NEC

iStorage V100/V300

HA Command Suite Configuration Manager REST API リファレンスガイド



iStorage

IV-UG-212-001-08

対象製品

HA Command Suite Configuration Manager 11.0.4

輸出時の注意

本製品を輸出される場合には、外国為替及び外国貿易法の規制並びに米国輸出管理規則など外国の輸出関連法規をご確認の上、必要な手続きをお取りください。

なお、不明な場合は、弊社担当営業にお問い合わせください。

商標類

Linux は，Linus Torvalds 氏の米国およびその他の国における登録商標です

Microsoft は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

Oracle と Java 及び MySQL は、Oracle、その子会社及び関連会社の米国及び その他の国における登録商標です。

This product includes software developed by IAIK of Graz University of Technology.

Red Hat is a registered trademark of Red Hat, Inc. in the United States and other countries.

Red Hat は，米国およびその他の国における Red Hat, Inc.の登録商標です。

Red Hat Enterprise Linux is a registered trademark of Red Hat, Inc. in the United States and other countries.

Red Hat Enterprise Linux は，米国およびその他の国における Red Hat, Inc.の登録商標です。

すべての SPARC 商標は，米国 SPARC International, Inc. のライセンスを受けて使用している同社の米国およびその他の国における商標または登録商標です。SPARC 商標がついた製品は，米国 Sun Microsystems, Inc. が開発したアーキテクチャに基づくものです。

Windows は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

Windows Server は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

その他記載の会社名，製品名などは，それぞれの会社の商標もしくは登録商標です。

Java is a registered trademark of Oracle and/or its affiliates.

発行

2025 年 10 月 (IV-UG-212)

著作権

© NEC Corporation 2021-2025

目次

第 1 章 REST API の環境構築 1

1.1 REST API のシステム構成 1
1.1.1 管理対象のストレージシステム 1
1.1.2 REST API のシステム構成 1
1.2 REST API の導入方法 2
1.3 REST API を使用するための準備の流れ 3
1.4 管理サーバの要件および前提プログラムの確認 3
1.5 REST API のインストール 5
1.5.1 REST API のインストール先の条件 5
1.5.2 root ユーザで REST API をインストールする（Linux の場合） 6
1.5.3 Amazon Corretto 17 をアップグレードする 7
1.5.4 ウィルス検出プログラムおよびプロセス監視ソフトウェアを使用する場合に必要な設定 8
1.6 REST API で使用するポート番号の設定 8
1.6.1 REST API で使用するポート 9
1.6.2 HTTPS 通信に使用するポート番号を変更する 10
1.6.3 HTTP 通信に使用するポート番号を変更する 11
1.6.4 HTTP 通信を無効化する 11
1.6.5 HTTP 通信を有効化する 12
1.6.6 HTTP 通信および HTTPS 通信に使用するポート番号の設定を初期状態に戻す. 13
1.6.7 REST API サーバが使用するポート番号を変更する 14
1.6.8 ストレージシステムの構成変更の通知を受信するポート番号を変更する 15
1.7 SSL 通信の設定 16
1.7.1 Configuration Manager REST API サーバと Platform REST API サーバ間の SSL 通信路 19
1.7.2 REST API クライアントと REST API サーバ間で SSL 通信するよう設定する（自己署名証明書を使用する場合） 20
1.7.3 REST API クライアントと REST API サーバ間で SSL 通信するよう設定する（認証局が発行したサーバ証明書を使用する場合） 23
1.7.4 Configuration Manager REST API サーバと Platform REST API サーバ間で SSL 通信するよう設定する 26
1.7.5 Configuration Manager REST API サーバとストレージシステム間の SSL 通信に使用する証明書を変更する（構成変更の通知を受信する場合） 27
1.8 REST API のサービスの起動と停止 30

