマシンにインストールされたRAS 13
サービス名の変更 13
Operations Orchestration Webインタフェースのタイムアウト制限の変更 14
データベースの削除スケジュール 15
削除フロー 15
削除API 15
通信プロトコルの指定 16
定期的な管理タスク 16
パスワードの暗号化と難読化 16
データベースパスワードの変更 17
データベースIPの変更 18
CentralまたはRASのホスト名/IPの変更 18
Quartzジョブのタイミングの調整 19
RAS - Centralの通信方向の転換 20
RAS側のCentral/ロードバランサーのURLの変更 20
Operations Orchestrationのバックアップと復元 21
バックアップ 21
復元 21
ディザスターリカバリのセットアップ 21
ディザスターリカバリの計画 21
ディザスターリカバリの実行 21
システムチューニング 22
データベースのメンテナンス 23
データベース環境のセットアップ 23
データベース環境の準備 23

概要 23
データベースクラスターの使用 24
データベースのセキュリティ 24
データベーススキーマ 25
OOデータベースのサイジング 25
ハードウェア要件 26
Microsoft SQL Serverデータベースのデプロイとメンテナンス 27
Microsoft SQL Serverデプロイメントのワークフロー 27
Microsoft SQL Serverのシステム要件 28
ハードウェア要件 28
ソフトウェア要件 28
テスト済みデプロイメントの例 29
言語サポート 29
SQL Serverの構成 30
Microsoft SQL ServerでのOOデータベースの手動作成 32
データベースオブジェクトの手動作成 33
Microsoft SQL Serverデータベースのメンテナンス 34
OOデータベースのバックアップ 34
バックアップ方法 34
バックアップ頻度 34
タイミング 34
保有期間 34
メンテナンス計画の作成 34
アップグレード/ロールバック手順 37
2012または2008R2からServer 2014またはServer 2016へのデータベースサーバーのアップグレード 37
常時オンのサポート 38
Oracleデータベースのデプロイとメンテナンス 39
Oracleデプロイメントのワークフロー 39
Oracleのシステム要件 39
ハードウェア要件 39
ソフトウェア要件 40
Oracleコネクター 40
テスト済みデプロイメントの例 41
言語サポート 41
Oracleデータベースの構成 41
OracleインスタンスでのOOデータベースの手動作成 43
データベースオブジェクトの手動作成 44
SIDまたはサービス名を使用したOracleへの接続 44
Oracleデータベースのメンテナンス 45
Operations Orchestrationデータベースのバックアップ 45
バックアップ方法 45
バックアップ頻度 45
タイミング 45
保有期間 45
メンテナンス計画の作成 45

アップグレード/ロールバック手順 48
MySQLデータベースのデプロイとメンテナンス 48
MySQLデプロイメントのワークフロー 48
MySQLのシステム要件 49
ハードウェア要件 49
ソフトウェア要件 49
MySQLコネクター 50
テスト済みデプロイメントの例 50
言語サポート 51
MySQLの構成 51
MySQLでのOOデータベースの手動作成 53
データベースオブジェクトの手動作成 55
MySQLデータベースのメンテナンス 56
OOデータベースのバックアップ 56
メンテナンス計画の作成 56
アップグレード/ロールバック手順 57
Postgresデータベースのデプロイとメンテナンス 57
Postgresデプロイメントのワークフロー 57
Postgresのシステム要件 58
ハードウェア要件 58
ソフトウェア要件 58
テスト済みデプロイメントの例 58
言語サポート 59
Postgresの構成 59
PostgresでのOOデータベースの手動作成 60
データベースオブジェクトの手動作成 61
Postgresデータベースのメンテナンス 61
OOデータベースのバックアップ 62
メンテナンス計画の作成 62
履歴データ削除用のユーティリティ 62
アップグレード/ロールバック手順 63
データベースの削除 64
削除フロー 64
削除のベストプラクティス 64
Microsoft SQL Serverに関する追加のガイドライン 65
Windows認証を使用したMicrosoft SQL Serverデータベースへのアクセス 65
Windows認証で稼働するOOの構成 65
常時オンで稼働するOOの構成 66
Oracleに関する追加のガイドライン 67
Oracle RAC 67
Single Client Access Name (SCAN) 68
Oracle RACで稼働するOOの構成 68
インストールウィザードの [Other database] オプション 69
Microsoft SQL Server名前付きインスタンスの例 70
Microsoft SQL ServerのWindows認証例 71
Oracle RAC例 73

Operations Orchestrationのセキュリティ保護とハードニング 75
概要 75
セキュリティの概要 76
セキュリティの概念 76
安全な実装およびデプロイメント 78
OOのセキュリティハードニング 78
物理的セキュリティ 78
ネットワークおよび通信のセキュリティ 78
通信チャネルのセキュリティ 78
管理インタフェースのセキュリティ 79
管理インタフェースへのアクセス 79
管理インタフェースのセキュリティ保護 - 推奨事項 79
ユーザー管理と認証 80
認証モデル 80
ユーザーのタイプ 80
認証の管理と構成 80
データベースの認証 80
Operations Orchestrationにおける権限付与 81
権限付与モデル 81
権限付与の構成 81
OO CentralのIDMモードでの構成 82
IDMセキュリティプロファイルへの切り替え 82
IDMプロパティファイルの更新 82
IDMモードでのCentralのリターンURLの構成 83
idm.propertiesファイルの例 83
IDMモードでは表示されないタブ 84
IDMにおける役割 84
LDAPのIDMモードでの構成 85
IDMサーバー用のLDAP構成の追加 85
データのバックアップ 86
Operations Orchestrationでの暗号化 87
暗号化モデル 87
暗号化の管理 87
デジタル証明書 88
コンテンツパックの機密情報 90
OOのハードニング 90
セキュリティハードニングの推奨事項 90
サーバーおよびクライアント証明書の使用 91
サーバー証明書を使用した通信の暗号化 92
Central TLSサーバー証明書の置き換え 92
関連トピック 93
Centralの信頼ストアへのCAルート証明書のインポート 93
RAS信頼ストアへのCAルート証明書のインポート 94
OOSH信頼ストアへのCAルート証明書のインポート 95
Studio信頼ストアへのCAルート証明書のインポート 96
証明書の失効ステータスの確認 97

キーストア/信頼ストアのパスワードの変更と暗号化/難読化 97
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 97
RAS、OOSH、およびStudioの信頼ストアのパスワードの変更 99
関連トピック 99
パスワードの暗号化と難読化 99
SSLサポート対象暗号からの脆弱性のある暗号の削除 100
HTTP/HTTPSポートの変更または無効化 101
ポートの値の変更 102
HTTPポートの無効化 102
HTTPSコネクターのトラブルシューティング 102
クライアント証明書の相互認証 103
クライアント証明書認証の構成 (Central) 103
クライアント証明書の構成の更新 (RAS) 105
Studio Remote Debuggerでのクライアント証明書の構成 106
OOSHでのクライアント証明書の構成 106
証明書ポリシーの処理 107
証明書のプリンシパルの処理 107
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 108
OOコンテンツパックへの署名 108
jarsignerユーティリティを使用してコンテンツパックに署名する 109
デジタル署名ファイルをPCKS＃12形式からjarsignerユーティリティで使用されるJKS形式に変換する 109
StudioツールのJARの整合性の検証 110
StudioツールのJARのデジタル署名を確認する 110
OOでのFIPS 140-2レベル1準拠の構成 111
アップグレードプログラムの前提手順 112
FIPS 140-2準拠の構成 113
ステップ1. Javaセキュリティファイルのプロパティの構成 113
ステップ2. encryption.propertiesファイルの構成とFIPSモードの有効化 114
ステップ3. FIPS準拠の暗号化の作成 114
ステップ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 115
ステップ5. OOの起動 115
関連トピック 115
FIPS暗号化の置き換え 115
CentralでのFIPS暗号化キーの変更 115
RAS暗号化プロパティの変更 116
関連トピック 116
TLSプロトコルの構成 117
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なるようにする 117
Javaセキュリティマネージャーの追加 118
Javaセキュリティマネージャーを追加するようRASを構成する 118
Javaセキュリティマネージャーを追加するようにCentral組み込みワーカーを構成する 119
HTTPSへのリダイレクトの有効化 119

ドキュメントのフィードバックを送信 121

Operations Orchestrationの管理

このセクションでは、以下のOperations Orchestration (OO) 構成タスクに関する情報を記載しています。

- 高度な構成タスクを実行する。
- OOで使用する4種類のデータベースを構成する。
- OOインスタンスを安全に展開および管理し、OOのセキュリティ強化を設定する。

使用開始時の管理タスク

このセクションでは、Operations Orchestrationのインストール後に、通常は1回だけ実行する管理タスクについて説明します。

認証の構成と有効化

Operations Orchestrationのインストール後に、認証メカニズムを構成し、認証を有効にする必要があります。

認証の構成

LDAP、SSO、内部ユーザーなど、Operations Orchestrationで使用する認証メカニズムを選択します。

認証の有効化

認証を有効にするには、Centralで、[**認証を有効にする**] チェックボックスをオンにします。

> **注:**
> [**認証を有効にする**] チェックボックスを使用できるのは、後から認証を無効にできるアクセス許可を持つ既存の内部ユーザーまたはLDAPユーザーがいる場合のみです。

関連トピック

システム構成のセットアップ

認証の有効化

central-wrapper.confでのシステムロケールの設定

Operations Orchestration (OO) システムがローカライズされている場合、システムロケールを反映するように**central-wrapper.conf**ファイルでプロパティを設定する必要があります。

1. テキストエディターで、**<インストールフォルダー>/central/ conf**の**central-wrapper.conf**ファイルを開きます。
2. 以下のプロパティを編集します。

   ```
   set.LANG=

   set.LC_ALL=

   set.LANGUAGE=

   wrapper.java.additional.<x>=-Duser.language=

   wrapper.java.additional.<x>=-Duser.country=
   ```

   たとえば、日本語の場合は次のようにします。

   ```
   set.LANG=ja_JP
   set.LC_ALL=ja_JP
   ```

3. **central-wrapper.conf**ファイルを保存します。

OOライセンスの管理

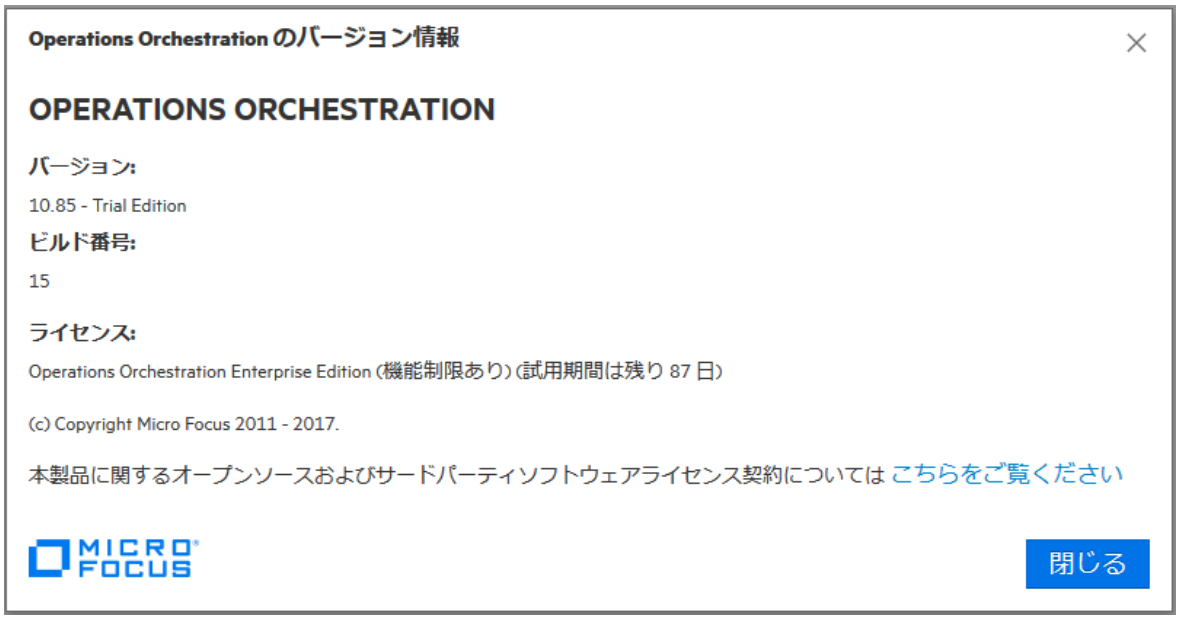
Operations Orchestration (OO) を初めてインストールする場合は、試用版ライセンスがインストールされます。Enterprise Editionライセンスを90日以内にインストールすることが必要になります。

OOでは、次のライセンスオプションが用意されています。

- **OO Enterprise Edition Trial**: 90日間実行される一時的なライセンスであり、すべての標準コンテンツパックが含まれています。試用ライセンスが期限切れになった後は、コンテンツパックを使用するためにはOO Enterprise Editionライセンスを購入する必要があります。このライセンスは、OOインストーラーに付属しているデフォルトのライセンスです。
- **OO Enterprise Edition**: 無期限に使用できるフルライセンスであり、すべての標準コンテンツパックが含まれています。このライセンスは、OO Community EditionとOO Enterprise Editionの試用ライセンスのロックを解除します。
- **OO Community Edition**: 部分ライセンスであり、BaseコンテンツパックとCEコンテンツパックが含まれています。毎月500の実行が可能です。
- **機能が限定されたOO Enterprise Edition**: OOバージョン10.80以降で導入されたすべての新機能を制限する部分的なライセンスです。以前のバージョンのOOで導入されたすべての機能は、OO Enterprise Editionライセンスで使用できます。

ライセンス情報の表示

Centralの右上隅にある [**情報**] ボタンをクリックして [**情報**] ポップアップウィンドウを表示することで、ライセンス情報を表示できます。このウィンドウには、ライセンスの現在の状態が表示されます。ライセンスに時間的な制限または実行数の制限があると、残り時間数または残りの実行数が表示されます。



新しいライセンスのインストール

新しいライセンスをインストールするには、Centralの [**システム構成**] > [**システム設定**] にある [**ライセンス**] タブを選択します。

ライセンスの購入

オプション1

OO Webサイトからライセンスをダウンロードします。

https://www.microfocus.com/licensing

オプション2

ライセンス管理システムを使用してライセンスを発行します。

- スタンドアロンのCentralインストールの場合、CentralサーバーのIPアドレスを使用してライセンスを発行します。
- CSAとOOを同じマシン上にインストールしている場合、そのIPアドレスに対して1つのライセンスを発行します。
- CSAとOOをそれぞれ異なるマシン上にインストールしている場合、2つのIPアドレスに対して2つのライセンスを発行します。
- クラスターの場合、ノードのいずれか1つを選択して、このノードのIPアドレスに対してライセンスを発行します。

  > **注:**
  > ライセンスをインストールする際は、必ず選択したノードのCentral UIをロードバランサーIPを介さずに開いてください。

- CSAとOOクラスターを配置している場合、CSAと選択したノードにそれぞれ1つずつ、合計2つのライセンスを発行します。

> **注:**
> OOのライセンスをインストールする際は、必ず選択したノードのCentral UIをロードバランサーIPを介さずに開いてください。

ライセンスの詳細については、セールスチームまたはアカウントマネージャーにお問い合わせください。

クラスターのライセンス

クラスターの場合は、いずれかのクラスターIP (ロードバランサーIPではない) に対してライセンスを発行するようにしてください。特定のノードに対してライセンスを直接適用します。これでクラスター全体のライセンスが設定されます。

MS SQL照合順序の言語の変更

Operations Orchestration (OO) Centralのインストール時に、英語の他にサポートする言語を選択するオプションがあります。この言語は、MS SQLの照合順序の言語およびコンテンツ (該当する場合) で使用されます。この言語サポートが必要なのは、たとえば、日本語で構成されるサーバーにpingを実行する必要がある場合などです。この言語を変更するには、**central-wrapper.conf**ファイルを編集します。

1. テキストエディターで、**<インストールフォルダー>/central/ conf**の**central-wrapper.conf**ファイルを開きます。
2. 言語プロパティを編集します。たとえば、`wrapper.lang=en_US`を`wrapper.lang=ja_JP`に変更します。
3. **central-wrapper.conf**ファイルを保存します。

関連トピック

言語サポート (29ページ)

ログレベルの調整

ログの記録によって Operations Orchestration (OO) のパフォーマンスが低下し、非常に大きなログファイルが生成される可能性があるため、OOを適切なログレベルで実行することが重要です。デフォルトのログレベルはINFOに設定されています。このレベルでは、パフォーマンスに影響を与えずに必要な情報を提供します。

また、ログに記録される情報の詳細度を、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るようになりました。

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
- `WARN` - より簡潔なログ情報
- `ERROR` - より簡潔なログ情報

ログの詳細度を調整するには:

1. **/<インストールフォルダー>/central/conf/log4j.properties**にある**log4j.properties**ファイルを開きます。

log4j.propertiesファイルの該当箇所のINFOをDEBUG、WARNまたはERRORに置き換えます。

例:

```
log.level=INFO

execution.log.level=DEBUG

deployment.log.level=DEBUG
```

> **注:**
> Tomcatサーバーでは、アクセスログは**/<インストールフォルダー>/central/var/logs**に保存されます。これらのログファイルのサイズが大きくならないようにするため、**localhost_access_log.yyyy-mm-dd.txt**という名前のファイルのサイズを監視し、必要に応じて古いファイルを削除することをお勧めします。WindowsまたはLinuxのいずれかのオペレーティングシステムで確立されているベストプラクティスに従ってください。
>
> たとえば、10日以上経過したファイルを削除するには、次のようにします。
>
> **Unix**
>
> ```
> find <OO_HOME>/central/var/logs/ -name " localhost_access*.txt"
> -type f -mtime +10 -exec rm -f {} \;
> ```
>
> **Windows**
>
> ```
> C:\Users\Administrator>forfiles /p "<OO_HOME>\central\var\logs" /s /m
> localhost_access*.txt /d -10 /c "cmd /c del @PATH"
> ```
>
> <OO_HOME>は、OOインストールフォルダーの場所です。
>
> OOのスケジュールまたはCronやWindows Schedulerなどを使用して、バックグラウンドでこれらのコマンドを実行するように設定できます。

LW-SSO設定の構成

OO 10.10以降をインストールする際に、LW-SSO設定をバージョン9.xからアップグレードするように選択した場合、LW-SSO設定は移行されますが、LW-SSOは9.xで有効になっていた場合でもOO 10.xでは無効になります。

後でLW-SSOを有効にすると、特定のシナリオで警告が表示されることがあります。ログから警告を消去するには、以下の手順に従って、完全修飾ドメイン名 (FQDN) を使用して管理URLプロパティを設定します。

同じマシンにインストールされたCentralとRAS

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。

1. RASプロセスを停止します。
2. **ras/conf/ras-wrapper.conf**ファイルで、次の内容を変更します。

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

   変更後

   ```
   wrapper.java.additional.<x>=-
   Dmgmt.url=<protocol>://<FullyQualifiedDomainName>:<port>/oo
   ```

3. RASプロセスを起動します。

別のマシンにインストールされたRAS

RASがCentralとは異なるマシンにインストールされ、LW SSOの設定が有効になっている場合は、RASのインストール中に、Centralの管理URLをIPアドレスではなく完全修飾ドメイン名で指定する必要があります。

LW-SSOを使用した別のアプリケーションとCentralの接続

LW SSOを介して他のアプリケーションをCentralに接続する場合は、Centralの管理URLを完全修飾ドメイン名で指定する必要があります。

1. Centralプロセスを停止します。
2. **central/conf/central-wrapper.conf**ファイルで、次の内容を変更します。

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

   変更後

   ```
   wrapper.java.additional.<x>=-
   Dmgmt.url=<protocol>://<FullyQualifiedDomainName>:<port>/oo
   ```

3. Centralプロセスを起動します。

サービス名の変更

サービス名と説明をデフォルト値から変更する場合は、**central-wrapper.conf**ファイルでサービス名と説明を編集します。

> **注:**
> この手順は、RASにも適用できます。RASでは、「Central」を「RAS」に置き換えてください。たとえば、**ras-wrapper.conf**、**ras\conf**などを使用します。

1. Centralサービスを停止します。
2. **central\bin**ディレクトリに移動します。デフォルトのパスは**C:\Program Files\Micro Focus\Operations Orchestration**です。
3. **uninstall-central-service.bat**を実行します。
4. サービスがアンインストールされていることを確認します。
5. **central\conf**ディレクトリに移動します。
6. **central-wrapper.conf**ファイルのコピーを作成します。

> **注:**
> この手順は、ロールバックが必要な場合、または変更されたファイルが使用できなくなった場合のバックアップとして元の構成を保存するためのオプションの手順です。

7. 必要に応じて次の行を変更して、**central-wrapper.conf**ファイルを編集します。

   ```
   wrapper.name

   wrapper.displayname

   wrapper.description
   ```

   例:

   ```
   # Name of the service

   wrapper.name=OOCentral

   # Display name of the service

   wrapper.displayname= Operations Orchestration Central

   # Description of the service

   wrapper.description=Operations Orchestration Central is the run time
   environment of Operations Orchestration.フローの実行、各種の実行の監視、レポートの
   生成などに使用されます。
   ```

8. **install-central-as-service.bat**を実行します。
9. サービスが新しい名前と説明でインストールされていることを確認します。
10. 新しいCentralサービスを起動します。

Operations Orchestration Webインタフェースのタイムアウト制限の変更

tomcat web.xmlファイルを設定することにより、Centralサーバーのデフォルトのセッションタイムアウト (30分) を変更できます。

1. **<インストールフォルダー>/central/tomcat/conf/web.xml**ファイルをXMLエディターで開きます。
2. 次の記述を探します。

   ```
   <!-- ==================== Default Session Configuration ================= -->
   <!-- You can set the default session timeout (in minutes) for all newly   -->
   <!-- created sessions by modifying the value below.                       -->

     <session-config>
         <session-timeout>30</session-timeout>
     </session-config>
   ```

3. **session-timeout**値を分単位で変更します。
4. Centralサーバーを再起動します。

データベースの削除スケジュール

Operations Orchestrationデータベースのサイズを制御するため、データベースの削除操作を1日目からスケジュールすることが重要です。

削除フロー

削除フローは、**Micro Focus Solutions**コンテンツパック(ITOM Marketplaceで入手可能)から入手できます。このコンテンツパックをデプロイし、必要な設定でフローを構成してから、Centralでスケジュールすることをお勧めします。

次の削除フローは、[**ライブラリ**] > [**統合**] > [**Micro Focus**] > [**Operations Orchestration**] > [**10.x**] > [**データベース**] に格納されています。

- **Purge Execution Summary** - 実行データを削除します。
  フローの実行を開始してすぐにこのフローをスケジュールすることをお勧めします。
- **Purge Debug Events** - Studio Remote Debuggerのイベントデータを削除します。
  Studio Remote Debuggerを使用する場合は、このフローをスケジュールします。
- **Purge Audit Records** - 監査が有効になっている場合、古い監査レコードを削除します。
  セキュリティ監査を有効にしている場合は、このフローをスケジュールします。
- **Purge Rerun Info** - 再実行データを削除します。
  フローで再実行ポイントを定義した場合は、このフローをスケジュールします。

これらの削除フローの詳細については、Centralのフローの説明を参照してください。

削除API

削除フローを使用する代わりに、APIを使用して削除機能を実行できます。

使用可能なAPIは次のとおりです。

- **DELETE /audit/records**: 監査が有効になっている場合、古い監査レコードを削除します。
- **DELETE /debugger-events**: Studio Remote Debuggerのイベントデータを削除します。
- **DELETE /executions**: バインドされた入力、出力、ステップログイベントなどの実行データを削除します。実行の数が多いとデータベースが最大テーブルサイズに達する可能性があるため、この実行データは定期的に削除する必要があります。

  > **注:**
  > この削除は、完了した実行のデータのみに影響します。

- **DELETE/executions/rerun**: 再実行データをデータベースから削除します。
- **DELETE /steps-log**: 削除する時間と実行回数に従ってステップデータを削除します。

削除APIを使用することで、データを必要に応じて手動で削除できます。また、これらのAPIを組み込んだ繰り返しフローをスケジューリングすることもできます。

通信プロトコルの指定

システム通知機能で使用する通信プロトコルを指定するには、`system.notifications.smtp.enabled.protocols`プロパティを**central-wrapper.conf**ファイルに追加します。方法:

1. `<インストールディレクトリ>\central\conf`ディレクトリにある**central-wrapper.conf**ファイルを開きます。
2. 次のように、`system.notifications.smtp.enabled.protocols`プロパティを追加します。

```
wrapper.java.additional.32=-DCentral.login.suspend=300000
wrapper.java.additional.33=-Doverride.startup.mode=LWSSO
wrapper.java.additional.34=-Dsystem.notifications.smtp.enabled.protocols=TLSv1.2 TLSv1.1

# Initial Java Heap Size (in MB)
wrapper.java.initmemory=512
```

`system.notifications.smtp.enabled.protocols`プロパティはオプションで、デフォルト値は`TLSv1.2 TLSv1.1 TLSv1 SSLv3`です。このデフォルト値は上書き可能で、スペース区切りの値のリストとして必要なプロトコルを指定できます。

たとえば、次のように、複数のプロトコルを指定できます。

```
wrapper.java.additional.34=Dsystem.notifications.smtp.enabled.protocols=<protocol 1> <protocol 2> <protocol 3>
```

定期的な管理タスク

このセクションでは、定期的にまたは状況に応じて実行する必要がある管理タスクについて説明します。

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>
```

パスワードの難読化

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読化するパスワードを指定します。

   難読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:

```
C:\Program Files\Micro Focus\Operations Orchestration\central\bin>encrypt-password.bat
Password (typing will be hidden):
Confirm password (typing will be hidden):
{ENCRYPTED}gAkPCLQsYDhoR1Y2q9bjCQ==

C:\Program Files\Micro Focus\Operations Orchestration\central\bin>
```

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。

データベースパスワードの変更

1. Centralが実行中の場合は、Centralサービスを停止します。
2. --password <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。<パスワード>はデータベースのパスワードです。

   > **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。

   例:

   ```
   <install-dir>/central/bin/encrypt-password --password <plain-text-pass>
   ```

3. 結果をコピーします。次のように表示されます。

   ```
   ${ENCRYPTED}<some_chars>.
   ```

4. **<installation_folder>/central/conf**フォルダーに移動し、**database.properties**ファイルを開きます。

db.passwordの値をコピーした値に変更します。必ずクラスターの各ノードでこれを実行してください。

> **注:**
> パスワードは暗号化されて保存されているため、{ENCRYPTED}という接頭辞を付けてください。=文字はエスケープする必要があります。たとえば、db.password=GFDGDS12\=\=のようになりま

> す。

データベースIPの変更

このセクションは、別のデータベースインスタンスを使用するために必要なOperations Orchestrationの設定について説明します。データベースの資格情報、スキーマ名、テーブルなど、すべてのデータベースパラメーターは同じである必要があります。

1. **<インストールフォルダー>/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;sendStringParametersAsUnicode\=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。

CentralまたはRASのホスト名/IPの変更

Centralのホスト名またはIPが変更された場合:

Centralでホスト名/IPが変更された場合は、いくつかの箇所で、このホスト名/IPアドレスを手動で更新する必要があります。

1. **<インストールフォルダー>/central/conf/central-wrapper.conf**ファイルを編集します。
2. `-Dmgmt.url`パラメーターを探し、正しいホスト名またはIPアドレスに更新します。例:

   ```
   -Dmgmt.url=http://[machine_name or IP]:8080/oo
   ```

3. **<インストールフォルダー>/ras/conf/ras-wrapper.conf**ファイルでも同じ`-Dmgmt.url`パラメーターを更新します。

   Centralと通信するすべてのRASで、この手順を実行する必要があります。
4. Central証明書を使用している場合は、新しいFQDNを使用して新しいCentral証明書を生成します。

   IPを使用することはお勧めしません。

   また、Centralキーストアの証明書を置き換える必要があります。
5. ロードバランサーのリンクが設定されている場合は、このリンクをCentralで更新します。

6. Centralと、影響のあるすべてのRASを再起動します。
7. Centralのホスト名またはIPアドレスを使用するリモートデバッグセッションを設定した場合は、Studioで接続URLを適切に変更する必要があります。

RAS IPが変更された場合:

RAS IPが変更された場合、この変更は、[**システム構成**] > [**トポロジ**] > [**ワーカー**] のCentral URLの表示に影響します。

- RASが「リバースRAS」の場合は、[**トポロジ**] > [**ワーカー**] タブでは無効になります。行を編集し、以前のIPを新しいIPに置き換える必要があります。
- RASが「標準RAS」の場合は、Centralに接続するため、変更は機能に影響しません。

関連トピック

リモートCentralのStudioでのデバッグ

Quartzジョブのタイミングの調整

Operations Orchestrationシステムでは、システムの保守のためにQuartzジョブを定期的に実行します。

各ジョブは一定時間実行され、設定された間隔で繰り返されます。ジョブトリガーの例を次に示します。

トリガー名	現在の繰り返し間隔	実行内容
onRolling:OO_EXECUTION_STATES_Trigger	4.5分	削除のため状態テーブルをロールする
queueCleanerTrigger	1分	キューテーブルを消去する
queueRecoveryTrigger	2分	システムの復旧が必要かどうかをチェックする
recoveryVersionTrigger	0.5分	復旧に使用するバージョンカウンター
splitJoinTrigger	1秒	終了した分割を結合する
onRolling: OO_EXECUTION_EVENTS_Trigger **注:** このトリガーは非推奨です。	12時間	削除のためイベントテーブルをロールする

パフォーマンスを向上させるため、これらのジョブのタイミングを微調整する場合は、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与える可能性があります。これらのトリガーを変更する前に、サービス担当者に相談してください。

1. [OO] タブを開きます。[**MBeans**] に、「**jobTriggersMBean**」という名前のオペレーションがあります。
2. このオペレーションを使用し、変更するトリガーの名前を使用して右側のタブに値を入力します。新しい繰り返し間隔の値を使用して、テーブルとまったく同じ名前を使用します。

これにより、ジョブのトリガー時間が変更されます。

> **注:**
> イベントの状態保持メカニズムは廃止されました(「**onRolling:OO_EXECUTION_EVENTS_Trigger**」を参照)。リモートデバッガーを使用する場合は、このジョブを構成できます。

RAS - Centralの通信方向の転換

RASのインストール時に、RASが、Centralへの接続を開始するのか(標準RAS)、Centralが接続を開始するのを待つのか(リバースRAS)を選択します。

この通信方向は、インストール後に変更することができます。

1. **<インストールフォルダー>/central/conf/central-wrapper.conf**ファイルを編集します。
2. `-register.ras`パラメーターを探します。
   - `-register.ras=true`の場合、RASがCentralへの接続を開始することを意味します(標準RAS)。
   - `-register.ras=false`の場合、Centralによる接続の開始をRASが待機することを意味します(リバースRAS)。
3. **<インストールフォルダー>/ras/conf/ras-wrapper.conf**ファイルでも同じ`-register.ras`パラメーターを更新します。

   Centralと通信するすべてのRASで、この手順を実行する必要があります。
4. Centralと、影響のあるすべてのRASを再起動します。
   - 標準RASが再起動すると、RASはCentralとの通信を開始し、CentralはRASの登録を更新します。
   - リバースRASの場合は、RASを登録するようにCentralを構成する必要があります。この作業はCentral UIの[**トポロジ**] > [**ワーカー**]タブで行います。

RAS側のCentral/ロードバランサーのURLの変更

Central/ロードバランサーのURLを設定するには、インストーラーを使用する方法が最良ですが、RASをインストールした後にURLを変更する必要がある場合は、**ras-wrapper.conf**ファイルを編集します。

たとえば、Central/ロードバランサーに対してRASをインストールし、Central/ロードバランサーのFQDNを変更した場合は、この操作が必要になります。RASがCentral/ロードバランサーと再度通信できるように、RASレベルで格納されているCentral/ロードバランサーのURLを変更する必要があります。

1. RASを停止します。
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. 次の行のURLを編集します。

   ```
   wrapper.java.additional.<x>=-Dmgmt.url=<http or https>://<FQDN>:<port>/oo
   ```
4. RASを再起動します。

Operations Orchestrationのバックアップと復元

Operations Orchestrationのバックアップと復元は、以下の手順で行います。

バックアップ

<インストールフォルダー>central\var\securityフォルダーと**<インストールフォルダー>central\conf\database.properties**ファイルのすべてのコンテンツをバックアップします。

復元

1. 既存のスキーマでCentralを新たにインストールします。インストールは、**Start Central**ステップで失敗します。
2. Centralサービスを停止し、Centralが稼働していないことを確認します。
3. **central/var/security**フォルダーを開きます。
4. **credentials.store**ファイルが存在する場合は、削除します。
5. **encryption_repository**と**encryption.properties**ファイルをバックアップバージョンで上書きします。
6. **central/conf/database.properties**ファイルをバックアップしたファイルで上書きします。
7. Centralサービスを起動します。

ディザスターリカバリのセットアップ

ディザスターリカバリをセットアップするには、以下の手順を実行します。

ディザスターリカバリの計画

ディザスターリカバリを有効にするには、次のデータを複製してセカンダリデータセンターを作成します。

- データベース構成
- スキーマ
- OO構成ファイルとセキュリティファイル: **<インストールフォルダー>\central\var\security**と**<インストールフォルダー>\central\conf**

ディスク複製やデータベース複製など、任意の方法で、複製を行うことができます。

ディザスターリカバリの実行

OOは、コールドディザスターリカバリをサポートしています。この方法では、プライマリデータセンターからセカンダリデータセンターに切り替えるために手作業による処理が必要です。この作業は、プライマリデータセンターの全体または一部に障害が発生した場合に必要になる場合があります。

1. すべてのノード (CentralとRAS) を再インストールして、OOサーバーを復元します。セカンダリデータセンター用に複製したデータベース構成、スキーマ、およびOO構成ファイルとセキュリティファイルを使用します。

> **注:**
> データベースは、**database.properties**ファイルを編集して構成できます。データベース接続構成の詳細については、「Operations Orchestrationのセキュリティ保護とハードニング」を参照してください。

2. データベース内の存在しない古いワーカーノードを削除します。
3. プライマリデータセンターでOOを無効にします。
4. セカンダリデータセンターに切り替えます。

関連トピック

Operations Orchestrationのセキュリティ保護とハードニング (75ページ)

システムチューニング

Centralの各ノードについて、次のパラメーターの1つ以上を構成できます。JVMスレッドとワーカースレッドは、RASの各ノードにも適用できます。

> **注:**
> 次のいずれかのパラメーターを変更する場合、構成済みのノードを再起動する必要があります。

- **JVMヒープサイズ** - **<インストールフォルダー>/<centralまたはras>/conf/**にある**central-wrapper.conf**ファイルと**ras-wrapper.conf**ファイルを構成して、Central/RASヒープの初期サイズと最大サイズを増やすことができます。

  ```
  wrapper.java.initmemory=<value in MB>

  wrapper.java.maxmemory=<value in MB>
  ```

  Centralでは4GB、RASでは2GBに構成することをお勧めします。
- **ワーカースレッド** - デフォルトでは、各ノードに20個のワーカースレッドがあります。フローに多数の並列またはマルチインスタンスレーンがある場合は、この繰り返し回数を徐々に増やすことをお勧めします。

  <インストールフォルダー>/<centralまたはras>/conf/にある**central-wrapper.conf**ファイルと**ras-wrapper.conf**ファイルで、ワーカースレッド数を設定できます。

  ファイルを開き、`-Dcloudslang.worker.numberOfExecutionThreads`プロパティを編集します。

  これは、10.22より前のバージョンではサポートされていなかった新しいプロパティです。したがって、このプロパティが存在しない場合は、次のように追加する必要があります。

  ```
  wrapper.java.additional.<next available number>=

  -Dcloudslang.worker.numberOfExecutionThreads=<new value>
  ```
- **データベース接続** - **<インストールフォルダー>/central/conf/**にある**database.properties**ファイルでデータベース接続数を増やすことができます。

  `db.pool.maxPoolSize`プロパティを編集します。接続数を100に変更することをお勧めします。

関連トピック

パフォーマンスとサイジングの情報

データベースのメンテナンス

データベースのメンテナンスは、OOの効率とスループットを維持するために重要です。

データベースのハウスキーピング(インデックスの再構築、テーブルの統計情報の更新など)の詳細については、「データベース環境のセットアップ」を参照してください。

関連トピック

データベース環境のセットアップ (23ページ)

データベース環境のセットアップ

このセクションでは、データベース管理者がOOを使用して4種類のデータベースを構成する方法について説明します。

データベース環境の準備

このセクションでは、Operations Orchestrationで使用されるデータベースの種類について説明します。

概要

「データベース」という用語は、使用するデータベースベンダー/テクノロジーにより、解釈が異なる場合があります。Oracleでは、「データベース」という用語は、データとメタデータを含むファイルのコレクションを意味します。1つのOracleデータベースには、複数のスキーマ(およびユーザー)が含まれる場合があります。Microsoft SQL Serverの「データベース」は、Oracleの「データベース」よりもOracleの「スキーマ」に近い概念です。

混乱を避けるため、このセクションでは、次の用語を使用します。

- インスタンス/サーバー – RDBMSサービスを提供するソフトウェアおよびメモリの構造
- データベース – テーブル、ビュー、インデックスなどを含むエンティティ。

OOでは、単一のデータベースを作成する必要があります。このデータベースは、データベースサーバー内の他のデータベースと共存できます。

OOデータベースは、次のデータベースサーバータイプのいずれかにセットアップできます。

- Microsoft SQL Server Standard/Enterprise (2008 R2/2012/2014/2016)
- Oracle 11gR2 Standard/Enterprise Server (RAC環境を含む)
- Oracle 12cR1 Standard/Enterprise Server - Regularインスタンス(非CDB) (RAC環境を含む)およびOracle 12cR2 Standard/Enterprise Server - Regularインスタンス(CDB)
- Postgres (9.3/9.4/9.5/9.6)
- MySQL Community/Standard/Enterprise Server (5.5/5.6/5.7)

言語サポート

OOは、任意のサポートされている言語 (英語、フランス語、ドイツ語、スペイン語、日本語、および簡体字中国語) でインストールして使用できます。目的の言語をサポートするには、データベースとデータベースサーバーを適切に構成する必要があります。

多言語環境でOOを使用する場合は、Unicode文字セットを使用するようにデータベースを構成することをお勧めします。詳細手順については、該当するデプロイメントのトピックを参照してください。

ユーザー入力を行う言語が英語以外に2つ (たとえば、ドイツ語と中国語) ある場合は、MS SQLは使用しないでください。その代替として、Oracle、MySQL、Postgresなどのデータベースを、OOで推奨されるUnicode構成で使用してください。

重要

- このセクションは、熟練したデータベース管理者を対象としています。使用するデータベースのタイプに詳しくない場合や、OOデータベースの作成や構成を行うのに必要な知識が不足していると思う場合は、データベースベンダーのドキュメントを参照し、このセクションで説明する操作の内容を十分に理解するようにしてください。
- OOデータベースは、接続にJava JDBCを使用します。お使いの環境で調整やセキュリティ対策が必要な場合は、JDBCドキュメント (またはデータベースベンダーのドキュメント) を参照し、JDBC接続URLの形式を正確に確認してください。
- このセクションでは、OOに必要なデータベース設定について説明します。このドキュメントで指定されていない設定については、デフォルト値をそのまま使用するか、組織のDBAが構成できます。

データベースクラスターの使用

データベースクラスターは、複数の種類の障害からデータベースを保護できるので、OOシステムの堅牢性を高めるために役立ちます。

OOは、データベース接続のフェールオーバーなどのデータベースクラスター関連の独自の機能は備えていません。使用されるJDBCパッケージの機能、およびSCAN/AGリスナーなどのデータベースクラスター環境の機能を利用します。

次の条件を満たす任意のタイプのデータベースクラスター環境と組み合わせてOOをインストールできます。

- 接続プールがサポートされている
- 単一の有効なデータベースURLを提供できる
- フェールオーバー中に信頼できるトランザクション処理を提供できる (単一の完全なトランザクションが完全な失敗または完全な成功になる必要があります)

OOインストーラーはデータベースクラスターに関するインストールオプションを提供しません。インストール中に、インストーラーに対して単純な/ "regular" 形式でデータベース接続を指定する必要があります。通常は、システムがインストールされた後で、**database.properties**構成ファイルでデータベースURLを適合させる必要があります。

データベースのセキュリティ

OOデータベースは、OOシステムの核となるものであり、ここには、OOシステム構成と企業の機密データが格納されています。次のガイドラインに従って、データベースに対して厳格なセキュリティ設定を実施することを強くお勧めします。

データベースベンダーおよびオペレーティングシステムベンダーによる推奨事項に従うことで、データベースサーバーをハードニングします。サーバーハードニングには、特に以下の事項があります。

- Linuxサーバーでの SSHアクセスを、十分に管理されたOSユーザーセットに制限する
- 厳格なOSユーザーパスワードポリシー (パスワードの長さ、複雑性、ロックアウトポリシーなど) を適用する
- システムへの不法な侵入 (ブレークイン) 試行の検出とレポートを実施するための監査システムを設定する

次のようにデータベースユーザー/ログインアカウントをハーディングします。

- パスワードポリシー (パスワードの長さ、複雑性、ロックアウトポリシーなど) を適用する
- アクセスを管理レベルのアカウントに制限する
- システムへの不法な侵入 (ブレークイン) 試行の検出とレポートを実施するための監査システムを設定する

ファイルシステムレベルでのデータベース暗号化は、OOのパフォーマンスに影響を与えない場合で、OOに完全に透過的である場合にかぎり、サポートされます。たとえば、パフォーマンスが低下したり、LOB空き領域の回収が妨げられない限り、Oracle TDEを使用できます。

データベーススキーマ

OOのデータベースは、常にデフォルトのデータベーススキーマを使用して作成されます。OOは、SQL Serverには**dbo**スキーマを使用し、PostgreSQLには**public**スキーマを使用します。現在、OOでは他のスキーマはサポートされていません。

OOデータベースのサイジング

OO 10.xでは、デフォルトですべてのフローおよびステップの実行データがデータベース内に保持されるので、前のフロー実行の包括的なデバッグを行うことができます。結果として、システムのスループットおよびフローの複雑さに応じてデータベースのサイズが増加します。

バージョン10.22以降では、システムの持続性レベルを設定することで、データベースに保持されるデータ量も制御できます。詳細については、『OO Tuning Guide』の「パフォーマンスとサイジングの情報」セクションを参照してください。

このセクションは、OOのインストールの準備に役立ちます。システムの規模 (Small/Standard/Enterprise) を推定することで、必要なディスク容量やデータベースによるメモリ (RAM) 使用量を算出し、追加のデータベースインストールパラメーターを特定できます。

1. 複雑さに従ってシステムの規模を推定します。

システム基準\規模	Small	Standard	Enterprise
フローあたりの平均ステップ数	50以下	100以上	1000以上
フローの平均時間	数秒から数分	1時間以上	最大24時間
フローあたりの平均ペイロードサイズ[1]	1KB以下	1MB以上	4MB以上

注: [1] –「ペイロード」という用語は相当なサイズのフロー/ステップのデータを意味します。たと

えば、フロー入力として使用されるCSVまたはXMLファイル、大規模なJSONオブジェクト/REST API呼び出しデータです。

2. 同時実行/頻度に従ってシステムの規模を推定します。

システム基準\規模	Small	Standard	Enterprise
1日あたりの平均フロー数	100未満	300以上	1000以上

次の表に、OOデプロイメントの規模に応じたディスク容量とメモリの要件を示します。

システム規模\パラメーター	OOデータベースディスクのサイズ要件	メモリ
Small	50GB	4GB
Standard	500GB	8GB
Enterprise	500GB～2TB	12GB

注:

- ディスク容量およびメモリの値は推定値です。実際のディスクとメモリの使用量は、データベースベンダーやデータベースサーバーの構成によって変わります。
- メモリ(RAM)は、データベースマシンで利用可能な全体のメモリ容量ではなく、データベースの推奨メモリを反映したものです。
- ディスク容量は、OOシステムの日常の運用および妥当な履歴データの保有に必要なディスク容量を反映したものです。これには、データベースバックアップは含まれていません。

  データベースサイズが環境上の制限を超えないようにするために、ITOM Marketplace上に公開されているストアドプロシージャーおよびMicro Focusソリューションコンテンツパックに備えられているデータベース削除フローを使用して、履歴データを定期的に消去することを強くお勧めします。
- OOデータベースのバックアップを保管するのに必要な追加のディスク容量は、バックアップポリシー(頻度や保有期間)に依存します。

ハードウェア要件

次の表に、それぞれのデータベースサーバーの推奨ハードウェア要件(CPUとメモリ)を示します。

注:
メモリの値は、データベースのメモリ使用量(マシン全体のメモリの一部)を反映したものです。

データベース\規模	Small/Standard				Enterprise			
	CPU数		RAM		CPU数		RAM	
	Small	Standard	Min	Rec	Min	Rec	Min	Rec
SQL Server	2	4	4GB	8GB	4	12	8GB	12GB
Oracle	2	4	4GB	8GB	4	12	8GB	12GB

データベース\規模	Small/Standard				Enterprise			
	CPU数		RAM		CPU数		RAM	
	Small	Standard	Min	Rec	Min	Rec	Min	Rec
MySQL	2	4	4GB	8GB	4	12	8GB	12GB
Postgres	2	4	4GB	8GB	4	12	8GB	12GB

Min = 最小値、**Rec** = 推奨値

上記の一般的なハードウェア要件に加えて、データベースごとの関連するハードウェア要件とソフトウェア要件のセクションも参照してください。

関連トピック

- Microsoft SQL Serverデータベースのデプロイとメンテナンス (27ページ)
- Oracleデータベースのデプロイとメンテナンス (39ページ)
- MySQLデータベースのデプロイとメンテナンス (48ページ)
- Postgresデータベースのデプロイとメンテナンス (57ページ)
- パフォーマンスとサイジングの情報

Microsoft SQL Serverデータベースのデプロイとメンテナンス

Microsoft SQL Serverを使用してOOをデプロイするには、既存のSQL Serverデータベースサービスが必要です。このセクションでは、データベースサービスの新規作成については扱いません。データベースサービスの新規作成が必要な場合は、Microsoft社が提供しているドキュメントを参照してください。このセクションには、SQL Serverの構成に関する推奨事項が記載されています。

データの削除とインデックスのメンテナンスジョブを効率的にスケジュールするためにSQL Server Agentサービスを使用することもお勧めします。

このセクションのトピックは、次のとおりです。

Microsoft SQL Serverデプロイメントのワークフロー

Microsoft SQL Serverを使用してOOをデプロイするには、次の手順を実行します。

1. **サイジングガイドラインを確認します**。詳細については、「データベース環境の準備」セクションの「OOデータベースのサイジング」トピックを参照してください。
2. **ハードウェア要件とソフトウェア要件を確認します**。詳細については、「Microsoft SQL Serverのシステム要件」セクションを参照してください。
3. **Microsoft SQL Serverを構成します**。詳細については、「SQL Serverの構成」を参照してください。
4. **Microsoft SQL Server上にOOデータベースを作成します**。詳細については、「Microsoft SQL ServerでのOOデータベースの手動作成」セクションを参照してください。

5. (オプション) **Windows認証をセットアップします**。詳細については、「Microsoft SQL Serverに関する追加のガイドライン」セクションの「Windows認証を使用したMicrosoft SQL Serverデータベースへのアクセス」を参照してください。この手順が必要なのは、SQL Server認証の代わりにWindows認証を使用する場合だけです。

関連トピック

- データベース環境の準備 (23ページ)
- Microsoft SQL Serverのシステム要件 (28ページ)
- SQL Serverの構成 (30ページ)
- Microsoft SQL ServerでのOOデータベースの手動作成 (32ページ)
- Microsoft SQL Serverに関する追加のガイドライン (65ページ)

Microsoft SQL Serverのシステム要件

このセクションでは、OOでMicrosoft SQL Serverを使用する場合のシステム要件について説明します。

ハードウェア要件

OOデータベースのサイジングガイドラインおよびハードウェア要件については、「データベース環境の準備」トピックの「OOデータベースのサイジング」および「ハードウェア要件」を参照してください。

Microsoft SQL Serverのハードウェア要件については、お使いのMicrosoft SQL Serverリリースおよびオペレーティングシステムのインストールのセクションを参照してください。

ソフトウェア要件

次の表に、OOでサポートされているMicrosoft SQL Serverリリースを示します。

Microsoft SQL Serverデータベースのリリース			
バージョン	**種類**	**32/64ビット**	**Service Pack**
2016	Standard	64ビット	1
	Enterprise	64ビット	1
2014	Standard	64ビット	1
	Enterprise	64ビット	1
2012	Standard	64ビット	2
	Enterprise	64ビット	2

Microsoft SQL Serverデータベースのリリース			
バージョン	**種類**	**32/64ビット**	**Service Pack**
2008 R2	Standard	x64	3
		x86	3
	Enterprise	x64	3
		x86	3

表に記載されたサービスパックのみをインストールしてください。『OOリリースノート』で特に記述がない限り、上記のサービスパックよりも新しいサービスパックもサポートされます。

サポートされるプラットフォームについては、Microsoftのドキュメントを参照してください。

テスト済みデプロイメントの例

次の表に、品質保証スタッフによる詳細なテスト済みのデプロイメント環境を示します。

データベースリリース			オペレーティングシステム
バージョン	**32/64ビット**	**Service Pack**	
Microsoft SQL Server 2016 Enterprise Edition	64ビット	0	Windows 2012 R2 Standard Edition (64ビット)
Microsoft SQL Server 2014 Enterprise Edition	64ビット	1	Windows 2012 Standard Edition (64ビット)
Microsoft SQL Server 2012 Enterprise Edition	64ビット	2	Windows 2012 Standard Edition (64ビット)
Microsoft SQL Server 2008 R2 Enterprise Edition	64ビット	3	Windows 2012 Standard Edition (64ビット)

言語サポート

Microsoft SQL Serverでは、他のデータベースと異なり、OOデータベースはUnicode照合順序を使用しません。

OOのインストール言語に応じて、次のいずれかの照合順序を使用します。

言語	データベース照合順序
英語	SQL_Latin1_General_CP1_CS_AS
日本語	Japanese_Unicode_CS_AS
簡体字中国語	Chinese_Simplified_Stroke_Order_100_CS_AS
ドイツ語	SQL_Latin1_General_CP1_CS_AS
フランス語	French_100_CS_AS
スペイン語	SQL_Latin1_General_CP1_CS_AS

注:
大文字と小文字を区別する照合順序でMS SQLデータベースが作成されている場合、テーブルやキーなどのオブジェクト名でも大文字と小文字が区別されるようになります。

たとえば、OO_STEP_LOG_BINDINGSテーブルに対して、`SELECT * FROM oo_step_log_bindings`のようなコマンドを使用すると、無効なオブジェクト名を使用しているとみなされます。

現在異なる照合順序を使用している場合は、OOの今後のバージョンをサポートするため、OOデータベースの照合順序を上記の照合順序のいずれかに変更することを強くお勧めします。

次に、既存のデータベース照合順序の変更手順の例を示します。

1. 管理者用ログイン(“sa”など)を使用してデータベースサーバーに接続します。
2. 変更を行うOOデータベースに対する既存のセッションをすべて切断します。

   重要: このデータベースのセッション数/接続数が0でない場合、コマンドは失敗します。

3. 次のコードを実行します(`my_database`を実際のデータベース名に変更します)。