1.8.1 REST API のサービスの起動 30
1.8.2 REST API のサービスの停止 31
1.8.3 REST API のサービスの稼働状態の確認 31
1.9 REST API のアンインストール 32
1.9.1 root ユーザで REST API をアンインストールする（Linux の場合） 32
第 2 章 REST API の共通仕様 34
2.1 管理対象のリソースの指定 34
2.2 オブジェクト ID の指定方法 35
2.3 サポートする HTTP メソッド 36
2.4 ユーザ認証 37
2.5 セッション管理 40
2.6 リクエストヘッダ 41
2.7 レスポンスヘッダ 43
2.8 HTTP ステータスコード 43
2.9 リクエストおよびレスポンスのフォーマット 45
2.10 クエリパラメータ 46
2.11 データ型 46
2.12 出力形式 47
2.13 データオブジェクト 47
2.14 ジョブオブジェクト 48
2.15 エラーオブジェクト 49
2.16 リクエストオブジェクト 51
2.17 Action テンプレートオブジェクト 51
第 3 章 REST API で共通の操作 53
3.1 バージョン情報を取得する 53
3.2 ストレージシステムの一覧を取得する 54
3.3 特定のストレージシステムの情報を取得する 56
3.4 ストレージシステムを登録する 58
3.5 ストレージシステムの情報を変更する 62
3.6 ストレージシステムの情報を削除する 64
3.7 セッションの一覧を取得する 66
3.8 特定のセッションの情報を取得する 68

3.9 セッションを生成する 69
3.10 セッションを破棄する 71
3.11 ジョブの情報の一覧を取得する 73
3.12 特定のジョブの情報を取得する 76
第 4 章 ストレージシステムの情報検索 79
4.1 REST API サーバのデータベースを最新にする方法 79
4.2 ストレージシステムの構成情報の更新 80
4.2.1 ストレージシステムの構成情報の更新状態を取得する 80
4.2.2 ストレージシステムの構成情報を更新する 82
付録 A. REST API サーバの通信モードの変更 85
A.1 REST API サーバの通信モードの変更とは 85
A.2 REST API サーバの通信モードを変更するための設定 86
A.3 REST API サーバの通信モードを変更する 87
付録 B. バックアップとリストア 89
B.1 REST API のデータベースおよび環境設定ファイルをバックアップする 89
B.2 REST API のデータベースおよび環境設定ファイルをリストアする 90
付録 C. トラブルシューティング 92
C.1 障害発生時に採取が必要な情報 92
C.2 REST API の保守情報を取得する 92
付録 D. ストレージシステムの構成変更の通知 94
D.1 ストレージシステムの構成変更の通知とは 94
D.1.1 構成変更の通知の処理の流れ 94
D.2 ストレージシステムの構成変更の通知先の一覧を取得する 96
D.3 ストレージシステムの構成変更の特定の通知先を取得する 98
D.4 ストレージシステムの構成変更の通知先を登録する 100
D.5 ストレージシステムの構成変更の通知先を削除する 102
付録 E. Configuration Manager のバージョン 104
E.1 Configuration Manager バージョン対応表 104
付録 F. リトライ処理の組み込み 105
F.1 リトライ処理の組み込み 105
F.2 リトライ処理のコード例 106

付録 G. このマニュアルの参考情報 109

G.1 このマニュアルでの表記 109

G.2 このマニュアルで使用している略語 109

G.3 KB（キロバイト）などの単位表記について 110

索引 112

はじめに

このマニュアルは、HA Command Suite Configuration Manager REST API の運用方法について説明したものです。

HA Command Suite Configuration Manager REST API は、ストレージシステムの情報取得や構成変更を行うための、REST（Representational State Transfer）の原則に従った Web API を提供します。

この REST API は、NEC Storage Plug-in for VMware vCenter で利用します。それ以外の用途では利用しないでください。

対象読者

このマニュアルは、次の方を対象読者として記述しています。

- ストレージシステムの運用に関する知識がある方
- REST API を利用したプログラムを作成するスキルがある方

マニュアルの構成

このマニュアルは、次に示す章と付録から構成されています。

第 1 章　REST API の環境構築

REST API を利用してストレージシステムを運用するために必要な環境構築について説明しています。

第 2 章　REST API の共通仕様

REST API でのリソースの指定方法、リクエストとレスポンスの形式および各オブジェクトについて説明しています。

第 3 章　REST API で共通の操作

セッションの生成やジョブの情報取得など、REST API で共通の操作について説明しています。

第 4 章　ストレージシステムの情報検索

REST API で実行するストレージシステムのリソースの情報検索について説明しています。

付録 A　REST API サーバの通信モードの変更

REST API サーバとストレージシステム間の接続方法を変更し、REST API サーバの通信モードを変更することによって、REST API の処理速度を向上する方法について説明しています。

付録 B　バックアップとリストア

REST API のデータベースおよび環境設定ファイルのバックアップ、リストアについて説明しています。

付録 C　トラブルシューティング

REST API サーバで障害が発生した場合の対処方法について説明しています。

付録 D　ストレージシステムの構成変更の通知

ストレージシステムの構成変更を通知する機能の概要と、その機能を利用して REST API のデータベースを更新する方法について説明しています。

付録 E　Configuration Manager のバージョン

対象製品のバージョン、REST API のバージョン、およびストレージシステムのマイクロコードのバージョンの対応について説明しています。

付録 F　リトライ処理の組み込み

REST API を使用したスクリプトにリトライ処理を実装する上で、考慮すべき点について説明しています。

付録 G　このマニュアルの参考情報

このマニュアルを読むに当たっての参考情報を説明しています。

マイクロソフト製品の表記について

このマニュアルでは、マイクロソフト製品の名称を次のように表記しています。

表記	製品名
Windows	次の製品を区別する必要がない場合の表記です。 • Windows® 10 • Microsoft® Windows Server® 2016 • Microsoft® Windows Server® 2019 • Microsoft® Windows Server® 2022 • Microsoft® Windows Server® 2025

図中で使用している記号

このマニュアルの図中で使用している記号を、次のように定義します。



このマニュアルで使用している記号

このマニュアルでは、次に示す記号を使用しています。

表記	製品名
＜ ＞	可変値であることを示します。

第 1 章
REST API の環境構築

この章では、REST API を利用してストレージシステムを運用するために必要な環境構築について説明します。

1.1　REST API のシステム構成

REST API の管理対象のストレージシステムや、REST API のシステム構成について説明します。

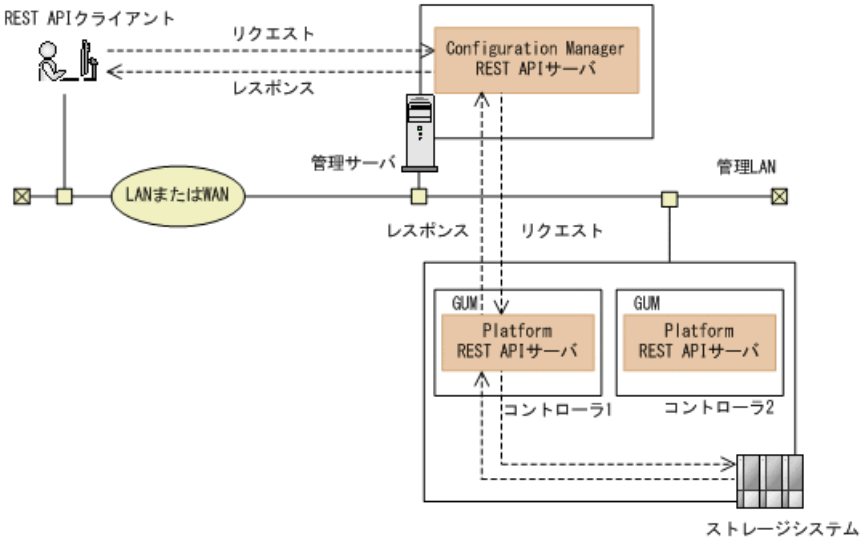
1.1.1　管理対象のストレージシステム

REST API は次に示すストレージシステムを対象としています。

ストレージシステム	サポートするファームウェアバージョン
iStorage V100、iStorage V300	93-07-27-XX 以降

1.1.2　REST API のシステム構成

REST API のシステム構成を次に説明します。



iStorage V シリーズの場合、ストレージシステムに REST API が内蔵されています。このマニュアルでは、この REST API を Platform REST API と呼びます。