```
USE [master]
GO
ALTER DATABASE [my_database] COLLATE Japanese_Unicode_CS_AS
GO
```

注:
この変更によって既存の列の照合順序が変わることはありません。新しい列またはテーブルはすべて、この時点からデフォルトで新しい照合順序を使用します。新しい照合順序の並べ替え規則は直ちに適用されます。つまり、新しい照合順序は並べ替え動作と今後のデータに影響し、既存のデータには影響しません。

ユーザー入力を行う言語が英語以外に2つ(たとえば、ドイツ語と中国語)ある場合は、MS SQLは使用しないでください。Oracle、MySQL、Postgresなどの代替のデータベースを使用し、Unicode構成を使用する必要があります。

関連トピック

データベース環境の準備 (23ページ)

SQL Serverの構成

このセクションでは、Microsoft SQL Serverおよびデータベース構成の設定について説明します。

OOデータベースは、クラスター環境を含む任意のSQL Server環境にインストールできます。

凡例:

- **必須**の構成オプション/値は**太字/オレンジ色**のフォントで表記します。
- **推奨**の構成オプション/値は**太字/紫色**のフォントで表記します。
- サポート構成オプション/値は標準フォントで表記し、コンマ区切りのリストで示す場合があります。
- コメントはイタリック体フォントで表記します。

Microsoft SQL Server 2008R2、2012、2014および2016	
サーバーオプション/機能	
構成アイテム	**サポート構成オプション**
サーバー構成オプション	デフォルト、別途指示がある場合を除く
インスタンス	デフォルト値、単一
認証モード	混合、Windows [1]
フルテキスト検索	(OOでは不要)

[1] 現時点でOO 10.xインストーラーはSQL認証のみサポートしています。Windows認証は後で構成できます。

Microsoft SQL Server 2008R2、2012、2014および2016			
インスタンス/サーバーオプション			
	必須	**推奨**	**サポート**
サーバーの順序照合		**SQL_Latin1_General_CP1_CS_AS**	任意の順序照合
ネットワークライブラリ	サーバー: **TCP/IP**クライアント: **TCP/IP**		
同時接続	**>=800**	**0** (無制限)	
最大サーバーメモリ	**>4GB**	**2,147,483,647** (デフォルト、無制限) 『OOインストールガイド』の「パフォーマンスとサイジングの情報」セクションに従い、システム規模に応じて4～12GBを割り当てる	

Microsoft SQL Server 2008R2、2012、2014および2016			
データベースオプション			
	必須	推奨	サポート
順序照合	「Microsoft SQL Serverのシステム要件」トピックの「言語サポート」セクションに表示される任意の照合。		
復旧モデル		完全	単純、完全
スナップショット分離を許可	True		
Is Read Committed Snapshot On	True		
自動圧縮	False		
統計の自動作成	True		

注:
SQL Serverは、デフォルトでスナップショット分離なしのREAD COMMITTEDトランザクション分離を使用します。[**スナップショット分離を許可**] フラグと[**Is Read Committed Snapshot On**] フラグを設定することが必須です。これ以外のトランザクション分離を使用すると、OOは正しく作動しません。

関連トピック

パフォーマンスとサイジングの情報

Microsoft SQL Serverのシステム要件 (28ページ)

Microsoft SQL ServerでのOOデータベースの手動作成

OOのセットアップ時には、OOインストーラーで新規データベースを自動作成するか、既存のデータベースを使用することができます。

インストール中に、**sysadmin**としてデータベースサーバーに接続する (つまり“sa” として接続する) 権限が与えられている場合は、[create the database/schema] オプションを使用してください。その場合はこのセクションはスキップできます。

このセクションでは、Microsoft SQL Server上でOOデータベース、ログイン、およびユーザーを手動作成する手順について説明します。

注:
この時点ではデータベース、ログイン、ユーザーのみが作成され、テーブルやインデックスなどのオブジェクトは作成されません。これらのオブジェクトはOOが最初に起動されたときに作成されます。

このセクションの操作が必要なのは、たとえば、セキュリティ上の制限により、OOのインストール時にシステム特権を持つログイン/ユーザー資格情報を使用しない場合などです。このような場合は、ユーザー (ま

たは組織のDBA) が最初にデータベース、ログイン、ユーザーを作成した後に、「低い権限」を使用してOOインストーラーで既存のデータベースに接続する必要があります。

データベースを作成するには、**CREATE DATABASE**のアクセス許可を持つログインを使用してSQL Serverインスタンスに接続する必要があります。

> **注:**
> - sysadminサーバーの役割のメンバーには、**CREATE DATABASE**のアクセス許可が自動的に付与されます。また、すべてのデータベースのdboにマップされます。
> - 次の手順は、熟練したMicrosoft SQL Serverデータベース管理者のみが行うようにしてください。
> - データベース作成ウィザード/GUIを使用する場合は、以下に示すT-SQLコードに対応したすべてのオプションを選択してください。たとえば、[新しいデータベース] ダイアログボックスの [**オプション**] ページ/[**その他のオプション**] ペイン/[**その他**] タブで [**スナップショット分離を許可**] をTRUEに設定します。
> - すべてのデータベース作成オプションを指定するのではなく、デフォルト値と異なるオプションのみを指定します。よくわからない場合は、デフォルト値を使用します。

固有の照合順序を選択するということは、SQL Serverを使用するOOシステムが、その固有の照合順序でサポートされる言語セットに制限されるということも意味します。たとえば、**SQL_Latin1_General_CP1_CS_AS**照合順序を使用する場合は、英語、ドイツ語、およびスペイン語の文字は使用できますが、日本語文字は使用できません。**Japanese_Unicode_CS_AS**を使用する場合は、フランス語のアクセント文字は適切に提示されません。各照合順序の完全な仕様については、Microsoft SQL Serverのドキュメントを参照してください。

データベースを作成するには、次の手順を実行します。

1. Microsoft SQL Serverに“sa”としてログインするか、**sysadmin**の役割または**CREATE DATABASE**のアクセス許可を持つ他のログインを使用してログインします。
2. **<インストールフォルダー>/central/bin/sql/MS-SQL/mssql_create_oo_db.sql**を実行し、データベース、ログイン、ユーザーが正常に作成されたことを確認します。スクリプトのパラメーターは、それぞれの環境に合わせて変更します。
3. (オプション) 新しいログインとユーザーでデータベースオブジェクトを作成できることを確認するには、OOで新しく作成されたログイン役割資格情報を使用してデータベースサーバーに接続し、**<インストールフォルダー>/central/bin/sql/MS-SQL/mssql_create_test_table.sql**を実行します。

   スクリプトが正しく実行され、エラーが表示されていないことを確認します。

データベースオブジェクトの手動作成

データベース、ログイン、およびユーザーの設定が済んだ後に、OOサービスが初めて起動されてデータベースに接続したときに、データベースオブジェクト (テーブル、インデックスなど) が作成されます。

OO用に作成されたユーザーがデータベースオブジェクトを作成または変更するアクセス許可を持っていない場合でも、データベースオブジェクトを手動で作成できます (DMLオペレーションのみに制限されます)。

手動による操作を行わずにOOでアップグレードオペレーションを実行できるようにするために、OOデータベースユーザーにDDL関連の権限を付与することをお勧めします。これは、これらのオペレーションでデータベース構造の変更が必要になる場合があるためです。

データベースオブジェクトを手動で作成するには、次の手順を実行します。

1. **docs\sql**の下にあるOOインストールzipファイルから**mssql.sql**ファイルを展開します。
2. OOデータベースのデータベースオブジェクトを作成および変更するアクセス許可を持っている適切なユーザーとしてMicrosoft SQL Serverにログインします。
3. **mssql.sql**ファイルを実行し、エラーが発生しないことを確認します。

Microsoft SQL Serverデータベースのメンテナンス

このセクションでは、データベースのバックアップ、データベース整合性のチェック、インデックス断片化の処理、およびデータベースの監視など、Microsoft SQL Server上に作成したOOデータベースに関する推奨されるメンテナンスタスクについて説明します。

OOデータベースのバックアップ

Microsoft SQL Serverデータベースは、**完全**復旧モデルまたは**単純**復旧モデルとして構成されます。これらの復旧モデルのいずれかを使用して、OOデータベースをバックアップできます。OOではすべての構成と運用履歴が1つのデータベースで管理されるため、常にデータベース全体をバックアップします。

OOのバックアップ計画を作成する際には、次のガイドラインを考慮してください。

バックアップ方法

バックアップ方法は、失われる可能性のある情報量やシステム復旧に使用できる時間など、ビジネス上の考慮事項に大きく依存します。特定の時点への復旧が必要な場合や、数時間分のデータ損失しか許されない場合は、完全復旧モデルを使用して、完全バックアップや差分バックアップを毎日実行し、ビジネス要件に応じてトランザクションログバックアップをN時間ごとに実行します。

データ損失に対する許容度が大きい場合は、単純復旧モデルを使用して、完全バックアップを毎日または毎週実行します。

バックアップ頻度

毎日バックアップを行うことをお勧めします (特に、OOを毎日使用/変更する場合)。

最低限、月に1回はバックアップします。

タイミング

OOが最もアクティブでない時間にバックアップをスケジュールします。

保有期間

保有期間はビジネスガイドラインや法規によって異なります。

メンテナンス計画の作成

OOデータベースのメンテナンスでは、インデックスの再構築や空き領域の回収などを行います。このセクションで説明するスクリプトとツールを使用して、OOデータベースの状態を正常に維持します。

OOデータベースのハウスキーピング用に提供されるユーティリティ

OOには、インデックスメンテナンス、統計情報メンテナンス、および履歴削除用のスクリプトが用意されています。これらのスクリプトではストアドプロシージャーが作成されます。これらは調整が可能で、定期的に実行するようにスケジュールできます。

> **注:**
> SQLスクリプトは、Centralサーバーインストール済み環境の**central/bin/sql**フォルダーにあります。

これらのプロシージャーを使用することをお勧めしますが、インデックスと統計情報が適切に維持されれば、会社のポリシーに従って別の方法を使用しても構いません。

インデックスのオンラインでの再構築 (OOシステムのダウンタイムなし) には、エンタープライズグレードのデータベースが必要です。オンラインでインデックスの再構築を行う際には、事前にMicrosoft SQL Serverのエンタープライズ版を使用していることを確認してください。

また、通常、メンテナンス操作では、データベースサーバーのリソースを余分に消費します。このため、OOの最もアクティブでない時間にメンテナンスをスケジュールする必要があります。

インデックスと統計情報のメンテナンスに使用するユーティリティ

ITOM Marketplaceから最新の**HPE_OO_DB_Maintenance.zip**パックをダウンロードし、一時フォルダーに解凍します。

OOメンテナンス用ストアドプロシージャーをインストールして使用するには、次の手順を実行します。

1. Microsoft SQL Serverに“sa”または**sysadmin**の役割の任意のメンバーとしてログインし**<インストールフォルダー>/central/bin/sql/MS-SQL/mssql_grant_view_server_state.sql**を実行して、OOユーザーが**dm_os_performance_counters**の動的管理ビュー (DMV) にアクセスできるようにします。
2. **<一時フォルダー>/HPE_OO_DB_Maintenance/MS-SQL/10.60_to_10.80/**にある**OOIndexMaintenance.sql**スクリプトと**OO_DB_MAINTENANCE_LOG.sql**スクリプト (必要な場合) を編集し、スクリプトの最初にある“`USE <your_db_name_here>`”を実際のOOデータベース名に置き換えます。たとえば、データベース名が“OOPROD”の場合は、“`USE OOPROD`”に置き換えます。

   この手順は省略しないでください。この手順を省略すると、プロシージャーが正しいデータベースに作成されません。
3. Microsoft SQL ServerにOOユーザーとしてログインします。
4. 次のT-SQLスクリプト、**OOIndexMaintenance.sql**と (任意で) OO_DB_MAINTENANCE_LOG.sqlを所定の順序で実行し、新規のオブジェクトが正常に作成されたことを確認します。
5. 以下の説明に従って、ストアドプロシージャーを調整します。

   <インストールフォルダー>/central/bin/sql/MS-SQL/mssql_execute_index_maintenance.sqlに、インデックスのメンテナンス手順の使用例があります。

- **@FragmentationXXX**はスクリプトの断片化レベル感度と、それぞれのケースでの対応策を設定します。これらのしきい値レベルと後続のアクションは、Microsoftのドキュメントで推奨されています。これらの値の調整は慎重に行ってください。
- **@SortInTempdb** ("Y" に設定した場合) では、インデックスの再編成/再構築時のソート操作をメモリ内ではなくtempdb内で実行して、パフォーマンスを向上させることができます。このオプションを使用する場合は、**tempdb**に十分な空き領域を確保する必要があります。
- **@Indexes**はメンテナンス操作でインデックスを対象に含める/除外するためのフィルターです。このフィルターは現状のままにして、すべてのインデックスを分析することをお勧めします。
- **@TimeLimit**はメンテナンス操作を完了するタイムアウト (秒) です。該当する場合は、メンテナンスウィンドウの範囲に従って設定します。

- **@LockTimeout**はオブジェクトロックを待機するタイムアウト (秒) です。指定された時間を経過すると、そのオペレーションは失敗し、プロシージャーは次のオブジェクトに進みます。
- **@LogToTable**では、メンテナンスオペレーションの結果をテーブルにログ記録するかどうかを指定します。これを使用すると、メンテナンスオペレーションを追跡記録でき、プロシージャーのデバッグに役立ちます。

  OO_DB_MAINTENANCE_LOG.sqlスクリプトを使用してテーブルを作成する必要があります。
- **@Execute**では、実際のオペレーション (インデックスの再構築など) を実行するかどうかを指定します。このパラメーターを 'N' に設定した場合、プロシージャーは "dry run" を実行し、関連オブジェクトの分析を表示します。

履歴データ削除用のユーティリティ

OO履歴削除用ストアドプロシージャーをインストールして使用するには、次の手順を実行します。

1. **<一時フォルダー>/HPE_OO_DB_Maintenance/MS-SQL/10.60_to_10.80/**にある**OOPurgeHistory.sql**スクリプトを編集し、スクリプトの最初にある "`USE <your_db_name_here>`" を実際のOOデータベース名に置き換えます。たとえば、データベース名が "OOPROD" の場合は、"`USE OOPROD`" に置き換えます。

   この手順は省略しないでください。この手順を省略すると、プロシージャーが正しいデータベースに作成されません。
2. Microsoft SQL ServerにOOユーザーとしてログインします。
3. T-SQLスクリプトの**OOPurgeHistory.sql**を実行し、新規のオブジェクトが正常に作成されたことを確認します。
4. 以下の説明に従って、ストアドプロシージャーを調整します。

   <インストールフォルダー>/central/bin/sql/MS-SQL/mssql_execute_purge_stored_procedure.sqlに、インデックスのメンテナンス手順の使用例があります。

- **@PurgeExecutionsOlderThan**では、プロシージャーの実行が開始された時間に対して相対的に、保存 (保護) 日数を指定します。デフォルトでは、90日間保存されます。最も古いレコードのデータから削除されます。
- **@PurgeExecutionsInBatchesOf**では、まとめて処理するフロー数を指定します。値を小さくすると、トランザクションが小さくてより頻繁になり、値を大きくすると、トランザクションが大きくて頻度が減ります。ほとんどのシステムで1000をお勧めします。
- **@ShouldPurgeExecutionSummary**では、**OO_EXECUTION_SUMMARY**テーブルを削除する必要があるかどうかを指定します。デフォルト値は "0" (このテーブルを削除しない) です。使用する領域は大きくないため、このテーブルにデータを保持することをお勧めします。関連するフローへの参照をすべて完全に削除する場合にのみ、"1" を使用してください。
- **@verbose**では、出力の詳細レベルを指定します。"0" は "quiet" 出力、"1" は通常出力、"2" は詳細情報出力に対応します。
- **@StopPurgingProcessAfter**はオペレーションを完了するタイムアウト (時間) です。該当する場合は、メンテナンスウィンドウの範囲に従って設定します。
- **@DeepClean**では、ディープクレンジングを実行するかどうかを指定します。たとえば、データベースが必要以上に大きくなる原因になる「孤立した」レコードを検索します。デフォルトは "0" (オフ) です。このフラグを "1" に設定すると、プロシージャーの実行時間が長くなりますが、タイムアウト制限は引き続き適用されます。
- **@DisableIndexes**は、削除オペレーションの間、特定のデータベースインデックスを無効にするかどうかを決定します。これらのインデックスは、削除オペレーションの終了時に再構築されます。この機能を使

用すると、フローのドリルダウン(ステップレベルのデバッグ)を行わず、削除オペレーションの速度が向上します。

アップグレード/ロールバック手順

OOをアップグレードまたはロールバックする前に、次の手順を実行します。

1. 不要なデータをデータベースから削除します。

   アップグレードやロールバックオペレーションを実行すると、フロー実行データが変換されます。データが少なければ、それだけアップグレードやロールバックにかかる時間が短くなります。すべてのフロー実行データが必要なデータである場合は、最初にデータベース全体をバックアップしてから、削除します。
2. できればデータベースを削除(不要なデータを削除)した後に、データベースをバックアップしてください。
3. このオペレーションを、データベースサーバーの非表示期間にスケジュールします。
4. 一時スペースやトランザクションログ、類似リソースなどの共有リソースが使用されていないことを確認します。

OOをアップグレードまたはロールバックした後で、次の手順を実行します。

1. メンテナンスおよび削除ストアドプロシージャーの該当するバージョンをインストールします。

   OOアップグレードユーティリティでは、ストアドプロシージャーと削除フローはアップグレードまたはロールバックされません。
2. データベースのメンテナンスと削除ルーチンを再スケジュールします。

2012または2008R2からServer 2014またはServer 2016へのデータベースサーバーのアップグレード

OO 10.60x以降のデータベースサーバーを2012または2008R2から2014または2016にアップグレードする場合は、別のJDBCコネクターjarを使用するためにOO Centralサーバーの構成を変更する必要があります。

SQL Server 2008R2と2012にはJTDS 1.3.0コネクターを使用したOOのインタフェースが用いられますが、2014または2016とのインタフェースにはMicrosoft JDBC 4.2を使用します。

SQL Serverデータベースサーバーを2014または2016にアップグレードする場合:

1. **central\conf**フォルダーの下にあるCentralサーバーの**database.properties**ファイルを編集します。
2. **jdbc.driver_class**の値を次のように変更します。

   変更前:

   net.sourceforge.jtds.jdbc.Driver

   変更後:

   com.microsoft.sqlserver.jdbc.SQLServerDriver
3. **jdbc.url**の値を次のように変更します。

   変更前:

   jdbc:jtds:sqlserver://db_host:db_port/db_name;sendStringParametersAsUnicode=true

   変更後:

   jdbc:sqlserver://db_host:db_port;databaseName=db_name;sendStringParametersAsUnicode=true

JDBC URLの編集と追加オプションの指定が完了したら、MicrosoftのJDBC 4.2のドキュメントを使用して、新しいURLの形式を正しく設定します。

4. OOシステム内のCentralサーバーごとに、上記のすべての手順を繰り返します。
5. OO Centralサーバーを再起動します。

OOを10.6x以降にアップグレードし、同時にSQL Serverデータベースを2014にアップグレードする場合は、次の手順を実行します。

1. 現在のデータベース(2008R2または2012)を使用して、OOのアップグレード手順を完了します。

   上記のセクションのすべての手順(事前の削除、バックアップなど)を通常どおり実行します。現在のデータベースを使用して、OOシステムの動作確認テストを実行することを強くお勧めします。
2. OOデータベースのバックアップ(完全、コピーのみバックアップを推奨)を作成します。必要に応じて、OOのログイン役割をバックアップします。
3. SQL Serverデータベースサーバーを2014または2016にアップグレードします。
4. 必要に応じて、OOのデータベースをバックアップから復元します(オプション)。

   OOのデータベースがデータベースサーバーと一緒にアップグレードされている場合は、復元する必要はありません。必要に応じて、ログイン役割権限とデータベース所有権を再構成します。

この時点で、JTDSではなくMicrosoft JDBCを使用するようにCentralサーバーを構成する必要があります。

常時オンのサポート

Microsoft SQL Server 2008R2および2012の「常時オン」は、高可用性/ディザスターリカバリソリューションを実装するために、レガシークライアント[1]としてOOによってサポートされます。

[1] OOではSQL Server 2008R2および2012の「MultiSubnetFailover」機能がサポートされていないため、クラスター/リスナー構成の**RegisterAllProvidersIP**が0に設定されている必要があります。

SQL Server 2014または2016「常時オン」への接続は、「MultiSubnetFailover」機能を含めてサポートされていますが、OOでは認められていません。

高可用性セットアップ

OOでは単一連絡点が必要なため、高可用性セットアップに可用性グループリスナー(AGリスナー)が存在していることが重要です。OOのデータベース接続は、1つのホスト名を含めた1つのURLで定義されています。

通常、OOは、可用性グループのプライマリレプリカ(読み取り/書き込み可能インスタンス)に接続するために、AGリスナーとインタフェースをとります。

データベースフェールオーバーシナリオの場合、データベース接続が失われている間、OOは新しい読み取り/書き込み可能インスタンスが接続を受け入れるまで、繰り返しデータベースへの接続を試行します。設定されたデータベースホスト名の別のIPアドレスへのリダイレクトは、AGリスナーとクラスター環境によって実行されます。

> **注:**
> OOでは、読み取り専用データベースは無用です(読み取り専用ルーティングも役に立ちません)。

ディザスターリカバリソリューション

ディザスターリカバリセットアップでは、OOをいったん停止して再構成(**database.properties**ファイルを編集)し、新しいデータベースインスタンスが利用可能になってから再度起動する必要があります。

関連トピック

Microsoft SQL Serverに関する追加のガイドライン (65ページ)

Operations Orchestrationのバックアップと復元

Oracleデータベースのデプロイとメンテナンス

Oracleを使用してOperations Orchestrationをデプロイするには、既存のOracleデータベースサービスが必要です。このセクションでは、データベースインスタンス/サービスの新規作成については扱いません。データベースインスタンス/サービスの新規作成が必要な場合は、Oracle社が提供しているドキュメントを参照してください。このセクションには、Oracleインスタンスの構成に関する推奨事項が記載されています。

このセクションのトピックは、次のとおりです。

Oracleデプロイメントのワークフロー

Oracleを使用してOOをデプロイするには、次の手順を実行します。

1. **サイジングガイドラインを確認します**。詳細については、「データベース環境の準備」セクションの「OOデータベースのサイジング」を参照してください。
2. **ハードウェア要件とソフトウェア要件を確認します**。詳細については、「Oracleのシステム要件」セクションを参照してください。
3. **Oracleデータベースを構成します**。詳細については、「Oracleデータベースの構成」セクションを参照してください。
4. **データベースを作成します**。詳細については、「OracleインスタンスでのOOデータベースの手動作成」セクションを参照してください。
5. (オプション) **OOをOracle RAC環境に接続します**。この手順は、Oracle RAC環境内でOOを使用する場合にのみ必要です。

関連トピック

- データベース環境の準備 (23ページ)
- Oracleのシステム要件 (39ページ)
- Oracleデータベースの構成 (41ページ)
- Microsoft SQL ServerでのOOデータベースの手動作成 (32ページ)

Oracleのシステム要件

このセクションでは、Operations OrchestrationでOracleを使用する場合のシステム要件について説明します。

ハードウェア要件

OOデータベースのサイジングガイドラインおよびハードウェア要件については、「データベース環境の準備」トピックの「OOデータベースのサイジング」および「ハードウェア要件」セクションを参照してください。

Oracleのハードウェア要件については、お使いのOracleリリースおよびオペレーティングシステムの、インストールのセクションを参照してください。

ソフトウェア要件

次の表に、OOでサポートされているOracleリリースを示します。

Oracleのリリース			
バージョン	**種類**	**32/64ビット**	**パッチセット**
11g R2	Standard	64ビット	11.2.0.1 ~ 11.2.0.4 [1]
	Enterprise	64ビット	11.2.0.1 ~ 11.2.0.4 [1]
12c R1 Regularインスタンス、非CDB	Standard	64ビット	12.1.0.1 ~ 12.1.0.2
	Enterprise	64ビット	12.1.0.1 ~ 12.1.0.2
12c R2 Regularインスタンス、CDB	Standard	64ビット	12.2.0.1
	Enterprise	64ビット	12.2.0.1

注: 表に記載されたパッチセットのみをインストールしてください。『OOリリースノート』で特に記述がない限り、上記のサービスパックよりも新しいサービスパックもサポートされます。

重要: [1] Oracleの“DATABASE PATCH SET UPDATE 11.2.0.4.6”を適用することを強くお勧めします。これにより、以前のOOバージョンへのロールバックの問題が解決されます。

サポートされるプラットフォームについては、Oracleのドキュメントを参照してください。

注: Oracle 12c R1は、後方互換性があるRegularインスタンスフォームでのみサポートされます。OO 10.xは、Oracle 12cコンテナーデータベース(CDB)への接続をサポートしません。

注: Oracle 12c RAC環境は、後方互換性があるフォームでのみサポートされます。SCANリスナーに対する複数のクラスターサブネットなどの新しい機能はサポートされません。

OOは、Oracle Data Guardで保護されたデータベースと互換性があります。OOでOracle Data Guardで保護されたデータベースを使用するには、Data Guard保護モードを「最大可用性」または「最大保護」のいずれかに設定する必要があります。Data Guard環境にファストスタートフェイルオーバーを構成して、フェイルオーバー時のOOサービスのダウンタイムを限定的にする必要があります。JDBC URLの要件は、Oracle RACの要件と同様です。

注: フェールオーバーの期間中に、データベースの変更が発生している間、OOが読み取り専用モード(新しいフローは実行できない)で短時間動作することがあります。また、フェールオーバー中は、履歴データが同期していない可能性があります。これは、フェールオーバーが完了すると解決されます。

Oracleコネクター

`jdbc connector.jar`ファイルはデフォルトでは提供されません。インストール時に、インストーラーに対してコネクターを提供します。以前のバージョンのOOからアップグレードしても、コネクターは削除されません。

使用しているコネクターのバージョンがご使用のデータベースサーバーバージョンと完全に互換性があることを確認してください。推奨されるドライバーは、Oracle JDBCドライバーのバージョン7-12.1.0.2です。

テスト済みデプロイメントの例

次の表に、品質保証スタッフによる詳細なテスト済みのデプロイメント環境を示します。

データベースリリース			オペレーティングシステム
バージョン	32/64ビット	パッチセット	
Oracle 11g R2 Enterprise Edition	64ビット	11.2.0.4.6	Red Hat Enterprise Linux 6.5 (64ビット)
Oracle 11g R2 Enterprise Edition	64ビット	11.2.0.4.0	Windows 2012 Standard Edition (64ビット)
Oracle 12c R1 Enterprise Edition	64ビット	12.1.0.2	Red Hat Enterprise Linux 6.5 (64ビット)
Oracle 12c R2 Enterprise Edition	64ビット	12.2.0.1	Oracle Enterprise linux 7.3 (64ビット)

言語サポート

Oracleインスタンスの文字セットは**AL32UTF8**に設定する必要があります。これにより、任意のUnicode文字 (すべての言語のほぼすべての一般文字) の使用が可能になります。

関連トピック

データベース環境の準備 (23ページ)

Oracleデータベースの構成

このセクションでは、Oracleインスタンスおよびデータベース構成の設定について説明します。

OOデータベースは、Oracleクラスター環境 (Oracle RACなど) にインストールできます。

凡例:

- **必須**の構成オプション/値は**太字/オレンジ色**のフォントで表記します。
- **推奨**の構成オプション/値は**太字/紫色**のフォントで表記します。
- サポート構成オプション/値は標準フォントで表記し、コンマ区切りのリストで示す場合があります。
- コメントはイタリック体フォントで表記します。

Oracle Database 11gR2/12cR1/12cR2			
インスタンス/サーバーオプション			
インスタンス構成オプション	デフォルト、別途指示がある場合を除く		
	必須	推奨	サポート
PROCESSES	>=500		
SESSIONS	>=555		
TIMED_ STATISTICS		TRUE	TRUE、FALSE
OPEN_CURSORS	>=900		
Shared/Dedicatedサーバー		Dedicated	Dedicated、Shared
UNDO_ MANAGEMENT		AUTO	Automatic、Manual
Undoサイズ	>=4GB	6GB～10GB	
メモリ管理		ASMM	AMM、ASMM
MEMORY_TARGET		0 (無効)	>= 5G (AMM用)
SGA_TARGET		8G～12G	>= 4G (ASMM用)
SGA_MAX_SIZE		8G～12G	>= 4G (ASMM用)
PGA_AGGREGATE_ TARGET		1G～2G	>= 500M (ASMM用)

注: 値はすべてOOで必要なリソースを反映したものです。OOがOracleインスタンスを他のユーザーと共有する場合は、他のユーザーによる使用状況にこれらの値を追加する必要があります。

Oracle Database 11gR2/12cR1/12cR2			
インスタンス/サーバーオプション			
	必須	推奨	サポート
ファイルシステム			ASM、任意
ストレージオプション		ローカル管理表領域	
		自動セグメント領域管理 (ASSM)	
		自動ローカルエクステント管理	
ARCHIVELOGモード		ARCHIVELOG	ARCHIVELOG、NOARCHIVELOG

Oracle Database 11gR2/12cR1/12cR2			
インスタンス/サーバーオプション			
	必須	推奨	サポート
REDOログ合計サイズ	>=600MB	1GB	

注: 値はすべてOOで必要なリソースを反映したものです。OOがOracleインスタンスを他のユーザーと共有する場合は、他のユーザーによる使用状況にこれらの値を追加する必要があります。

- 範囲で示された値を決定するには、『OO管理ガイド』の「パフォーマンスとサイジングの情報」セクションを参照してください。

関連トピック

パフォーマンスとサイジングの情報

OracleインスタンスでのOOデータベースの手動作成

OOのセットアップ時には、OOインストーラーで新規データベースを自動作成するか、既存のデータベースを使用することができます。

インストール中に、DBAとしてデータベースサーバーに接続する(“SYSTEM”として接続する)権限が与えられている場合は、[create the database/schema]オプションを使用してください。その場合はこのセクションはスキップできます。

注:
「データベース」という用語が使われていても、Oracleでは「ユーザー」と解釈した方がよい場合があります。

このセクションでは、Oracleインスタンス上でOOデータベースを手動作成する手順について説明します。

注:
この時点ではデータベースのみが作成され、テーブルやインデックスなどのオブジェクトは作成されません。これらのオブジェクトはOOが最初に起動されたときに作成されます。

このセクションの操作が必要なのは、たとえば、セキュリティ上の制限により、OOのインストール時にシステム特権を持つユーザー資格情報を使用しない場合などです。このような場合は、ユーザー(または組織のDBA)が最初にユーザー(データベース)を作成した後に、基本的な権限を使用してOOインストーラーで既存のデータベースに接続する必要があります。

データベースを作成するには、systemユーザーなどの、**CREATE USER**のシステム権限を持つログインを使用してOracleインスタンスに接続する必要があります。

注:
- DBAの役割を持つユーザーには、新規ユーザーを作成するのに十分な権限があります。
- 次の手順は、熟練したOracleデータベース管理者のみが行うようにしてください。

- データベース作成ウィザード/GUIを使用する場合は、以下に示すSQLコードに対応したすべてのオプションを選択してください。
- すべてのデータベース作成オプションを指定するのではなく、デフォルト値と異なるオプションのみを指定します。よくわからない場合は、デフォルト値を使用します。

データベースを作成するには、次の手順を実行します。

1. “system” またはDBAの役割を持つ別のユーザーとしてOracleにログインします。
2. **<インストールフォルダー>/central/bin/sql/Oracle/oracle_create_oo_db.sql** SQLスクリプトを編集して実行し、データベースが正常に作成されたことを確認します。
3. (オプション) 新しいユーザーでデータベースオブジェクトを作成できることを確認するには、OOユーザーとしてOracleインスタンスに接続して、次のように編集、実行します。**<インストールフォルダー>/central/bin/sql/Oracle/oracle_create_test_table.sql**

   スクリプトが正常に実行され、エラーや警告が表示されていないことを確認します。

データベースオブジェクトの手動作成

データベースの設定が済んだ後に、OOサービスが初めて起動されてデータベースに接続したときに、データベースオブジェクト (テーブル、インデックスなど) が作成されます。

OO用に作成されたユーザーがデータベースオブジェクトを作成または変更するアクセス許可を持っていない場合、データベースオブジェクトを手動で作成できます (DMLオペレーションのみに制限されます)。

手動による操作を行わずにOOでアップグレードオペレーションを実行できるようにするために、OOデータベースユーザーにDDL関連の権限を付与することをお勧めします。これは、これらのオペレーションでデータベース構造の変更が必要になる場合があるためです。

データベースオブジェクトを手動で作成するには、次の手順を実行します。

1. **docs\sql**の下にあるOOインストールzipファイルから**oracle.sql**ファイルを展開します。
2. **oracle.sql**ファイルを編集し、各オブジェクトにOOユーザーのプレフィックスを指定し、オブジェクトが確実にOOユーザーの下に作成されるようにします。
3. OOデータベースのデータベースオブジェクトを作成および変更するアクセス許可を持っている適切なユーザーとしてOOデータベースに接続します。
4. **oracle.sql**ファイルを実行し、エラーが発生しないこと、およびすべてのオブジェクトがOOユーザーによって作成および所有されていることを確認します。

SIDまたはサービス名を使用したOracleへの接続

Oracleデータベースサーバーに接続するには、SID (システムID) またはサービス名を指定する必要があります。バージョン10.20以降のOOでは、(インストールウィザードまたはサイレントインストールを使用した) インストール中にサービス名を指定できます。

以下の例では、Centralの「**central\conf**」フォルダーにある**database.properties**ファイルに表示される各オプションのJDBC URLの形式を示します。

SIDを使用したOracleインスタンスへの接続:

```
jdbc.url=jdbc\:oracle\:thin\:@DB_HOSTNAME_OR_IP\:PORT\:SID
```

サービス名を使用したOracleインスタンスへの接続:

```
jdbc.url=jdbc\:oracle\:thin\:@//DB_HOSTNAME_OR_IP\:PORT/SERVICE_NAME
```

Oracleデータベースのメンテナンス

このセクションでは、データベースのバックアップ、データベース整合性のチェック、インデックス断片化の処理、およびデータベースの監視など、Oracle上で作成したOOデータベースに関する推奨されるメンテナンスタスクについて説明します。

Operations Orchestrationデータベースのバックアップ

Oracleデータベースは、**expdp**や**RMAN**などのツールを使用してバックアップできます。OOデータベースは、データベース全体がバックアップされる限り、どのような種類の方法/ツールを使用してバックアップしても構いません。

OOのバックアップ計画を作成する際には、次のガイドラインを考慮してください。

バックアップ方法

バックアップ方法は、失われる可能性のある情報量やシステム復旧に使用できる時間など、ビジネス上の考慮事項に大きく依存します。特定の時点への復旧が必要な場合や、数時間分のデータ損失しか許されない場合は、完全復旧モデルを使用して、完全バックアップや差分バックアップを毎日実行し、ビジネス要件に応じてトランザクションログバックアップを "n" 時間ごとに実行します。

バックアップ頻度

毎日バックアップを行うことをお勧めします (特に、OOを毎日使用/変更する場合)。

最低限、月に1回はバックアップします。

タイミング

OOが最もアクティブでない時間にバックアップをスケジュールします。

保有期間

保有期間はビジネスガイドラインや法規によって異なります。

メンテナンス計画の作成

OOデータベースのメンテナンスでは、インデックスの再構築や空き領域の回収などを行います。このセクションで説明するスクリプトとツールを使用して、OOデータベースの状態を正常に維持します。

OOデータベースのハウスキーピング用に提供されるユーティリティ

OOには、インデックスメンテナンス、統計情報メンテナンス、および履歴削除用のスクリプトが用意されています。これらのスクリプトでは、調整が可能で、定期的に実行するようにスケジュールできるストアドプロシージャーを含むパッケージが作成されます。

> **ヒント:** これらのプロシージャーを使用することをお勧めしますが、インデックスと統計情報が適切に維持されれば、会社のポリシーに従って別の方法を使用しても構いません。

> **注:** インデックスのオンラインでの再構築 (OOシステムのダウンタイムなし) には、エンタープライズグレードのデータベースが必要です。オンラインでインデックスの再構築を行う際には、事前にOracleのエンタープライズ版を使用していることを確認してください。

> **注:** また、通常、メンテナンス操作では、データベースのリソースを余分に消費します。このため、OOの最もアクティブでない時間にメンテナンスをスケジュールする必要があります。

インデックスと統計情報のメンテナンスに使用するユーティリティ

ITOM Marketplaceから最新の**HPE_OO_DB_Maintenance.zip**パックをダウンロードし、一時フォルダーに解凍します。

OOメンテナンス用ストアドプロシージャーをインストールして使用するには、次の手順を実行します。

1. 「システム」またはDBAの役割を持つ他のユーザーとしてOracleにログインし、**<インストールフォルダー>/central/bin/sql/Oracle/oracle_privs_for_index_maintenance.sql**を編集して実行します。次の手順で作成したストアドプロシージャーに、インデックスの分析と再構築を行う明示的な (役割ベースではない) 権限があることを確認するには、これらのシステム権限が必要です。
2. **<一時フォルダー>/HPE_OO_DB_Maintenance/Oracle/10.60_to_10.80/**フォルダーを開き、OO用に作成されたユーザーとしてOracleデータベースにログインして**HPE_OO_DB_MAINT.sql**スクリプトを実行します。新しいパッケージとプロシージャーが正常に作成されたことを確認します。

   同じフォルダー内の**oracle_execute_IndexMaintenance.sql**に、このプロシージャーの使用例が示されています。

- **pMaxHeight (IN)** - インデックスを再構築するインデックスの高さの最小値のしきい値。Oracleのドキュメントでは3が推奨されています。それよりも小さな値を指定すると、不必要な再構築オペレーションが発生する場合があります。
- **pMaxLeafsDeleted (IN)** - インデックスを再構築する最小削除リーフ数のしきい値。Oracleのドキュメントでは15が推奨されています。それよりも小さな値を指定すると、不必要な再構築オペレーションが発生する場合があります。
- **pRebuild (IN)** - インデックスを再構築 (1) またはdry-runを実行 (0) します。dry-runでは、インデックスの再構築に関する推奨事項が表示されるだけです。
- **pReturnValue (OUT)** - 再構築されたインデックスの数。

> **注:**
> オンラインでのインデックス再構築は、Enterprise Editionを使用している場合にのみ行うようにしてください。Enterprise Edition以外を使用している場合にインデックスの再構築を行うと、テーブルやインデックスがロックされ、OOの動作に差し支える可能性があります。

履歴データ削除用のユーティリティ

OO履歴削除用ストアドプロシージャーをインストールして使用するには、次の手順を実行します。

1. **<一時フォルダー>/HPE_OO_DB_Maintenance/Oracle/10.60_to_10.80**フォルダーを開き、OOで作成されたユーザーとしてOracleデータベースにログインして**HPE_OO_DB_MAINT.sql**スクリプトを実行します。新しいパッケージとプロシージャーが正常に作成されたことを確認します。

   同じフォルダー内の**oracle_execute_PurgeHistory.sql**に、このプロシージャーの使用例が示されています。

   - **pPurgeExecutionsOlderThan**では、プロシージャーの実行が開始された時間に対して相対的に、保存 (保護) 日数を指定します。最も古いレコードのデータから削除されます。このパラメーターにはデフォルト値がないので値を指定する必要があります。

- **pPurgeExecutionsInBatchesOf**では、各バッチで処理する最大フロー数を指定します。デフォルト値は10000です。
- **pShouldPurgeExecutionSummary**では、**OO_EXECUTION_SUMMARY**テーブルを削除する必要があるかどうかを指定します。デフォルト値は "0" (このテーブルを削除しない) です。使用する領域は大きくないため、このテーブルにデータを保持することをお勧めします。関連するフローへの参照をすべて完全に削除する場合にのみ、"1" を使用してください。
- **pVerbose**では、出力の詳細レベルを指定します。"0" は “quiet” 出力、"1" は通常出力、"2" は詳細情報出力に対応します。該当する場合は、メンテナンスウィンドウの範囲に従って設定します。
- **pStopPurgingProcessAfter**はオペレーションを完了するタイムアウト (時間) です。
- **pDeepClean**では、ディープクレンジングを実行するかどうかを指定します。たとえば、データベースが必要以上に大きくなる原因になる「孤立した」レコードを検索します。デフォルトは “0” (オフ) です。このフラグを “1” に設定すると、プロシージャーの実行時間が長くなりますが、タイムアウト制限は引き続き適用されます。
- **@DisableIndexes**は、削除オペレーションの間、特定のデータベースインデックスを無効にするかどうかを決定します。これらのインデックスは、削除オペレーションの終了時に再構築されます。この機能を使用すると、フローのドリルダウン (ステップレベルのデバッグ) を行わず、削除オペレーションの速度が向上します。

LOB領域を回収するためのユーティリティ

Oracleデータベースは、LOBセグメントによって占有されている領域の解放に関しては処理が遅くなります。データベースサーバーの構成によっては、LOBセグメント領域が削除オペレーションの直後には解放されない場合があります。LOB領域はデータベースのサイズに大きな影響を与える可能性があるため、LOBセグメントを強制的に解放するために、削除後オペレーションを追加できます。

コマンドの一般的な構造は、次のとおりです。

ALTER TABLE FOO MODIFY LOB (LOB_COLUMN_NAME) (shrink space);

関係するテーブルと列のリストは、OOバージョンごとに異なる場合があります。

OO LOB領域回収用ストアドプロシージャーをインストールして使用するには、次の手順を実行します。

<一時フォルダー>/HPE_OO_DB_Maintenance/Oracle/10.60_to_10.80フォルダーを開き、OOで作成されたユーザーとしてOracleデータベースにログインして**HPE_OO_DB_MAINT.sql**スクリプトを実行します。新しいパッケージとプロシージャーが正常に作成されたことを確認します。

同じフォルダー内の**oracle_execute_ForceLobShrink.sql**に、このプロシージャーの使用例が示されています。

注:

圧縮オペレーションが成功するためには、以下のようないくつかの条件が満たされる必要があります。

- TABLESPACEとデータベーステーブルを圧縮してはならない。
- すべてのテーブルでROW MOVEMENTが有効になっている必要がある。
- TABLESPACEがAUTOセグメント領域管理として構成されている必要がある。
- LOBストレージはSECUREFILE以外にする。

条件一式については、Oracleのドキュメントを参照してください。

かなり大きなサイズのLOBセグメント (数10または数100ギガバイト) の圧縮は、完了するまで時間がかかったり、相当な量のデータベースリソース (CPUおよびI/O) が消費されたりする可能性があります。

したがって、このプロセスは、データベースサーバーが最もアクティブでないときに実行することをお勧めします。

アップグレード/ロールバック手順

OO 10.5x以降を以前のバージョンにロールバックするには、11.2.0.4.6へのOracle 11gR2データベースサーバーのパッチが必要です。ロールバックオペレーションを実施する前に、Oracleの“DATABASE PATCH SET UPDATE 11.2.0.4.6”を適用します。

Operations Orchestrationをアップグレードまたはロールバックする前に、次の手順を実行します。

1. 不要なデータをデータベースから削除します。

   アップグレードやロールバックオペレーションを実行すると、フロー実行データが変換されます。データが少なければ、それだけアップグレードやロールバックにかかる時間が短くなります。すべてのフロー実行データが必要なデータである場合は、最初にデータベース全体をバックアップしてから、削除します。
2. できればデータベースを削除 (不要なデータを削除)した後に、データベースをバックアップしてください。
3. このオペレーションを、データベースサーバーの非表示期間にスケジュールします。
4. 一時スペースやUNDO/REDO、類似リソースなどの共有リソースが使用されていないことを確認します。

OOをアップグレードまたはロールバックした後で、次の手順を実行します。

1. メンテナンスおよび削除ストアドプロシージャーの該当するバージョンをインストールします。

   OOアップグレードユーティリティでは、ストアドプロシージャーと削除フローはアップグレードまたはロールバックされません。
2. データベースのメンテナンスと削除ルーチンを再スケジュールします。

MySQLデータベースのデプロイとメンテナンス

MySQLを使用してOperations Orchestrationをデプロイするには、既存のMySQLデータベースサービスが必要です。このセクションでは、データベースサービスの新規作成については扱いません。データベースサービスの新規作成が必要な場合は、MySQLのドキュメントを参照してください。このセクションには、MySQLの構成に関する推奨事項が記載されています。

このセクションのトピックは、次のとおりです。

MySQLデプロイメントのワークフロー

MySQLを使用してOperations Orchestration (OO)をデプロイするには、次の手順を実行します。

1. **サイジングガイドラインを確認します。**
2. **ハードウェア要件とソフトウェア要件を確認します。**
3. **MySQLを構成します。**
4. **MySQL上にOOデータベースを作成します。** MySQLでのOOデータベースの手動作成 (53ページ)を参照してください。

関連トピック

データベース環境の準備 (23ページ)

MySQLのシステム要件 (49ページ)

MySQLの構成 (51ページ)

MySQLでのOOデータベースの手動作成 (53ページ)

MySQLのシステム要件

このセクションでは、OOでMySQLを使用する場合のシステム要件について説明します。

ハードウェア要件

OOデータベースのサイジングガイドラインおよびハードウェア要件については、「データベース環境の準備」トピックの「OOデータベースのサイジング」および「ハードウェア要件」を参照してください。

MySQLのハードウェア要件については、お使いのMySQLリリースおよびオペレーティングシステムのインストールのセクションを参照してください。

ソフトウェア要件

次の表に、OOでサポートされているMySQLリリースを示します。

MySQLデータベースのリリース		
バージョン	**種類**	**32/64ビット**
5.5	Community	x86 32ビット
		x86 64ビット
	Standard	x86 32ビット
		x86 64ビット
	Enterprise	x86 32ビット
		x86 64ビット
5.6	Community	x86 32ビット
		x86 64ビット
	Standard	x86 32ビット
		x86 64ビット
	Enterprise	x86 32ビット
		x86 64ビット
5.7	Community	x86 32ビット

MySQLデータベースのリリース		
バージョン	**種類**	**32/64ビット**
		x86 64ビット
	Standard	x86 32ビット
		x86 64ビット
	Enterprise	x86 32ビット
		x86 64ビット

> **注:**
> MySQL 5.6.20および5.6.21の場合、**innodb_log_file_size**の要件が大幅に増加します。
>
> MySQL 5.6.1 - 19の場合、推奨サイズは256Mですが、MySQL 5.6.20 - 21の場合は2GBです。

サポートされるプラットフォームについては、MySQLのドキュメントを参照してください。

MySQLコネクター

MySQLコネクターのjarファイルは、OOインストールでは提供されません。ダウンロードし、各Central **<インストールフォルダー>/central/lib**フォルダーに保存してください。

使用しているコネクターのバージョンがご使用のデータベースサーバーバージョンと完全に互換性があることを確認してください。現時点では、サポートされているバージョンは5.1.35のみです。

アップグレードノート: バージョン10.60より前は、MySQL JDBC 5.1.21が使用されていたため、アップグレードプロセス中は、5.1.35をダウンロードして使用するようにしてください。

テスト済みデプロイメントの例

次の表に、品質保証スタッフによる詳細なテスト済みのデプロイメント環境を示します。

データベースリリース			オペレーティングシステム
バージョン	**32/64ビット**	**パッチ**	
MySQL Server 5.6.19 Community Edition	64ビット		Windows 2012 Standard Edition (64ビット)
MySQL Server 5.6.12 Community Edition	64ビット		Red Hat Enterprise Linux 6.3 (64ビット)

言語サポート

MySQL Serverの文字セットは`utf8`に設定する必要があります。これにより、任意のUnicode文字 (すべての言語のほぼすべての一般文字) を使用できます。OOデータベースでは`utf8_bin`照合順序を使用することに注意してください。

関連トピック

データベース環境の準備 (23ページ)

MySQLの構成

このセクションでは、MySQLおよびデータベース構成の設定について説明します。

凡例:

- **必須**の構成オプション/値は**太字/オレンジ色**のフォントで表記します。
- **推奨**の構成オプション/値は**太字/紫色**のフォントで表記します。
- サポート構成オプション/値は標準フォントで表記し、コンマ区切りのリストで示す場合があります。
- コメントはイタリック体フォントで表記します。

MySQL 5.5、5.6、5.7			
インスタンス/サーバーオプション			
サーバー構成オプション	デフォルト、別途指示がある場合を除く		
[mysqld]	**必須**	**推奨**	**サポート**
character-set-server	**utf8**		
collation-server	**utf8_bin**		
transaction-isolation	**READ-COMMITTED**		
max-allowed-packet	**250M**		
max-connections	**>=1000**		
default-storage-engine	**INNODB**		
innodb_log_file_size (MySQL 5.6.1 - 19の場合)	**256M**		
innodb_log_file_size (MySQL 5.6.20 - 21の場合)	**2GB**		
innodb_log_file_size (MySQL 5.6.22 - 5.7の場合)	**256M**		
explicit-defaults-for-timestamp	**0**		
sql-mode	下記を参照[1]		
explicit_defaults_for_timestamp	**0 (オフ/非アクティブ)** **一部のリリースでは非推奨**		
innodb_file_per_table		**1**	
innodb_thread_concurrency		**0**	
table_open_cache		**1000**	

[1]**sql-mode**は、以下の値に設定したり、以下の値を含めたり**しないでください**。

- NO_ZERO_DATE
- NO_ZERO_IN_DATE
- "TRADITIONAL"

有効な**sql-mode**の例:

- "ANSI"
- ERROR_FOR_DIVISION_BY_ZERO,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION

MySQL 5.5、5.6、5.7			
インスタンス/サーバーオプション			
sort_buffer_size		**2M**	
read_buffer_size		**2M**	
tmp_table_size		**400M**	
max_heap_table_size		**400M**	
innodb_buffer_pool_size		**4096M**	
innodb_additional_mem_pool_size		**20M**	
binlog_format		**row**	
innodb_flush_log_at_trx_commit		**2**	
innodb_flush_method		**O_DIRECT** (Linuxのみ)	
innodb_doublewrite		**0**	
MySQL 5.5、5.6、5.7			
その他のオプション			
サーバー構成オプション	デフォルト、別途指示がある場合を除く		
	必須	**推奨**	**サポート**
[client]			
default-character-set	**utf8**		
[mysql]			
default-character-set	**utf8**		
[mysqldump]			
max_allowed_packet	**250M**		

注:
- MySQLオプションは、さまざまなコンテキストで下線またはハイフンを使用して記述されることがあります。サーバーのバージョンや使用コンテキストに合わせて正しいフォームを使用してください。

MySQLでのOOデータベースの手動作成

OOのセットアップ時には、OOインストーラーで新規データベースを自動作成するか、既存のデータベースを使用することができます。

インストール中に、DBAの役割を使用してデータベースサーバーに接続する(“root”として接続する)権限が与えられている場合は、[create the database/schema]オプションを使用してください。その場合はこのセクションはスキップできます。

このセクションでは、MySQL上でOOデータベースを手動作成する手順について説明します。

> **注:**
> この時点ではデータベースとユーザーのみが作成され、テーブルやインデックスなどのオブジェクトは作成されません。これらのオブジェクトはOOが最初に起動されたときに作成されます。

このセクションの操作が必要なのは、たとえば、セキュリティ上の制限により、OOのインストール時にシステム権限を持つログイン資格情報を使用しない場合などです。このような場合、ユーザー(または組織のDBA)はデータベースを作成した後に、基本的な権限を使用してOOインストーラーで既存のデータベースに接続する必要があります。

データベースを作成するには、**CREATE** (またはそれ以上)のアクセス許可を持つユーザーを使用してSQL Serverインスタンスに接続する必要があります。

> **注:**
> - **root**にはすべての権限があります。DBAの役割のメンバーは、ユーザーやデータベースの作成を行うこともできます。
> - 次の手順は、熟練したMySQLデータベース管理者のみが行うようにしてください。
> - MySQL WorkbenchのGUIを使用する場合は、以下に示すSQLコードに対応したすべてのオプションを選択してください。
> - すべてのデータベース作成オプションを指定するのではなく、デフォルト値と異なるオプションのみを指定します。よくわからない場合は、デフォルト値を使用します。

データベースを作成するには、次の手順を実行します。

1. “root”またはDBA役割のその他のメンバーとしてMySQLにログインします。
2. **<インストールフォルダー>/central/bin/sql/MySQL/mysql_create_oo_db.sql**を編集して実行し、データベースとユーザーが正常に作成されたことを確認します。スクリプトのパラメーターは、それぞれの環境に合わせて変更します。

```
SET @OODB='OODB';
SET @OOUSER='OOUSER';
SET @OOPASS='OOPa55WorD';

SET @SQL1 = CONCAT('CREATE DATABASE IF NOT EXISTS `',@OODB,'` COLLATE utf8_
bin');
SET @SQL2 = CONCAT('CREATE USER ''',@OOUSER,'''@''%'' IDENTIFIED BY
''',@OOPASS,'''');
SET @SQL3 = CONCAT('GRANT ALL PRIVILEGES ON `',@OODB,'`.* TO
''',@OOUSER,'''');
PREPARE stmt1 FROM @SQL1;
PREPARE stmt2 FROM @SQL2;
PREPARE stmt3 FROM @SQL3;
EXECUTE stmt1;
EXECUTE stmt2;
EXECUTE stmt3;
DEALLOCATE PREPARE stmt1;
DEALLOCATE PREPARE stmt2;
DEALLOCATE PREPARE stmt3;
FLUSH PRIVILEGES;
```

3. 新しく作成したデータベースとの接続をテストし、正常にログインできることを確認します。
4. (オプション) 新しいログインとユーザーでデータベースオブジェクトを作成できることを確認するには、OOで新しく作成されたログイン役割資格情報を使用してデータベースサーバーに接続し、以下を実行します。

   <インストールフォルダー>/central/bin/sql/MySQL/mysql_create_test_table.sql

   スクリプトが正しく実行され、エラーが表示されていないことを確認します。

```
USE OO;

CREATE TABLE TEST_TABLE(
       TEST_COLUMN int NULL
);

INSERT INTO TEST_TABLE (TEST_COLUMN) VALUES ( 1 );
INSERT INTO TEST_TABLE (TEST_COLUMN) VALUES ( 2 );
```

データベースオブジェクトの手動作成

データベースおよびユーザーの設定が済んだ後に、OOサービスが初めて起動されてデータベースに接続したときに、データベースオブジェクト (テーブル、インデックスなど) が作成されます。

OO用に作成されたユーザーがデータベースオブジェクトを作成または変更するアクセス許可を持っていない場合、データベースオブジェクトを手動で作成できます (DMLオペレーションのみに制限されます)。

手動による操作を行わずにOOでアップグレードオペレーションを実行できるようにするために、OOデータベースユーザーにDDL関連の権限を付与することをお勧めします。これは、これらのオペレーションでデータベース構造の変更が必要になる場合があるためです。

データベースオブジェクトを手動で作成するには、次の手順を実行します。

1. **docs\sql**の下にあるOOインストールzipファイルから**mysql.sql**ファイルを展開します。
2. OOデータベースのデータベースオブジェクトを作成および変更するアクセス許可を持っている適切なユーザーとしてOOデータベースに接続します。
3. **mysql.sql**ファイルを実行し、エラーが発生しないことを確認します。

MySQLデータベースのメンテナンス

このセクションでは、データベースのバックアップ、データベース整合性のチェック、インデックス断片化の処理、およびデータベースの監視など、MySQL上で作成したOOデータベースに関する推奨メンテナンスタスクについて説明します。

OOデータベースのバックアップ

MySQLデータベースは、**mysqldump**や**mysqlbackup**などの、ツールを使用してバックアップできます。OOデータベースは、データベース全体がバックアップされる限り、どのような種類の方法/ツールを使用してバックアップしても構いません。

OOのバックアップ計画を作成する際には、次のガイドラインを考慮してください。

- **バックアップ方法:** バックアップ方法は、失われる可能性のある情報量やシステム復旧に使用できる時間など、ビジネス上の考慮事項に大きく依存します。特定の時点への復旧が必要な場合や、数時間分のデータ損失しか許されない場合は、完全復旧モデルを使用して、完全バックアップや差分バックアップを毎日実行し、ビジネス要件に応じてトランザクションログバックアップをN時間ごとに実行します。
- **バックアップ頻度:** 毎日バックアップを行うことをお勧めします (特に、OOを毎日使用/変更する場合)。最低限、月に1回はバックアップします。
- **タイミング:** OOが最もアクティブでない時間にバックアップをスケジュールします。
- **保有期間:** 保有期間はビジネスガイドラインや法規によって異なります。

メンテナンス計画の作成

OOデータベースのメンテナンスでは、インデックスの再構築や空き領域の回収などを行います。このセクションで説明するスクリプトとツールを使用して、OOデータベースの状態を正常に維持します。

データベースメンテナンス用の推奨ユーティリティ

OOデータベースの状態を正常に維持するためには、システムのメンテナンスウィンドウ中に**mysqlcheck**ユーティリティを実行するようにスケジュールすることをお勧めします。

> **重要:** この操作ではテーブルがロックされます。必ずOOシステムが稼働していないメンテナンスウィンドウ中に実行してください。

以下は、このユーティリティの実行例です。

```
mysqlcheck -uoouser -p????? -os --auto-repair OO
```

"oouser" と"OO" はそれぞれ、OOの実際のユーザー名とデータベース名に置き換えてください。

> **重要:** パスワードは明示しないようにしてください。データベースパスワードのセキュリティ保護に関する推奨事項ついては、MySQLのドキュメントを参照してください。

アップグレード/ロールバック手順

OOをアップグレードまたはロールバックする前に、次の手順を実行します。

1. 不要なデータをデータベースから削除します。

   アップグレードやロールバックオペレーションを実行すると、フロー実行データが変換されます。データが少なければ、それだけアップグレードやロールバックにかかる時間が短くなります。すべてのフロー実行データが必要なデータである場合は、最初にデータベース全体をバックアップしてから、削除します。
2. できればデータベースを削除 (不要なデータを削除) した後に、データベースをバックアップしてください。
3. このオペレーションを、データベースサーバーの非表示期間にスケジュールします。
4. 一時スペースやトランザクションログ、類似リソースなどの共有リソースが使用されていないことを確認します。
5. MySQLコネクターのjarファイルのバックアップコピーを作成します。アップグレードシナリオでは、最新のjarファイルを使用するようにしてください。

OOをアップグレードまたはロールバックした後で、次の手順を実行します。

1. メンテナンスおよび削除ストアドプロシージャーの該当するバージョンをインストールします。

   OOアップグレードユーティリティでは、ストアドプロシージャーと削除フローはアップグレードまたはロールバックされません。
2. データベースのメンテナンスと削除ルーチンを再スケジュールします。

Postgresデータベースのデプロイとメンテナンス

Postgresを使用してOOをデプロイするには、既存のPostgresデータベースサービスが必要です。このセクションでは、データベースサービスの新規作成については扱いません。データベースサービスの新規作成が必要な場合は、Postgresが提供しているドキュメントを参照してください。このセクションには、Postgresの構成に関する推奨事項が記載されています。

このセクションのトピックは、次のとおりです。

Postgresデプロイメントのワークフロー

Postgresを使用してOOをデプロイするには、次の手順を実行します。

1. **サイジングガイドラインを確認します。**
2. **ハードウェア要件とソフトウェア要件を確認します。**
3. **Postgresを構成します。**
4. **PostgresにOO データベースを作成します。**

関連トピック

データベース環境の準備 (23ページ)

Postgresの構成 (59ページ)

PostgresでのOOデータベースの手動作成 (60ページ)

Postgresのシステム要件

このセクションでは、OOでPostgresを使用する場合のシステム要件について説明します。

ハードウェア要件

OOデータベースのサイジングガイドラインおよびハードウェア要件については、「データベース環境の準備」トピックの「OOデータベースのサイジング」および「ハードウェア要件」を参照してください。

Postgresのハードウェア要件については、お使いのPostgresリリースおよびオペレーティングシステムのインストールのセクションを参照してください。

ソフトウェア要件

次の表に、OOでサポートされているPostgresリリースを示します。

Postgresデータベースのリリース	
バージョン	**種類**
9.3	x86 32ビット
	x86 64ビット
9.4	x86 32ビット
	x86 64ビット
9.5	x86 32ビット
	x86 64ビット
9.6	x86 32ビット
	x86 64ビット

注: サポートされているバージョンのみを使用してください。

サポートされるプラットフォームについては、Postgresのドキュメントを参照してください。

テスト済みデプロイメントの例

次の表に、品質保証スタッフによる詳細なテスト済みのデプロイメント環境を示します。

データベースリリース		オペレーティングシステム
バージョン	**32/64ビット**	
Postgres 9.3.2	64ビット	Red Hat Enterprise Linux 6.3 (64ビット)
Postgres 9.4.6	64ビット	Red Hat Enterprise Linux 7 (64ビット)

データベースリリース		オペレーティングシステム
バージョン	32/64ビット	
Postgres 9.5.1	64ビット	Red Hat Enterprise Linux 7 (64ビット)
Postgres 9.6.3	64ビット	Windows 2012 R2

言語サポート

Postgresでは、データベースレベルで文字セットと照合順序が特定されます。OOデータベースは、Unicode (utf8) エンコードと照合順序を使用します。これにより、任意のUnicode文字 (すべての言語のほぼすべての一般文字) を使用できます。

関連トピック

データベース環境の準備 (23ページ)

Postgresの構成

このセクションでは、Postgresおよびデータベース構成の設定について説明します。

凡例:

- **必須**の構成オプション/値は**太字/オレンジ色**のフォントで表記します。
- **推奨**の構成オプション/値は**太字/紫色**のフォントで表記します。
- サポート構成オプション/値は標準フォントで表記し、コンマ区切りのリストで示す場合があります。
- コメントはイタリック体フォントで表記します。

Postgres 9.3～9.6			
インスタンス/サーバーオプション			
インスタンス構成オプション	デフォルト、別途指示がある場合を除く		
	必須	**推奨**	**サポート**
max_connections	>=1000		
default_transaction_isolation	'read committed'		
autovacuum	on		
track_counts	on		
shared_buffers	>=512MB [1]		
effective_cache_size	>=2048MB [1]		
work_mem	>=1MB [1]		
maintenance_work_mem	>=32MB [1]		

Postgres 9.3～9.6			
インスタンス/サーバーオプション			
lc_messages		**'en_US.UTF-8'**	任意
lc_monetary		**'en_US.UTF-8'**	任意

注: [1] - 最小値。これらの値の環境に合わせた調整方法については、Postgresのドキュメントを参照してください。

PostgresでのOOデータベースの手動作成

OOのセットアップ時には、OOインストーラーで新規データベースを自動作成するか、既存のデータベースを使用することができます。

インストール中に、特権を持つユーザーとしてデータベースサーバーに接続する(“postgres” として接続する)権限が与えられている場合は、[create the database/schema] オプションを使用してください。その場合はこのセクションはスキップできます。

このセクションでは、Postgres上でOOデータベースを手動作成する手順について説明します。

注:
この時点ではデータベースと役割のみが作成され、テーブルやインデックスなどのオブジェクトは作成されません。これらのオブジェクトはOOが最初に起動されたときに作成されます。

このセクションの操作が必要なのは、たとえば、セキュリティ上の制限により、OOのインストール時にシステム特権を持つログイン/ユーザー資格情報を使用しない場合などです。このような場合は、ユーザー(または組織のDBA)が最初にデータベース、ログイン、ユーザーを作成した後に、「基本的な権限」を使用してOOインストーラーで既存のデータベースに接続する必要があります。

データベースを作成するには、少なくとも**CREATEUSER**と**CREATEDB**の権限を持つログインを使用してPostgresインスタンスに接続する必要があります。

注:
- 標準で作成される**postgres**ユーザーには、必要なすべての権限があります。
- 次の手順は、熟練したPostgresデータベース管理者のみが行うようにしてください。
- PgAdminのGUIを使用する場合は、以下に示すSQLコードに対応したすべてのオプションを選択してください。
- すべてのデータベース作成オプションを指定するのではなく、デフォルト値と異なるオプションのみを指定します。よくわからない場合は、デフォルト値を使用します。

データベースを作成するには、次の手順を実行します。

1. “postgres” または**CREATEUSER**および**CREATEDB**の権限を持つその他のログインの役割としてPostgresにログインします。
2. 次を編集し、実行します。

   <installation folder>/central/bin/sql/PostgreSQL/postgres_create_oo_db_linux.sql

   または、

   postgres_create_oo_db_windows.sql

データベース、ログイン、ユーザーが正常に作成されたことを確認します。スクリプトのパラメーターは、それぞれの環境に合わせて変更します。

```
CREATE ROLE "oouser" LOGIN
UNENCRYPTED PASSWORD '???????'
NOSUPERUSER INHERIT NOCREATEDB NOCREATEROLE NOREPLICATION;

CREATE DATABASE "OO"
WITH OWNER = "oouser"
ENCODING = 'UTF8'
TABLESPACE =
LC_COLLATE =
LC_CTYPE =
CONNECTION LIMIT = 1000;
```

3. (オプション) 新しいログインとユーザーでデータベースオブジェクトを作成できることを確認するには、OOで新しく作成されたログイン役割資格情報を使用してデータベースサーバーに接続し、以下を実行します。

   <installation folder>/central/bin/sql/PostgreSQL/postgres_create_test_table.sql

   スクリプトが正しく実行され、エラーが表示されていないことを確認します。

```
CREATE TABLE TEST_TABLE(
       TEST_COLUMN int NULL
);

INSERT INTO TEST_TABLE (TEST_COLUMN) VALUES ( 1 );
INSERT INTO TEST_TABLE (TEST_COLUMN) VALUES ( 2 );
```

データベースオブジェクトの手動作成

データベースおよび役割の設定が済んだ後に、OOサービスが初めて起動されてデータベースに接続したときに、データベースオブジェクト (テーブル、インデックスなど) が作成されます。

データベースオブジェクトを (OOサービスの代わりに) 手動で作成するには、次の手順を実行します。

1. **docs\sql**の下にあるOOインストールzipファイルから**postgres.sql**ファイルを展開します。
2. OOデータベースユーザーとしてOOデータベースに接続します。
3. **postgres.sql**ファイルを実行し、エラーが発生しないことを確認します。

Postgresデータベースのメンテナンス

このセクションでは、データベースのバックアップ、データベース整合性のチェック、インデックス断片化の処理、およびデータベースの監視など、Postgres上で作成したOOデータベースに関する推奨メンテナンスタスクについて説明します。

OOデータベースのバックアップ

Postgresデータベースは、**pg_dump**や**pg_backup**などのツールを使用してバックアップできます。OOデータベースは、データベース全体がバックアップされる限り、どのような種類の方法/ツールを使用してバックアップしても構いません。

OOのバックアップ計画を作成する際には、次のガイドラインを考慮してください。

- **バックアップ方法:** バックアップ方法は、失われる可能性のある情報量やシステム復旧に使用できる時間など、ビジネス上の考慮事項に大きく依存します。特定の時点への復旧が必要な場合や、数時間分のデータ損失しか許されない場合は、完全復旧モデルを使用して、完全バックアップや差分バックアップを毎日実行し、ビジネス要件に応じてトランザクションログバックアップをN時間ごとに実行します。データ損失に対する許容度が大きい場合は、単純復旧モデルを使用して、完全バックアップを毎日または毎週実行します。
- **バックアップ頻度:** 毎日バックアップを行うことをお勧めします(特に、OOを毎日使用/変更する場合)。最低限、月に1回はバックアップします。
- **タイミング:** OOが最もアクティブでない時間にバックアップをスケジュールします。
- **保有期間:** 保有期間はビジネスガイドラインや法規によって異なります。

メンテナンス計画の作成

OOのPostgresデータベースでは**autovacuum**を有効にする必要があるため、メンテナンスでは主にテーブルのREINDEXを行います。以下の例を使用して、OOデータベースの状態を正常に維持します。

データベースメンテナンス用の推奨ユーティリティ

OOデータベースの状態を正常に維持するためには、システムのメンテナンスウィンドウ中にREINDEXを実行することをお勧めします。

> **重要:** この操作ではテーブルがロックされます。必ずOOシステムが稼働していないメンテナンスウィンドウ中に実行してください。

次に、**reindexdb**ユーティリティを使用してデータベース全体のREINDEXを行う例を示します。

```
reindexdb -d OO -U oouser -W ?????
```

"OO" と"oouser" を実際のOOのデータベースとユーザー名に置き換え、"?????" をOO DBユーザーのパスワードに置き換えます。

> **重要:** パスワードは明示しないようにしてください。データベースパスワードのセキュリティ保護に関する推奨事項ついては、Postgresのドキュメントを参照してください。

履歴データ削除用のユーティリティ

OO 10.xでは、デフォルトですべてのフローおよびステップの実行データがデータベース内に保持されるので、前のフロー実行の包括的なデバッグを行うことができます。結果として、システムのスループットおよびフローの複雑さに応じてデータベースのサイズが増加します。データベースサイズを追跡し、古い無関係な情報が定期的に削除されていることを確認します。

Posgtresデータベースでの空き領域の回収には、次の2つのフェーズが必要です。

1. DELETEコマンドの後に、領域が削除済みとしてマークされます。
2. 空き領域を再利用できるようにするためにバックグラウンドで “vacuum” プロセスが実行されます。

> **注:** 下で説明する削除プロシージャーの実行の後に、vacuumプロセスが動作を開始し、テーブルとインデックスの “vacuum” 処理が開始されます。

> **重要:** 別の削除を開始する前に、vacuumプロセスを正常に完了させる必要があります。削除とvacuumが重なった場合、それらがオブジェクトのロックを競合するために相互に停止させます。

“Vacuum Full” オペレーション

PostgreSQLデータベースは、**Vacuum Full**オペレーションが実行されたときにラージオブジェクト (LOB) 領域のみを回収できます。OOの膨大なデータベース領域がLOBとして保存されます。したがって、OOが最もアクティブでない時間に何度か、定期的な**Vacuum Full**オペレーションをスケジュールする必要があります。**Vacuum Full**オペレーションは排他テーブルロックを必要とするため、OOのアクティビティを妨げる場合があります。

OO履歴削除用ストアドプロシージャーをインストールして使用するには、次の手順を実行します。

1. ITOM Marketplaceから最新の**HPE_OO_DB_Maintenance.zip**パックをダウンロードし、一時フォルダーに解凍します。『OO管理ガイド』の「データベース環境のセットアップ」セクションの指示に従い、PostgreSQLの削除ストアドプロシージャーをインストールして実行します。

```
SELECT OOPurgeHistory(90,10000,0,1,4,0);
```

アップグレード/ロールバック手順

OOをアップグレードまたはロールバックする前に、次の手順を実行します。

1. 不要なデータをデータベースから削除します。

   アップグレードやロールバックオペレーションを実行すると、フロー実行データが変換されます。データが少なければ、それだけアップグレードやロールバックにかかる時間が短くなります。すべてのフロー実行データが必要なデータである場合は、最初にデータベース全体をバックアップしてから、削除します。
2. "vacuum full" オペレーションをサポートするのに十分なディスク容量があることを確認してから、"vacuum full" オペレーションを開始します。
3. できればデータベースを削除 (不要なデータを削除) した後に、データベースをバックアップしてください。
4. このオペレーションを、データベースサーバーの非表示期間にスケジュールします。
5. トランザクションログファイルに対して十分なディスク空き領域があることを確認します。

OOをアップグレードまたはロールバックした後で、次の手順を実行します。

1. 削除するストアドプロシージャーの該当するバージョンをインストールします。

   OOアップグレードユーティリティでは、ストアドプロシージャーと削除フローはアップグレードまたはロールバックされません。
2. データベースの削除ルーチン (ストアドプロシージャーや削除フロー) を再スケジュールします。
3. 定期的なデータベース "vacuum full" オペレーションを再スケジュールします。

関連トピック

データベース環境のセットアップ (23ページ)

データベースの削除

削除フロー

削除フローはMicro Focusソリューションコンテンツパックのコンテンツとして提供されています。OOデータベースを削除する際には、削除フローを使用することをお勧めします。削除フローの「スループット」(所定の時間内に十分な量の行を削除できる)が十分である限り、削除ストアドプロシージャーではなく、削除フローを使用することをお勧めします。

削除フローには、ストアドプロシージャーに比べていくつかの利点があります。

- 管理性 - OO管理者は削除オペレーションのスケジュールと追跡を簡単に行うことができます。
- データベース権限 - 削除オペレーションのインストールとスケジュール設定にDBAの支援を必要としません。
- 削除機能のアップグレード - OOをアップグレードし、新しいテーブルでデータの削除が必要になると、この機能は既存のフローに追加されるか、新しいフローが削除のために作成されます。

  ただし、新しいストアドプロシージャーは手動でダウンロードしてインストールする必要があります。

> **注:**
> 各削除タイプ(再実行、ステップログなど)に対して、一度に1つの削除フローのみを実行できます。

削除のベストプラクティス

削除オペレーションに関しては、次の2つの異なる基準を明確にすることが重要です。

- 保持期間 - 保持するデータの保存期間
- 削除頻度 - 削除オペレーションが実行される頻度

これらの2つの基準は完全に別のものです。1年分のデータを保持する一方で、削除の頻度を30分にすることができます。

保持期間は、データベースサイズとビジネス要件のトレードオフとして決定されることが多いため、削除頻度は、時間単位にも、日単位または週単位にもなる可能性があります。推奨される削除頻度は、「時間単位」と「日単位」の間です。

少量のデータを頻繁に削除するほうが、毎週または毎月のメンテナンス期間中に大量のデータを削除するよりも、より良い方法であることが分かっています。

削除頻度を最適化するための推奨ワークフローを以下に示します。

- 4時間ごとに定期的な削除オペレーションをスケジュールし、以下の観点で観察します。
  - 削除オペレーションが完了するまでにかかる時間 (平均および最大期間)
  - データベースのパフォーマンスへの影響
  - Operations Orchestrationのパフォーマンス(ユーザーインタフェースやフローの実行)への影響
- 削除オペレーションが30分以内に完了し、DBAまたはデータベースに大きな効果が表れない限り、この頻度を維持します。

  1時間単位から1日単位までの頻度を試して、最も効率の良い頻度を確認してもよいでしょう。

  データベースとOOにある程度影響があるのは普通です。
- このような状態ではない場合、OOの稼働状態が一日の間に大きく変化し、一定の「静止期間」が

存在するのであれば、この期間に削除オペレーションを実行するようにスケジュールします。

Microsoft SQL Serverに関する追加のガイドライン

この付録では、Microsoft SQL Server上でのDBAのデプロイメントに関連する追加のガイドラインを示します。

Windows認証を使用したMicrosoft SQL Serverデータベースへのアクセス

特別な構成を行わない限り、DBAではMicrosoft SQL Server認証を使用してMicrosoft SQL Serverデータベースにアクセスします。OOインストーラーは、OOインストール時のWindows認証の使用をサポートしていませんが、OOのインストール後にはWindows認証を使用できます。

この付録では、OOでWindows認証を使用してMicrosoft SQL Serverデータベースにアクセスできるようにする手順について説明します。

Windows認証で稼働するOOの構成

OOでは、Microsoft SQL Server認証の代わりにWindows認証を使用してOOデータベースにアクセスできるように構成できます。

OOでWindows認証を使用してMicrosoft SQLデータベースにアクセスできるようにするには、次の手順を実行します。

1. 次のように、**<OOインストール>/central/bin**にある**encrypt-password.bat**ユーティリティを使用して、Windowsのユーザーパスワードを暗号化します。

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

   次の手順で使用するため、生成された文字列を保存します。
2. 既存の(使用可能な)データベース接続がある場合は、**<OOインストール>/central/conf**の下にある現在の**database.properties**ファイルをバックアップします。
3. **<OOインストール>/central/conf**の下にある**database.properties**ファイルを編集し、次の例に合わせて関連するパラメーターの構文を変更します。

   > **注:**
   > スクリプトをコピーする際は、コピーしたバージョンから余分な改行を削除する必要がある場合があるので注意してください。

   SQL Server 2008R2および2012の場合:

   ```
   db.username=[ユーザー名]
   db.password=[encrypt-password.batで生成された文字列]
   ```

   ```
   jdbc.url=jdbc\:jtds\:sqlserver\://[DBホスト名]\:[ポート]/[DB名]
   ;sendStringParametersAsUnicode\=true
   domain\=[ドメイン名]
   ```

SQL Server 2014および2016の場合:

```
db.username=[ユーザー名]

db.password=[encrypt-password.batで生成された文字列]
```

```
jdbc.url=jdbc\:sqlserver\://[DBホスト名]\:[ポート];databaseName\=[DB名];sendStringParametersAsUnicode\=true;integratedSecurity\=true;
```

SQL Server 2014および2016の場合は、Centralサーバーマシンのシステムパス(`<Centralインストール>\java\bin`)に**sqljdbc_auth.dll**をコピーする必要があるかもしれません。詳細については、Microsoft JDBCのドキュメントを参照してください。

> **注:** `sqljdbc_auth.dll`は`<Centralインストール>\java\bin`にあります。

強調表示した項目の値は、それぞれの環境に合わせて変更します。

jdbc.urlパラメーターは、読みやすくするために複数の行に分けて記述されています。この例を**database.properties**ファイルにコピーする際は、すべての改行を削除して、**jdbc.url**パラメーターを空白のない単一の行で構成してください。

常時オンで稼働するOOの構成

SQL Server 2008R2および2012の場合

OOは、jtdsコネクターを使用してSQL Server 2008R2および2012データベースに接続します。したがって、可用性グループリスナー用に1つのIPアドレスしか登録できません。また、OOは、マルチサブネットフェールオーバーをサポートしていません。

クラスター構成

クラスター構成環境は次のように構成します。

1. 可用性グループリスナーに関連付けられたネットワーク名に対して、クラスターオプション**RegisterAllProvidersIP**を0に設定します。
2. オプションで、データベース再接続時間を減らすために、同じクラスターリソースの**HostRecordTTL**の値を小さくします。

構成オプションと構成例の完全な説明については、Microsoft SQL Serverマニュアル(「Create or Configure an Availability Group Listener」)を参照してください。

可用性グループリスナーを使用したOperations Orchestrationのインストール

OOをインストールする前に、可用性グループリスナーのネットワーク名がすべてのCentralマシンからアクセス可能なIPアドレスを解決することを確認します。

1. OOのインストールウィザードの[**データベース接続構成**]ページで、[**ホスト名またはIPアドレス**]フィールドに可用性グループリスナーのネットワーク名を入力します。
2. その他の詳細を入力して、[**接続テスト**]ボタンをクリックします。

可用性グループリスナーのネットワーク名を指定すると、障害が発生した場合にOOのCentral Serverがデータベースに再接続できます。

サイレントインストールの場合は、以下を使用します。

```
db.url=jdbc:jtds:sqlserver://[AGネットワーク名]:[ポート]/[DB
名];sendStringParametersAsUnicode=true
```

SQL Server 2014および2016の場合

OOは、Microsoft JDBC 4.2コネクターを使用してSQL Server2014および2016データベースに接続します。

常時オンクラスターに接続するためにJDBC URLをフォーマットする方法の例を以下に示します。

```
db.url=jdbc:sqlserver://[AG-NET-NAME];instanceName=[NAMED-INST-NAME];
databaseName=[DB-NAME];multiSubnetFailover=true;applicationIntent=ReadWrite;
sendStringParametersAsUnicode=true
```

> **注:** **multiSubnetFailover**と**applicationIntent**は、より複雑な動作を可能にするために追加されています。

> **注:** SQL Server 2014および2016常時オンクラスターと連動したマルチサブネットフェールオーバーのシナリオは、OOでは認められていませんでした。

追加情報と接続オプションについては、Microsoftのドキュメント『高可用性、障害回復のためのJDBC Driverのサポート』と、Microsoft JDBCのドキュメント『接続プロパティの設定』を参照してください。

Oracleに関する追加のガイドライン

この付録では、Oracle 11gR2および12cR1 Real Application Cluster (RAC) 環境でOOを使用するのに必要な構成について説明します。これは上級ユーザー向けです。

Oracle RAC

クラスターは相互に接続されたサーバーの集合で、エンドユーザーやアプリケーションからは1つのサーバーとして認識されます。Oracle Real Application Cluster (RAC) は、高可用性、スケーラビリティ、フォールトトレランスを実現するOracleのソリューションです。Oracle RACでは、クラスター化されたサーバーで同じストレージを共有します。

Oracle RACは、複数のハードウェアサーバーのクラスターにインストールされた単一のOracleデータベースです。各サーバーでデータベースのインスタンスを1つずつ実行し、すべてのインスタンスで同じデータベースファイルを共有します。

Oracle RACの詳細については、お使いのリリースのOracleドキュメントセット内の『Oracle Clusterwareガイド』および『Oracle Real Application Clusters管理およびデプロイメントガイド』を参照してください。

この付録では、次のOracle RACを使用します。

- Oracle RACクラスター名 : OORAC
- サービス名 : ORCL.MY.DOMAIN
- マシン名 : Server1、Server2
- 各マシンには、次のようにOORACのOracleインスタンスが1つずつ存在します。
  - Server1上のSID: OORAC1
  - Server2上のSID: OORAC2
- 各マシンには、次のように仮想IPが1つずつ存在します (Server1-VipとServer2-Vip)。

  - Server1-VipはServer1に割り当てられています
  - Server2-VipはServer2に割り当てられています

  仮想IPはマシンに割り当て済みの静的IPに追加で割り当てられます。
- SCANリスナーは、DNS/GNSを使用して通常公開される以下の仮想IPを使用します。

  SCAN-Vip
- 両方のサーバー上のローカルリスナーはデフォルトポート1521でリッスンし、データベースサービスOORACをサポートします。SCANリスナーはいずれかのクラスターノード上に配置され、障害が発生した場合はその仮想IPアドレスに従ってフェールオーバーを実施します。

> **注:**
> 12c SCANでは複数のSCANリスナーを(サブネットごとに)構成できますが、OOでは単一の12c SCANリスナーへの接続のみがサポートされています。

Single Client Access Name (SCAN)

Oracleはリリース11gで、RACに接続するクライアント用の優先アクセス方式として、Single Client Access Name (SCAN)を導入しました。この方式では、クライアントはRAC内の個別ノードを構成する必要がなく、SCANまたはSCAN VIPと呼ばれる単一の仮想IPを使用します。

SCANは、組織のドメインネームサーバー(DNS)、またはクラスター内の複数のリスナーを反映して複数のIPアドレスを循環するグリッドネーミングサービス(GNS)のいずれかでクラスター用に定義された単一のネットワーク名です。SCANを使用することで、クラスターでノードを追加または削除した場合にクライアントを変更する必要がなくなります。

SCANとSCANに関連付けられたIPアドレスにより、クラスターを構成するノードに関係なく、クライアントは接続にいつでも同じ名前を使用できます。11gのSCANアドレス、仮想IPアドレス、パブリックIPアドレスはすべて同じサブネット上に存在している必要があります。12c SCANでは複数のSCANリスナー(サブネットごとに1つ)を構成できます。OOでは単一の12c SCANリスナーへの接続のみがサポートされています。

Oracle 11g RAC環境でOOを使用する場合は、SCAN方式の使用をお勧めします。

Oracle RACで稼働するOOの構成

SCANリスナーの仮想IPへの接続

1. OOのインストールウィザードの[**データベース接続構成**]ページでは、[**ホスト名またはIPアドレス**]フィールドにSCANリスナーの仮想IPアドレスまたはネットワーク名を入力します。
2. [**SID**]ラジオボタンではなく[**サービス名**]ラジオボタンを選択し、Oracle RACサービス名を入力します。
3. その他の詳細を入力して、[**接続テスト**]ボタンをクリックします。

SCANリスナーの仮想IPアドレスを指定すると、障害が発生した場合にOOのCentralサーバーがデータベースクラスターに再接続できます。

この接続方式は、OOのインストールウィザードでデータベースを作成し、そのデータベースを組み込むことができるため、推奨される方式です。

サイレントインストールの場合は、以下を使用します(強調表示されたテキストには実際の値を指定してください)。

```
db.url=jdbc:oracle:thin:@//[SCAN-Vip]:[ポート]/[ORCL.MY.DOMAIN]
```

明示的な接続文字列を使用した負荷分散

Oracle SCANリスナーの仮想IPの使用は推奨する方式ですが、[**Other database**] インストールオプションを使用すると、明示的な接続文字列を指定することもできます。

ここではCentralの**database.properties**ファイルでのロードバランシング接続文字列の例が示されています。

1. 既存の(使用可能な)データベース接続がある場合は、**<OOインストール>/central/conf**の下にある現在のdatabase.propertiesファイルをバックアップします。
2. 次のように**database.properties**ファイルを編集し、強調表示された項目をご使用の環境に合った値に置き換えます。

> **注:**
> スクリプトをコピーする際は、コピーしたバージョンから余分な改行を削除する必要がある場合があるので注意してください。

```
jdbc.url=jdbc\:oracle\:thin\:@
(DESCRIPTION=(LOAD_BALANCE=on)(ADDRESS_LIST=
(ADDRESS=(PROTOCOL=TCP)(HOST=Server1-Vip)(PORT=1521))
(ADDRESS=(PROTOCOL=TCP)(HOST=Server2-Vip)(PORT=1521)))
(CONNECT_DATA=(SERVICE_NAME=ORCL.MY.DOMAIN)))
```

> **注:** 上記の**jdbc.url**は読みやすいように複数行に分けて表示されていますが、実際の構成ファイルでは1行で表示されます。

ロードバランシングが「オン」の場合、デフォルトではリスナー間のフェールオーバーは有効になります。

関連トピック

インストールウィザードの [Other database] オプション (69ページ)

インストールウィザードの [Other database] オプション

このトピックでは、OOのインストールウィザードの [**Other database**] オプションについて説明します。

このオプションでは、特定のJDBCドライバーと接続オプションを使用できます。次の場合にはこのオプションを使用します。

- OOのインストールで提供されるJDBCドライバーとは異なるバージョンを使用する場合 (以下の注を参照してください)。
- 標準のデータベース接続オプションでは現在提供されていないオプションを含めるためにJDBC接続URLを自分で指定する場合。

注:

- 接続は、「データベース環境の準備」セクションの「概要」セクションに記述されているデータベースのタイプとバージョンに制限されます。
- 提供されていないJDBCドライバーの使用については、弊社では責任を負いません。以下のJDBCドライバーがサポートされています。

- jtds-1.3.0.jar – SQL Server 2008R2および2012の場合
- sqljdbc4-4.2.jar – SQL Server 2014および2016の場合
- ojdbc7-12.1.0.2.jar
- postgresql-9.4.1207.jar
- mysql-connector-java-5.1.35 (OOのインストールでは提供されません)

OOをインストールする際に[**Other database**]オプションを選択すると、インストールウィザードの実行時に、データベースや、関連するユーザーまたは役割が作成されません。これらは、インストールウィザードを使用してあらかじめ作成しておく必要があります。データベース、ユーザー、役割の作成方法の詳細については、本ドキュメントの各データベースのトピックの「OOデータベースの手動作成」セクションに記載されています。

Microsoft SQL Server名前付きインスタンスの例

SQL Server 2008R2および2012

以下は、jtds JDBCコネクターを使用したMicrosoft SQL Serverの名前付きインスタンスへの接続例です。データベース、ログインの役割、ユーザーはDBAによって事前に作成されます。ログインの役割はデータベースの所有者 (DMLとDDLのすべての権限がある) になります。

インストールウィザードで以下の詳細を使用して、強調表示された値をご使用の環境に合わせて修正します。

各インスタンスに一意のTCPポートが割り当てられている場合は、[JDBC URLオプション#1] を使用します。

[**データベースブラウザー: service**] がアクティブであり、インスタンス名に基づいて接続を指定できる場合は、[JDBC URLオプション#2] を使用します。このオプションには、TCPポート番号は表示されません。

フィールド	値	コメント
JDBCドライバーjar	C:\my\path\jtds-1.3.0.jar	
JDBCドライバークラス名	net.sourceforge.jtds.jdbc.Driver	
JDBC URL オプション#1	jdbc:jtds:sqlserver://<DB_IP_OR_HOSTNAME>:<DB_ PORT>/<DB_ NAME>;instance=<INSTANCE_ NAME>;sendStringParametersAsUnicode=true	
JDBC URL オプション#2	jdbc:sqlserver://<DB_IP_OR_HOSTNAME>:<INSTANCE_ PORT>;databaseName=<DB_ NAME>;sendStringParametersAsUnicode=true	
ユーザー名	<LOGIN_ROLE>	
パスワード	<LOGIN_ROLE_PASSWORD>	

SQL Server 2014および2016

以下は、Microsoft JDBCコネクターを使用したMicrosoft SQL Serverの名前付きインスタンスへの接続例です。データベース、ログインの役割、ユーザーはDBAによって事前に作成されます。ログインの役割はデータベースの所有者 (DMLとDDLのすべての権限がある) になります。

インストールウィザードで以下の詳細を使用して、強調表示された値をご使用の環境に合わせて修正します。

各インスタンスに一意のTCPポートが割り当てられている場合は、[JDBC URLオプション#1] を使用します。

データベースブラウザーサービスがアクティブであり、インスタンス名に基づいて接続を指定できる場合は、[JDBC URLオプション#2] を使用します。このオプションには、TCPポート番号は表示されません。

フィールド	値	コメント
JDBCドライバーjar	C:\my\path\sqljdbc4-4.2.jar	
JDBCドライバークラス名	com.microsoft.sqlserver.jdbc.SQLServerDriver	
JDBC URL オプション#1	jdbc:sqlserver://<DB_IP_OR_HOSTNAME>:<INSTANCE_PORT>;databaseName=<DB_NAME>;sendStringParametersAsUnicode=true	INSTANCE_PORTは、別のインスタンスを表す
JDBC URL オプション#2	jdbc:sqlserver://<DB_IP_OR_HOSTNAME>;instanceName=<INSTANCE_NAME>;databaseName=<DB_NAME>;sendStringParametersAsUnicode=true	データベースブラウザーサービスがアクティブの場合
ユーザー名	<LOGIN_ROLE>	
パスワード	<LOGIN_ROLE_PASSWORD>	

Microsoft SQL ServerのWindows認証例

SQL Server 2008R2および2012

以下は、Windows認証とjtds JDBCコネクターを使用したMicrosoft SQL Serverへの接続例です。データベースはDBAによって事前に作成されます。Windowsログインアカウントはデータベースの所有者 (DMLとDDLのすべての権限がある) になります。

インストールウィザードで以下の詳細を使用して、強調表示された値をご使用の環境に合わせて修正します。

フィールド	値	コメント
JDBCドライバーjar	C:\my\path\jtds-1.3.0.jar	jtds JDBCドライバーのみを使用
JDBCドライバークラス名	net.sourceforge.jtds.jdbc.Driver	

フィールド	値	コメント
JDBC URL	jdbc:jtds:sqlserver://<DB_IP_OR_HOSTNAME>:<DB_PORT>/<DB_NAME>;domain=<DOMAIN_NAME>;sendStringParametersAsUnicode=true	
ユーザー名	<WINDOWS_USERNAME>	
パスワード	< WINDOWS_USERNAME_PASSWORD>	

> **注:**
> 互換性のない照合順序に関するエラーを無視します。ただし、照合順序を、「データベース環境の準備」セクションの「言語サポート」トピックで説明されているサポートされる照合順序に設定するようにします。

OO Centralサービスを、データベース認証で使用するのと同じユーザーとして実行することをお勧めします(必須ではありません)。認証に対してドメインアカウントを使用することは、ドメイン管理者によって変更される可能性のあるセキュリティポリシーが強制されることがあるので注意してください。

ドメインアカウントのパスワードを変更するときは必ず、新しいパスワードを暗号化し、新しく暗号化したパスワードを**<OOインストール>/central/conf**の下にある**database.properties**ファイルに保存する必要があります。OOクラスターを使用する場合は、すべてのOO Centralサーバーに対して、このオペレーションを繰り返し実行する必要があります。

SQL Server 2014および2016

以下は、Windows認証とMicrosoft JDBCコネクターを使用したMicrosoft SQL Server 2014および2016への接続例です。

前提条件:

1. データベースがDBAによって事前に作成されていること
2. 関連するWindowsアカウントがデータベースの所有者であること(DMLとDDLのすべての権限がある)
3. OOのインストールウィザードがunzipされ、フォルダー形式で抽出されていること
4. **sqljdbc_auth.dll**がダウンロードされ、OOのインストールウィザードの<インストールフォルダー>\java\binにコピーされていること。このdllファイルは、OOのインストールでは提供されていません。MicrosoftのWebサイトからダウンロードできます。Microsoft JDBC 4.2をダウンロードして、64ビットバージョンの**sqljdbc_auth.dll**を抽出してください。

インストールの手順:

1. OOデータベースを所有しているのと同じWindowsアカウントで**<インストールフォルダー>\installer.bat**を実行して、OOのインストーラーを実行します。
2. [**Connectivity**] ページで [**Do not start central server after installation...**] チェックボックスをオンにします。
3. [**Database Connection**] ページで、[**Other database**] オプションを選択して、以下の表に従って詳細を入力します。強調表示されている部分を編集して実際の値で置換します。
4. インストール後に、次のことを実行します。
   a. **sqljdbc_auth.dll**が**<central installation>\java\bin**にあることを確認します。
   b. OOデータベースを所有しているのと同じWindowsアカウントでサービスが実行されるように構成します。
   c. OOのサービスを開始して、データベースに正常に接続できることを確認します。

Centralサーバーのすべてのインストール済み環境で、上記の手順を繰り返します。

インストールウィザードで以下の詳細を使用して、強調表示された値をご使用の環境に合わせて修正します。

フィールド	値	コメント
JDBCドライバーjar	C:\my\path\sqljdbc4-4.2.jar	
JDBCドライバークラス名	com.microsoft.sqlserver.jdbc.SQLServerDriver	
JDBC URL	jdbc:sqlserver://<DB_IP_OR_HOSTNAME>:<DB_PORT>;databaseName=<DB_NAME>;integratedSecurity=true;sendStringParametersAsUnicode=true	
ユーザー名	完全修飾Windowsアカウント: **ドメイン\ユーザー名**	
パスワード	**ドメイン\ユーザーパスワード** -	

注:
互換性のない照合順序に関するエラーを無視します。ただし、照合順序を、「データベース環境の準備」セクションの「言語サポート」トピックで説明されているサポートされる照合順序に設定するようにします。

注: 認証に対してドメインアカウントを使用すると、ドメイン管理者によって変更される可能性のあるセキュリティポリシーが強制されることがあります。

Oracle RAC例

以下は、明示的なJDBC URLを使用したOracle RACクラスターへの接続例です。データベース(ユーザー)はDBAによって事前に作成され、データベースに対してDMLとDDL権限を付与する必要があります。

インストールウィザードで以下の詳細を使用して、強調表示された値をご使用の環境に合わせて修正します。

フィールド	値	コメント
JDBCドライバーjar	C:\my\path\ojdbc7-12.1.0.2.jar	
JDBCドライバークラス名	oracle.jdbc.OracleDriver	
JDBC URL	jdbc:oracle:thin:@(DESCRIPTION=(LOAD_BALANCE=on)(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=Server1_VIP)(PORT=1521))(ADDRESS=(PROTOCOL=TCP)(HOST=Server2_VIP)(PORT=1521)))(CONNECT_DATA=(SERVICE_NAME=RAC1.MY.DOMAIN)))	

フィールド	値	コメント
ユーザー名	<PRE-CREATED-DATABASE>	
パスワード	< PRE-CREATED-DATABASE-PASSWORD>	

関連トピック

データベース環境の準備 (23ページ)

Operations Orchestrationのセキュリティ保護とハードニング

このセクションでは、安全な方法でOOインスタンスをデプロイし、管理する方法、およびOOのセキュリティ強化の構成方法について説明します。

概要

このセクションは、Micro Focus Operations Orchestration (OO) のインスタンスを安全な方法でデプロイおよび管理するITの専門家を支援することを目的としています。OOのさまざまな機能について十分な知識を持って決定を下すことができるように支援し、企業のセキュリティに対する最新ニーズを満たすことを目的としています。

> **重要:** 企業のセキュリティ要件は常に進化しているため、このセクションでは厳しい要件に対応できるようにMicro Focusが最善を尽くしています。このセクションでカバーしていないセキュリティ要件がある場合は、記録しますのでサポート事例をMicro Focusのサポートチームに率直にお話ください。お話いただきましたサポート事例は、このセクションの今後の版に掲載します。

テクニカルシステムランドスケープ

OOは、Java 2 Enterprise Edition (J2EE) テクノロジーをベースとするエンタープライズワイドなアプリケーションです。J2EEテクノロジーは、エンタープライズアプリケーションを設計、開発、アセンブル、デプロイするためのコンポーネントベースの手法を提供します。

セキュリティ更新

OO 10.20と10.50の間では、以下のセキュリティ更新が行われました。

- Centralで [**ログインしているユーザーの資格情報のキャプチャーを有効にする**] チェックボックスが選択されている場合は、OOは、ログインしているユーザーがリモートデバッガーでフローを実行した時に、そのユーザーの資格情報を安全な方法で一時的にキャプチャーします。資格情報がキャプチャーされる可能性があることを警告するメッセージが表示されます。
- OO 10.5xでは、デフォルトでは、デフォルトの役割はありません。ユーザーが取得できる役割は、自分または自分のLDAPグループに明示的に割り当てられた役割に限られるため、管理者はユーザーの権限付与をより適切に制御できます。
- OOに複数のLDAP構成がある場合、管理者がそのいずれかにデフォルトのフラグを付けると、それに属しているユーザーはログイン時にドメインを選択する必要がありません。
- OO 10.5xは、実行中は機密データ (パスワードなど) をセキュリティで保護します。Studioで変数を機密とマークした場合は、スクリプトレットへの使用時に、変数が暗号化形式で取得されます。

OO 10.10と10.20の間では、以下のセキュリティ更新が行われました。

- OOでシステムアカウントのアクセス許可を付与することができるようになりました。これにより、どのユーザーがどのシステムアカウントを表示可能か、またそのアカウントを使用するフローを実行可能かについて、管理者が制御できます。この機能は、複数の組織があり、一部のシステムアカウントを一部のユーザーに表示しないようにする場合便利です。

- [アクセス許可の編集] ダイアログボックスで、アクセス許可を複数の役割に適用できるようになりました。以前のバージョンでは、一度に1つの役割しか選択できませんでした。
- OOインストールを前の10.xバージョンからアップグレードする場合、Oracleから発行された最新の信頼されたルート証明書を含むようにSSL信頼ストアが更新されます。この処理では、期限切れの証明書の削除と、新しい証明書のインポートが行われます。
- OOでイベントを監査するオプションが提供され、セキュリティ違反を追跡できるようになりました。監査を行うと、Central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。

  監査証跡は、現在のところAPI経由のみで取得できます。
- OOが、2048ビット長 (およびそれ以上) の暗号化キーをサポートするようになりました。これで、OOで使用する暗号化キーがFIPS 186-4標準に添うようになります。
- **server.xml** (**<インストールフォルダー>/central/tomcat/conf/server.xml**) ファイルに新しくsslEnabledProtocolsプロパティが追加されました。

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

  このプロパティにより、TLS v1、TLS v1.1、TLS v1.2だけを許可し、SSL 3.0は許可しないことを徹底できます。これは、“POODLE” 攻撃 (Padding Oracle On Downgraded Legacy Encryption) に対する脆弱性を防止します。

セキュリティの概要

このセクションでは、OOの安全な実装を実現するためのセキュリティモデルと推奨事項の概要を説明します。これには、認証、権限付与、暗号化などが含まれます。該当する場合には、他のOOのトピックへの参照もあります。ドキュメントでは、セキュリティ関連のタスクを完了する方法を説明しています。

セキュリティの概念

OO用語集

用語	説明
役割のアクセス許可	アクセス許可とは、あらかじめ定義されたタスクの実行許可です。OO Centralには、役割に割り当てられるアクセス許可のセットがあります。 たとえば、**スケジュール**のアクセス許可は、実行スケジュールを表示および作成できる権限を付与します。
役割	役割とは、アクセス許可の集まりです。たとえば、[**フロー管理者**] の役割は、[**スケジュールの表示**] アクセス許可と[**スケジュールの管理**] アクセス許可を割り当てることができます。
ユーザー	ユーザーは、個人を表すオブジェクトであり、個人に関連付けられ、その権限付与を定義します。役割はユーザーに割り当てられ、Centralでの実行が許可される操作を定義します。たとえば、ユーザー「ジョー・スミス」には、[**フロー管理者**] の役割を割り当てることができます。 別のタイプのユーザーを構成することもできます。

	• [**LDAPユーザー**] は、LDAPユーザー名とパスワードでCentralにログオンします。たとえば、Active Directoryユーザー名とパスワードを使用します。 • [**内部ユーザー**] は、Centralでローカルに設定したユーザー名とパスワードでCentralにログオンします。 • **LW SSO** - Lightweight Single Sign On (SSO) は、1回のユーザー認証および権限付与の操作で、LW SSOをサポートするすべてのシステムにユーザーがアクセスできるようにするメカニズムです。たとえば、ユーザーがLW SSOが有効な別製品のWebクライアントにログオンした場合、このユーザーは、OO Centralログオン画面をバイパスして、直接OO Centralアプリケーションに入ることができます。 同じ役割を持つ内部ユーザーとLDAPユーザーがログインした場合、両者のアクセス許可に違いはありません。 **注:** LDAPユーザーはLDAPプロバイダーが実装したポリシーに従ってセキュリティで保護されているため、内部ユーザーよりLDAPユーザーの使用をお勧めします。
コンテンツのアクセス許可	コンテンツのアクセス許可は、個々のフローまたは特定のフォルダーのフローを表示または実行するための権限です。 特定の役割に割り当てられたユーザーは、その役割に割り当てられたコンテンツ権限に従ってフローにアクセスできます。 たとえば、[**管理者**] の役割を持つユーザーは、システム内のすべてのフローを表示および実行できますが、[**ユーザー**] の役割を持つユーザーは、特定のフローの実行と他のフローの表示のアクセス許可を付与される場合があります。
一般的なセキュリティの概念	
システムのセキュリティ	コンピューターベースの機器、情報、サービスが意図しないまたは許可されていないアクセス、変更、または損傷から保護するためのプロセスおよびメカニズム。
最小限の権限	通常の動作を許可する最小限のレベルに制限する方法。つまり、ユーザーアカウントにユーザーの作業に不可欠な権限だけを付与します。
認証	通常はユーザー名とパスワード、または証明書に基づいて個人を識別するプロセス。
権限付与	個人のIDに基づいたシステムオブジェクトへのアクセス許可。
暗号化	コンテンツにスクランブルをかけて、正しい暗号化キーを持っている人だけが読み取ってエンコードできるようにすることにより、メッセージやファイルのセキュリティを強化する方法。たとえば、TLSプロトコルは通信データを暗号化します。
対策	脅威リスクを低減する方法。
多層防御	保護層。1つのセキュリティ対策だけに依存する必要はありません。
リスク	損傷の原因となる可能性があるイベント。たとえば、財務上の損失、企業イメージへのダメージなど。
脅威	脆弱性を利用したリスクイベントのトリガー。
脆弱性	セキュリティ脅威によって利用される可能性のあるターゲットの弱点。

関連トピック

主要コンセプト

安全な実装およびデプロイメント

OOのセキュリティハードニング

「ハードニング」セクションには、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が示されています。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。

OOシステムを包括的に保護するには、OOのセキュリティの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 (インフラストラクチャーやオペレーティングシステムなど) のセキュリティ保護の両方が必要です。

このセクションには、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も示されています。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。

物理的セキュリティ

組織が定義する物理的なセキュリティ管理によってOOを保護することをお勧めします。OOサーバーコンポーネントは、ベストプラクティスに従って、物理的にセキュリティ保護された環境にインストールされています。たとえば、サーバーはアクセス制御された密室に設置する必要があります。

ネットワークおよび通信のセキュリティ

『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。

『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

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」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

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、『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。

関連トピック

Central TLSサーバー証明書の置き換え (92ページ)

トポロジのセットアップ – ワーカーとRAS

管理インタフェースのセキュリティ

管理インタフェースへのアクセス

管理インタフェースへのアクセスを制御するにはいくつかの方法があります。

- 資格情報
- クライアント証明書
- SAML

管理インタフェースのセキュリティ保護 - 推奨事項

1. Centralで認証を有効にする必要があります。
2. TLSプロトコルを使用して管理インタフェースをセキュリティで保護することをお勧めします。クライアントとCentralインタフェースの間のTLSを設定して暗号化する必要があります。
3. LDAPユーザーの方が安全なので、内部ユーザーよりLDAPユーザーを使用して作業することをお勧めします。
4. クライアント証明書を使用してCentralにアクセスするための認証を設定することをお勧めします。これは、ユーザーパスワードより安全です。

関連トピック

サーバーおよびクライアント証明書の使用 (91ページ)

認証の有効化

ユーザー管理と認証

認証モデル

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。

認証はインストール後すぐに有効にする必要があります。

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(LW SSO)

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。

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。

> **注:**
> 内部ユーザーを使用せずに、LDAPユーザーなど他のより安全なユーザーを使用することをお勧めします。

データベースの認証

OOは4つのデータベースをサポートしています。Oracle、MS SQL、MySQL、Postgresです。

データベース認証用の強いデータベースパスワードと強いパスワードポリシーを使用することをお勧めします。たとえば、何度も試行に失敗したらブロックします。

MS SQLを使用している場合は、データベース認証かOS認証を使用できます。可能であれば、OS認証を使用することをお勧めします。たとえば、Microsoft SQL Serverデータベースへのアクセスには Windows認証を使用できます。

関連トピック

認証の有効化

Operations Orchestrationにおける権限付与

権限付与モデル

OOリソースへのユーザーアクセス権は、ユーザーの役割、およびその役割に対して設定されているアクセス許可に基づいて付与されます。

最小限のアクセス許可に関するガイドライン

推奨事項:

- 役割に適切なアクセス許可を選択します。
- 役割の作成時には最小限のアクセス許可を使用します。
- 最小限のアクセス許可を付与し、必要な場合にだけアクセス許可を拡大して、不必要な権限のエスカレーションを回避します。たとえば、表示のアクセス許可から始め、必要に応じて個別にアクセス許可を追加します。

権限付与の構成

Centralには多数の設定済みの役割がインストールされているので、構成してユーザーに割り当てることができます。デフォルトでは、設定済みの役割には次のアクセス許可が割り当てられています。

役割	デフォルトのアクセス許可
Administrator	すべて
End_user	なし
Everybody	なし
Promoter	すべての**コンテンツ**のアクセス許可
System_admin	すべての**システム**のアクセス許可

デフォルトの役割

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。

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。

関連トピック

セキュリティのセットアップ – 役割

システムアカウントへのアクセス許可の割り当て

コンテンツアクセス許可の設定

OO CentralのIDMモードでの構成

CentralがIDMに接続されると、起動時に認証がデフォルトで有効になります。OO CentralをIDMモードで使用するには、LW-SSOセキュリティプロファイルからIDMセキュリティプロファイルに切り替える必要があります。

IDMセキュリティプロファイルへの切り替え

1. **<インストールディレクトリ>\central\conf\central-wrapper.conf**ファイルをテキストエディターで開き、`# Java Additional Parameters`セクションの`-Doverride.startup.mode`という`wrapper.java.additional`パラメーターを探します。
2. `-Doverride.startup.mode`パラメーターの値を`IDM`に更新します。例:

   ```
   wrapper.java.additional.33=Doverride.startup.mode=IDM
   ```

   LW-SSOセキュリティプロファイルに戻すには、`-Doverride.startup.mode`パラメーターの値を`LWSSO`に更新します。
3. **central-wrapper.conf**ファイルを保存してから閉じます。
4. Centralサービスを再起動します。

IDMプロパティファイルの更新

IDMのプロパティは、**<インストールディレクトリ>\central\conf\idm.properties**ファイルで構成できます。プロパティを設定したら、ファイルを保存して閉じます。その後、Centralサービスを再起動します。

次の表に、**idm.properties**ファイルのプロパティを示します。

プロパティ	説明
idm.configuration.url	IDMサービスのURLを表します。例: <idm_protocol>://<idm_hostname>:<idm_port>/<idm_service_path>
idm.configuration.username	REST IDM統合のユーザー名。
idm.configuration.password	REST IDM統合のパスワード。この値は、**<インストールディレクトリ>\central\bin\encrypt-password.bat**ツールを使用して暗号化できます。
idm.configuration.internal.username	内部アカウントのユーザー名 (IDMサービスに対するPAS認証およびAPI呼び出しに使用されます)。このユーザーには、消費者団体のIDM

プロパティ	説明
	管理オペレーションを実行するためのIDM_ADMIN権限があります。
idm.configuration.internal.password	内部アカウントのパスワード (IDMサービスに対するPAS認証およびAPI呼び出しに使用されます)。この値は、**<インストールディレクトリ>\central\bin\encrypt-password.bat**ツールを使用して暗号化できます。
idm.configuration.scheduler.username	Centralのフロースケジュールの実行に使用されるスケジューリングアカウントのユーザー名。
idm.configuration.scheduler.password	Centralのフロースケジュールの実行に使用されるスケジューリングアカウントのパスワード。この値は、**<インストールディレクトリ>\central\bin\encrypt-password.bat**ツールを使用して暗号化できます。
idm.configuration.signing.key	IDM構成の署名鍵
idm.configuration.oo.central.tenant	IDMサービスで構成されたCentralのテナント値。デフォルト値は`OO_Central`です。
idm.configuration.oo.central.return.url	IDMログインページからリダイレクトされた後のCentralのURLを返します。例: <Centralのプロトコル>://<Centralのホスト名>:<Centralのポート>/oo

IDMモードでのCentralのリターンURLの構成

CentralがIDMモードで実行されている場合、Centralにアクセスしようとするユーザーは、最初にIDMログインページにリダイレクトされ、資格情報を提供します。このリターンURLは、ログインが成功した後、ユーザーをCentralのダッシュボードにリダイレクトするために使用されます。リターンURLの値は、idm.propertiesファイルの`idm.configuration.oo.central.return.url`プロパティを使用して構成できます。

> **注:** リターンURLで使用されるプロトコルとポートが、ログインフロー全体で一貫していることを確認します。たとえば、HTTP経由でログインフローを開始して、HTTPSのリターンURLを提供することはできません。

idm.propertiesファイルの例

```
# IDM Properties

idm.configuration.url=https://idm-hostname:8443/idm-service/

idm.configuration.username=idmTransportUser

idm.configuration.password={ENCRYPTED}uNEBrRr29t/78P700j3BiA==

idm.configuration.internal.username=admin

idm.configuration.internal.password={ENCRYPTED}30l15qHg/7770l15qHgP700d2MiA==
```

```
idm.configuration.signing.key={ENCRYPTED}uWRUrRQ4BzUNr2EG/4PDfB700j9LiA==

idm.configuration.oo.central.tenant=OO_Central

idm.configuration.oo.central.return.url=https://central-hostname:8433/oo
```

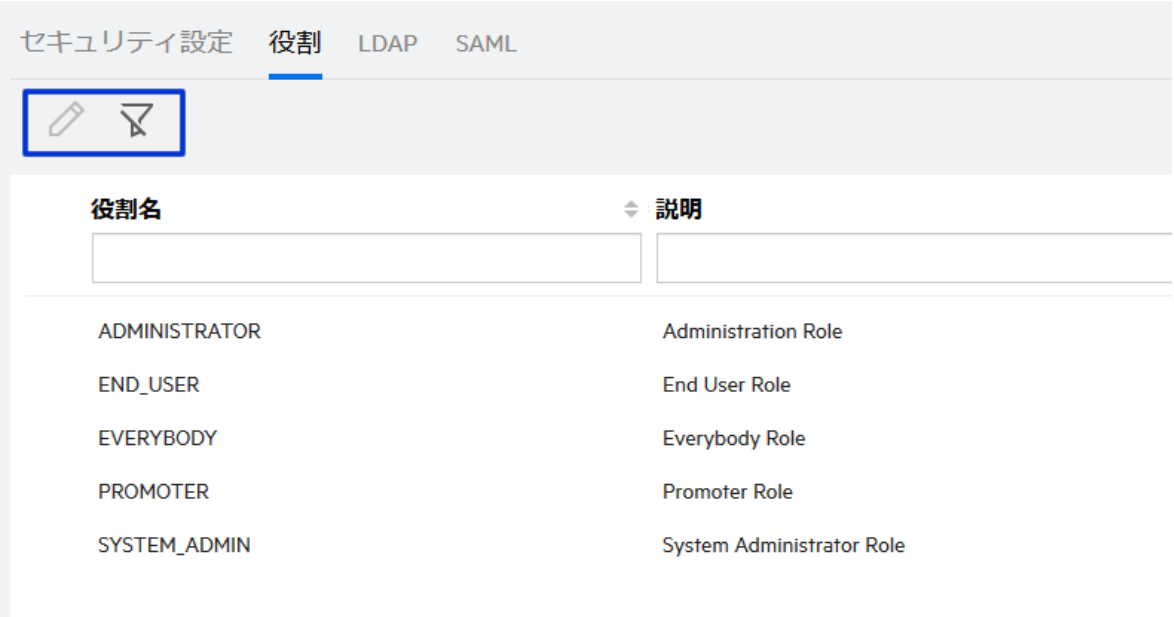
IDMモードでは表示されないタブ

OO認証では、OO管理者は内部ユーザーを定義できますが、IDMモードではOO管理者は内部ユーザーを定義できません。したがって、OOをIDMモードで使用する場合、一部のタブは非表示になります。

次の図は、IDMモードで非表示になっているタブを示しています。

IDMにおける役割

IDMモードでは、OO管理者は追加の役割を作成できません。管理者は、既存の役割のみをグループにマップできます。したがって、UIの [**役割**] タブには、[**編集**] および [**フィルターのクリア**] アイコンのみが表示されます。



LDAPのIDMモードでの構成

Centralが認証用にIDMを使用するように構成されている場合、CentralからIDM用のLDAP構成を直接追加できます。

IDMサーバーは、このLDAP構成を使用して、Centralへのユーザーアクセスの認証と許可を行います。Centralは、IDMサーバー用の単一のLDAP構成をサポートしています。

IDMサーバー用のLDAP構成の追加

1. [**システム構成**] ボタンをクリックして、**システム構成** ワークスペースを表示します。
2. **[セキュリティ] > [LDAP]** を選択します。
3. [**追加**] をクリックします。
   [LDAP] タブに、LDAP設定に関連するフィールドが表示されます。
4. [**ベースDN**] フィールドに、ユーザーを取得するためのルートDNを入力します。
5. [**ユーザーID**] フィールドに、検索特権があるユーザーの識別名を入力します。
   スケジューラーなどの一部のOO機能では検索を使用するため、特権ユーザーを指定することをお勧めします。特権ユーザーがいない場合LDAPの設定内容によってはこの機能が適切に動作しない可能性があります。
   例: `uid=john,ou=people,dc=devlab,dc=ad`

   > **注:** このような特権を持つユーザーの正確なDNを使用してください。

6. [**パスワード**] フィールドに、検索特権があるユーザーのパスワードを入力します。
7. [**接続**] セクションで [**セキュリティで保護されたチャンネル**] を選択して、セキュリティで保護されたチャネルトランスポートをサポートします。
8. [**接続**] セクションで [**追加**] をクリックして、LDAPホストの接続の詳細を追加します。
9. [**ホスト**] フィールドに、LDAPサーバーのIPアドレスまたはホスト名を入力します。

10. [**ポート**] フィールドに、LDAPサーバーのポート番号を入力します。この値は0～64435の間にしてください。
11. [**ユーザー名属性**] フィールドに、ユーザー名を含むLDAP属性を入力します。
12. [**共通名**] フィールドに、画面に表示されるユーザー名を定義する属性を入力します。ほとんどのLDAP実装では、**cn**がデフォルトです。
13. [**ユーザー検索ベース**] フィールドに、ユーザーを取得するためのルートDNを入力します。
14. [**ユーザー検索フィルター**] フィールドに、ユーザーをフィルタリングするための検索フィルターパラメーターを入力します。このフィルターは、[**ユーザー検索ベース**] フィールドに入力したユーザーDNに適用されます。
15. [**ユーザーの電子メール**] フィールドに、ユーザーの電子メールを含むLDAP属性を入力します。
16. ユーザーフィルターを再帰的に実行する場合、[**ユーザーのサブ検索範囲**] チェックボックスを選択します。
17. [**グループメンバーシップ**] フィールドに、ユーザーが属するグループを定義するLDAP属性を入力します。
18. [**マネージャーID**] フィールドに、グループを識別するLDAP属性を入力します。
19. [**マネージャーID値**] フィールドに、[**マネージャーID**] フィールドの検索値を入力します。
20. [**保存**] をクリックして、設定を保存します。

LDAP設定が保存されていない場合、アスタリスクが左側のLDAPナビゲーションペインのドメイン名の横に表示されます。保存する前にページを閉じると、未保存の変更内容があることが警告されます。

データのバックアップ

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**security**フォルダーにあります。

関連トピック

Operations Orchestrationのバックアップと復元

ディザスターリカバリのセットアップ

Operations Orchestrationでの暗号化

暗号化モデル

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、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。

許可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。

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、クリアテキストで表示される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。

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準拠の構成を行うと、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

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、Keytoolを使用して証明書を管理するために次の2つの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。

キーストアおよび信頼ストアへのアクセス制御

信頼ストアおよびキーストアの保存では、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。

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難読化されますが、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。

関連トピック

HTTP/HTTPSポートの変更または無効化 (101ページ)

FIPS 140-2準拠の構成 (113ページ)

Central TLSサーバー証明書の置き換え (92ページ)

コンテンツパックのデプロイと管理

コンテンツパックの構成アイテムのセットアップ

OOのハードニング

このセクションでは、OO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:**
> 管理タスクについては、「管理」セクションを参照してください。

関連トピック

管理

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OOをインストールします。
2. (オプション) FIPS 140-2に準拠するようにOO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。
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。
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、OO Centralサーバーをハードニングします。
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、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。

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。
7. (オプション) TLSプロトコルのバージョンを設定します。
8. Centralでの認証を有効にします。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。
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。
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。
12. Centralで監査が有効なことを確認します。

関連トピック

インストール

OOでのFIPS 140-2レベル1準拠の構成 (111ページ)

サーバーおよびクライアント証明書の使用 (91ページ)

HTTP/HTTPSポートの変更または無効化 (101ページ)

キーストア/信頼ストアのパスワードの変更と暗号化/難読化 (97ページ)

SSLサポート対象暗号からの脆弱性のある暗号の削除 (100ページ)

TLSプロトコルの構成 (117ページ)

認証の有効化

セキュリティのセットアップ – LDAP認証

セキュリティバナーのセットアップ

Microsoft SQL Serverに関する追加のガイドライン (65ページ)

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。

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、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。

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証明書 (秘密キー) が含まれています。

> **注:**
> クライアント証明書をLDAPで使用する場合は、CentralでLDAPをデフォルトとして構成する必要があります。

推奨事項:

- 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。
- 信頼ストアとキーストアは、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。

関連トピック

セキュリティのセットアップ – LDAP認証

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**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 <キーストアのパスワード>
   ```

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。この手順は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に問い合わせて証明書の生成プロセスの詳細を確認してください。

> **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。

   ```
   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。

   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。

8. Centralを起動します。

関連トピック

キーストア/信頼ストアのパスワードの変更と暗号化/難読化 (97ページ)

Centralの信頼ストアへの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**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、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**または**<OOSHディレクトリ>/var/security/client.truststore** (スタンドアロンOOSHの場合) にある元の**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)。スタンドアロンOOSHの場合、これはメインOOSHディレクトリの下にあります。

   ```
   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、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、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**または<OOSHディレクトリ>/var/security/client.truststore (スタンドアロンOOSHの場合) 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>/central/bin**またはスタンドアロンOOSHのメインOOSHディレクトリにある**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信頼ストアへの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、以下の手順を実行する必要はありません。

デフォルトで、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、信頼ストアの場所は**<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。

関連トピック

リモートCentralのStudioでのデバッグ

証明書の失効ステータスの確認

証明書失効リスト (CRL) は、失効済みの証明書のリスト (具体的には、証明書のシリアル番号のリスト) です。このリストの (失効済みの) 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。

RAS側

RASでは、リモートサーバーでCentral証明書が失効していることを特定できます。RASが起動してCentralとハンドシェイクを行う際に、RASはCentral証明書を取得します。この証明書には、CRLファイルの場所へのリンクが含まれています。RASはCRLファイルにアクセスし、このCRLファイルに対してCentral証明書を検証します。

証明書が失効している場合、RASとCentralとの間は接続されません。また、RAS側のログファイルにはエラーメッセージが表示されます。

Central側

Central側では、トポロジ領域にRASがオフライン (未接続) 状態で表示されます。

実行する操作

失効ステータスチェックの有効化

1. RASのラッパーファイル**ras-wrapper.conf**を開きます。このファイルは**<RASインストールディレクトリ>/ras\conf**にあります。
2. RASに次の行を追加します。

   ```
   wrapper.java.additional.<n>=-Dcom.sun.security.enableCRLDP=true
   wrapper.java.additional.<n>=-Dcom.sun.net.ssl.checkRevocation=true
   ```

3. 次のフラグをfalseに設定します。

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

キーストア/信頼ストアのパスワードの変更と暗号化/難読化

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。

   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。

   キーストア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
   ```

   信頼ストア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/Micro Focus/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/Micro Focus/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しないこと、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。

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、信頼ストアのパスワードを、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>
   ```

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:**
> 次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password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password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。**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、信頼ストアのパスワードを取得します。

関連トピック

パスワードの暗号化と難読化 (99ページ)

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>
   ```

パスワードの難読化

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読化するパスワードを指定します。

   難読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:

```
C:\Program Files\Micro Focus\Operations Orchestration\central\bin>encrypt-password.bat
Password (typing will be hidden):
Confirm password (typing will be hidden):
{ENCRYPTED}gAkPCLQsYDhoR1Y2q9bjCQ==

C:\Program Files\Micro Focus\Operations Orchestration\central\bin>
```

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サポート対象暗号からの脆弱性のある暗号の削除

TLSプロトコルの製品で使用されるDESおよびTriple DES暗号は、約40億ブロックのbirthday boundを持っているため、CBCモードでTriple DESを使用するHTTPSセッションで実証されたように、遠隔の攻撃

者は長期間の暗号化セッションに対する誕生日攻撃を通じてクリアテキストを比較的容易に取得できます。これは「Sweet32」攻撃とも呼ばれます。

この攻撃の詳細については、https://sweet32.info/を参照してください。

OOでRC4、DES、およびTriple DES暗号を無効にするには、次の手順を実行します。

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ます。

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

jdk.tls.disabledAlgorithms=DES, DESede, 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-jrelevelを参照してください。

> **注:** 前のバージョンの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/Micro Focus/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/Micro Focus/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。

HTTPSコネクターの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:\Micro Focus\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:/Micro Focus/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」トピックの手順に従ってTLSサーバー証明書を構必要があります。

> **注:** デフォルトでは、ユーザー証明書はOperations Orchestration (OO) Centralの内部ユーザーと照合されます。証明書をLDAPユーザーと照合する場合は、目的のLDAPをデフォルトLDAPとして設定してください。

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スタックがクライアント証明書を要求するようにする場合は、`clientAuth`属性を`want`に設定します。属性が`want`に設定されると

- 証明書が提示されない場合、認証は失敗せず、ユーザーはログインページにリダイレクトされます。
- 無効な証明書が提示された場合、認証が失敗し、ユーザーはログインページにリダイレクトされません。

`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/Micro Focus/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/Micro
   Focus/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。
   - クライアント認証が必要な場合では、OO 9x後方互換のSOAP/REST APIはサポートされません。

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
```

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。
>
> 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信頼ストアにインポートする必要があります。

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。

RAS IDは、Centralの**[トポロジ]**タブで確認できます(「トポロジのセットアップ – ワーカーとRAS」に記載)。

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。

Studio Remote Debugger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信頼ストアにインポートする必要があります。

Studio Remote Debugger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:

- 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。
- 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。
- 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、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信頼ストアにインポートする必要があります。

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(スタンドアロンOOSHの場合、メインOOSHディレクトリの下にあります)。
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。

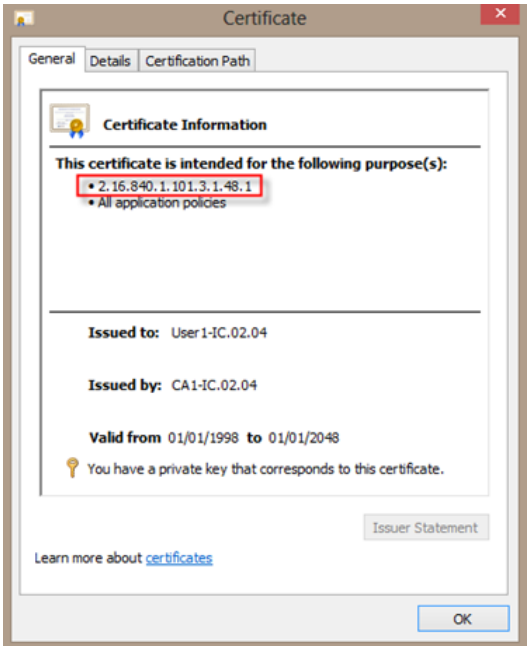
> **注:**
> OO 10.20以降では、パスワードがデフォルトのままだった場合に、keyStorePasswordパラメーター

> 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。
>
> 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。
>
> 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、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

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。
- 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、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、_FTc_Basic_Variables._FT_Product_Acronym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の抽出を試みます。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。

関連トピック

サーバーおよびクライアント証明書の使用 (91ページ)

パスワードの暗号化と難読化 (99ページ)

RAS信頼ストアへのCAルート証明書のインポート (94ページ)

トポロジのセットアップ – ワーカーとRAS

Studio信頼ストアへのCAルート証明書のインポート (96ページ)

OOSH信頼ストアへのCAルート証明書のインポート (95ページ)

OOコンテンツパックへの署名

コード署名とは、ソフトウェアとコンテンツの発行者が証明書ベースのデジタル署名を使用して、コードのユーザーに対して発行者のIDを実証するメカニズムです。ユーザーは発行者の信頼性に基づいてインストールするかどうかを決定できます。

セキュリティの上で以下のようなメリットがあるため、Operations Orchestrationのコンテンツパックを電子署名でコード署名することが重要です。

- Operations Orchestration管理者は、コンテンツパックをOperations Orchestration Centralにデプロイする際に、会社をコンテンツパックの作成者、発行元または配布元として指定できます。
- これにより、コンテンツパックの完全性を提供します。つまり、コンテンツパックが改ざんされないように保護します (コンテンツパックが変更された場合、デジタル署名は無効になります)。

デジタル証明書は、認証局 (CA) によって発行された電子文書です。

デジタル証明書には、デジタル署名の公開キーが含まれ、組織の名前など、鍵に関連付けられたIDを指定します。

証明書は、公開キーが特定の組織に属することを確認するために使用されます。CAは保証人の役割を果たします。

デジタル証明書は、信頼できる機関によって発行され、指定された期間のみ有効です。デジタル署名を作成するためにはデジタル証明書が必要になります。

デジタル署名ファイルを格納する一般的な方法は、.pfx拡張子のPCKS#12ファイルです (例: **digital_signature.pfx**)。

実行する操作

jarsignerユーティリティを使用してコンテンツパックに署名する

1. Java SE 8u102を次のURLからダウンロードします。

   http://www.oracle.com/technetwork/java/javase/downloads/index.html
2. Java SE 8u102をローカルファイルシステム上の適切な場所にインストールし、<JDKインストール>/binにあるjarsignerユーティリティを探します。
3. デジタル署名を使用して、コンテンツパックファイル.jarに署名します。

デジタル署名ファイルをPCKS＃12形式からjarsignerユーティリティで使用されるJKS形式に変換する

1. **<OOインストール>/java/bin/keytool**にあるkeytoolユーティリティを使用して、次のコマンドを実行します。

   ```
   keytool -importkeystore -srckeystore
   <digital_signature.pfx> -srcstoretype pkcs12 -srcalias
   <certificate_alias_pfx> -destkeystore
   <digital_signature.jks> -deststoretype jks -deststorepass
   <password_jks> -destalias <certificate_alias_jks>
   ```

   用語

名前	説明
certificate_alias_pfx	pkcs12形式のソースデジタルファイル内のエイリアス。コンテンツパックJARファイルへの署名に使用される秘密キーと、キーに関連付けられた証明書を識別します。
digital_signature.jks	JKS形式の出力デジタルファイルの名前
deststorepass	JKS形式の出力デジタルファイルのパスワード
certificate_alias_jks	JKS形式の出力デジタルファイル内のエイリアス。コンテンツパックJARファイルに署名するために使用される秘密キーと、キーに関連付けられた証明書を識別します。
2. 次に、次のコマンドを実行します。

   ```
   jarsigner -keystore <digital_signature.jks>
   -signedjar <signed-content-pack.jar> <content_pack.jar>
    <certificate_alias_jks>
   ```

   用語

名前	説明
digital_signature.jks	JKS形式の出力デジタルファイルの名前
signed-content-pack.jar	出力された署名付きコンテンツパックファイルJAR
content_pack.jar	署名の入力に使用されるJARファイル
certificate_alias_jks	JKS形式の出力デジタルファイル内のエイリアス。コンテンツパックJARファイルに署名するために使用される秘密キーと、キーに関連付けられた証明書を識別します。

StudioツールのJARの整合性の検証

JARファイルは<OOインストールフォルダー>/studio/tools/libフォルダー内にあり、OOSHAの実行可能コード、ウィザード、**hpln-index-generator**ツールを提供します。

JARファイルの整合性を検証するため、デジタル署名を確認して、Operations Orchestrationのインストール以降にJARファイルが改ざんされていないことを実証できます。

デジタル証明書は、認証局 (CA) によって発行された電子文書です。

デジタル証明書には、デジタル署名の公開キーが含まれ、組織の名前など、鍵に関連付けられたIDを指定します。

証明書は、公開キーが特定の組織に属することを確認するために使用されます。CAは保証人の役割を果たします。

デジタル証明書は、信頼できる機関によって発行され、指定された期間のみ有効です。デジタル署名を作成するためにはデジタル証明書が必要になります。

実行する操作

StudioツールのJARのデジタル署名を確認する

1. Java SE 8u102を次のURLからダウンロードします。
   http://www.oracle.com/technetwork/java/javase/downloads/index.html
2. Java SE 8u102をローカルファイルシステム上の適切な場所にインストールし、次の場所からjarsignerユーティリティを探します。
   <JDK_INSTALLATION>/bin
3. コマンドラインから次のコマンドを実行します。

   ```
   jarsigner -verify -keystore
   <oo_installation>/central/var/security/internal.truststore
    -strict -verbose -certs <jar_file>
   ```

   <jar_file>は、次のいずれかのjarファイルです。
   - hpln-index-generator.jar
   - oosha-jar-with-dependencies.jar
   - ps-wizard-jar-with-dependencies.jar

- rest-wizard-jar-with-dependencies.jar
- shell-wizard-jar-with-dependencies.jar
- third-party-cp-wizard-jar-with-dependencies.jar
- ws-wizard-jar-with-dependencies.jar

コマンドの出力は "jar verified" で終了すると予想されます。さらに、出力にリストされているすべてのJARファイルのエントリには、"smk" ラベルが付いていなければなりません。

OOでのFIPS 140-2レベル1準拠の構成

当セクションでは、Operations Orchestration (OO) をFederal Information Processing Standards (FIPS) 140-2レベル1準拠になるよう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

OOでFIPS 140-2準拠の構成を行うと、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

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

前提条件

アップグレードする場合の注意事項:

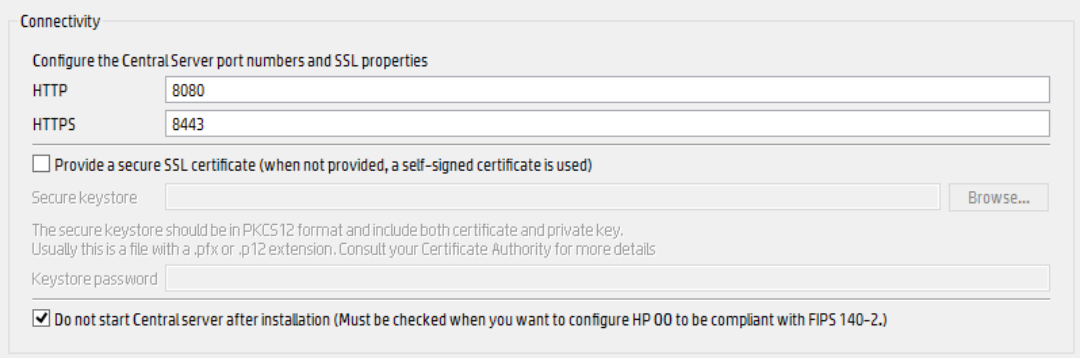
すでにFIPSで構成済みのOO 10.10 (およびそれ以降) のインストールからアップグレードする場合は、「Operations Orchestration 10.xの旧バージョンからOperations Orchestration 10.8xへのアップグレード」の「FIPSを使用したアップグレードの手順」を参照してください。

OOでFIPS 140-2準拠の構成を行う前は、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準拠の構成には、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. 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**フォルダー一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。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、OOが使用するJREのインストール先。これは、デフォルトでは**<インストールディレクトリ>\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**

アップグレードプログラムの前提手順

1. 次のサイトから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。
   http://www.oracle.com/technetwork/java/javase/downloads/server-jre8-downloads-2133154.html
   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。

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。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、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

関連トピック

FIPS 140-2準拠の構成 (113ページ)

Operations Orchestration 10.xの旧バージョンからOperations Orchestration 10.8xへのアップグレード

FIPS 140-2準拠の構成

FIPS 140-2に準拠するためにOOで必要な構成手順を次のリストに示します。

ステップ1. Javaセキュリティファイルのプロパティの構成

FIPS 140-2準拠のために、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、そのプロパティを構成します。

> **注:**
> 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、「OOでのFIPS 140-2レベル1準拠の構成」セクションの「アップグレードプログラムの前提手順」トピックを実行してから、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。

ステップ2. encryption.propertiesファイルの構成とFIPSモードの有効化

OO暗号化プロパティファイルは、FIPS 140-2に準拠するように更新する必要があります。

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\Micro Focus\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。

ステップ3. FIPS準拠の暗号化の作成

FIPS準拠の設定には、OO暗号化ストアファイルの作成または置換が必要です。

> **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。

注:
使用していないデフォルトのCAルート証明書は、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)。

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。

ステップ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

データベースパスワードを、「データベースパスワードの変更」トピックの説明に従って、再暗号化します。

ステップ5. OOの起動

関連トピック

OOでのFIPS 140-2レベル1準拠の構成 (111ページ)

FIPS暗号化の置き換え (115ページ)

キーストア/信頼ストアのパスワードの変更と暗号化/難読化 (97ページ)

データベースパスワードの変更 (17ページ)

FIPS暗号化の置き換え

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。

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. 「OOでのFIPS 140-2レベル1準拠の構成」セクションの「前提条件」の手順をすべて実行します。
2. 「FIPS 140-2準拠の構成」セクションの「Javaセキュリティファイルのプロパティの構成」トピック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。
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。

関連トピック

OOでのFIPS 140-2レベル1準拠の構成 (111ページ)

FIPS 140-2準拠の構成 (113ページ)

TLSプロトコルの構成

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。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なるようにする

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なるようにしたり、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、Operations Orchestration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:${oo.home}/var/cache/-" {
   permission java.lang.RuntimePermission "getClassLoader";
   permission java.io.FilePermission "${oo.home}/var/cache/-",
     "read, write";
   permission java.io.FilePermission "${oo.home}/var/logs",
     "read, write";

};
```

> **注:**
> 実行するコンテンツにより、アクセス許可の追加が必要になることがあります。

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
                  .....
};
```

Javaセキュリティマネージャーの追加

ユーザーがローカルホストサーバーをシャットダウンするフローを作成することがないようにするには、「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なるようにする」に記載されている仕様を適用します。

以下のセクションでは、スタンドアロンのRASとCentralに組み込まれたワーカーのJavaポリシーファイルを設定する方法について説明します。

Javaセキュリティマネージャーを追加するようRASを構成する

スタンドアロンRASでセキュリティマネージャーを有効にするには:

1. **%RAS_HOME%\conf**フォルダーに`ras.policy`という名前のテキストファイルを作成し、次の内容を追加します。

```
grant codeBase "file:${oo.home}/lib/score-worker-manager-impl.jar"
{
permission java.lang.RuntimePermission "exitVM.75";
}
```

2. **%RAS_HOME%\conf**フォルダーにある**ras-wrapper.conf**ファイルに、次のラッパーを追加します。

```
wrapper.java.additional.<next_index>=-Djava.security.policy="%RAS_
HOME%/conf/ras.policy"
```

3. RASを再起動します。

Javaセキュリティマネージャーを追加するようにCentral組み込みワーカーを構成する

Central組み込みワーカーでセキュリティマネージャーを有効にするには:

1. `%CENTRAL_HOME%/tomcat/conf/catalina.policy`ファイルの**System Code Permissions**の前に次の内容を追加します。

```
grantcodeBase "file:${oo.home}/tomcat/webapps/oo/WEB-INF/lib/score-
worker-manager-impl.jar"
{
permission java.lang.RuntimePermission "exitVM.75";
}
```

関連トピック

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なるようにする (117ページ)

HTTPSへのリダイレクトの有効化

インストールウィザードで選択した **[Allow HTTP access]** を変更して、HTTPトラフィックをHTTPSにリダイレクトする場合は、**web.xml**ファイルで手動で設定できます。

1. Centralサーバーを停止します。
2. **<インストールフォルダー>/central/tomcat/conf/web.xml**ファイルを開きます。
3. `<security-constraint>`プロパティを見つけます。
4. `<transport-guarantee>`の値を`CONFIDENTIAL`に変更します。

   例:

```
<security-constraint>

<web-resource-collection>

<web-resource-name>HTTPSOnly</web-resource-name>

<url-pattern>/*</url-pattern>

</web-resource-collection>

<user-data-constraint>

<transport-guarantee>CONFIDENTIAL</transport-guarantee>

</user-data-constraint>
```

```
</security-constraint>
```

5. Centralサーバーを起 動します。

ドキュメントのフィードバックを送信

本ドキュメントについてのご意見、ご感想については、電子メールでドキュメント制作チームまでご連絡ください。このシステムで電子メールクライアントが設定されていれば、このリンクをクリックすることで、以下の情報が件名に記入された電子メールウィンドウが開きます。

Feedback on 管理ガイド (Micro Focus Operations Orchestration 2018.05)

本文にご意見、ご感想を記入の上、**[送信]**をクリックしてください。

電子メールクライアントが利用できない場合は、上記の情報をコピーしてWebメールクライアントの新規メッセージに貼り付け、docs.feedback@microfocus.com宛にお送りください。

お客様からのご意見、ご感想をお待ちしています。