REST API クライアント

REST API サーバへリクエストを発行するクライアントです。REST API を利用したソフトウェアまたはスクリプトが該当します。

管理サーバ

Configuration Manager REST API をインストールするサーバです。

Configuration Manager REST API サーバ

REST API クライアントから REST API のリクエストを受け付け、ストレージシステムに命令を発行し、実行結果を REST API クライアントに返す役割を担うコンポーネントです。

Configuration Manager REST API サーバは、REST API クライアントから受け付けたリクエストを、Platform REST API サーバに送信し、Platform REST API サーバから受け取った実行結果を REST API クライアントに返します。

メモ

コントローラ 1 側とコントローラ 2 側の Platform REST API サーバは、それぞれ独立して動作します。

デフォルトでは、コントローラ 1 側の Platform REST API サーバが使用されます。コントローラ 2 側の Platform REST API サーバに変更することもできます。その場合は、ストレージシステムの登録または情報変更の API で、操作対象のコントローラを変更してください。

GUM (Gateway for Unified Management)

ストレージシステムの基本的な管理機能を持つコンピュータです。外部からストレージシステムを管理する場合には、GUM と通信します。コントローラ 1 とコントローラ 2 に存在します。

Platform REST API サーバ

iStorage V シリーズの GUM に内蔵されている REST API のサーバです。REST API クライアントから REST API のリクエストを受け付け、ストレージシステムに命令を発行し、実行結果を REST API クライアントに返します。

ストレージシステム

REST API での情報取得や構成変更の対象となるストレージシステムです。

関連リンク

管理サーバの要件および前提プログラムの確認（3 ページ）

1.2　REST API の導入方法

REST API のインストールメディアを使用してインストールします。

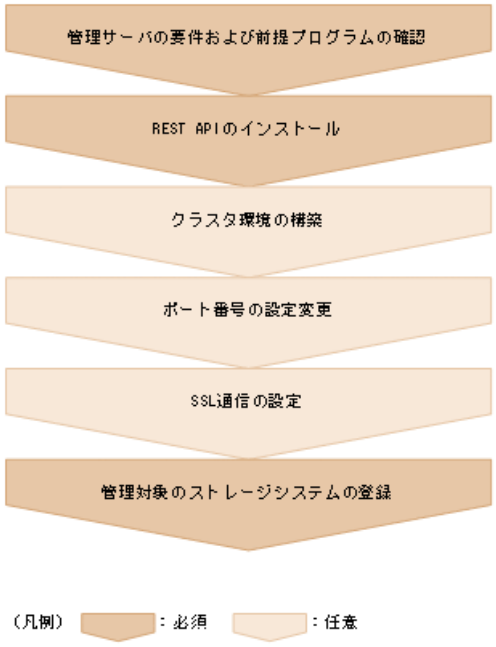
関連リンク

REST API を使用するための準備の流れ（3 ページ）
root ユーザで REST API をインストールする（Linux の場合）（6 ページ）

1.3　REST API を使用するための準備の流れ

REST API を使用するために必要な環境設定や操作について、全体的な流れを説明します。

Linux の root ユーザで REST API をインストールした環境で REST API を運用するための準備の流れを示します。



関連リンク

管理サーバの要件および前提プログラムの確認（3 ページ）
REST API のインストール（5 ページ）
REST API で使用するポート番号の設定（8 ページ）
SSL 通信の設定（16 ページ）
ストレージシステムを登録する（58 ページ）

1.4　管理サーバの要件および前提プログラムの確認

REST API をインストールする前に、管理サーバのマシンや OS の要件を確認し、前提プログラムとして OS のライブラリをインストールしておく必要があります。また、必要に応じて、RAID Manager のインストール状況を確認してください。

管理サーバの要件

管理サーバの要件、必要な OS のライブラリについては、『ソフトウェア添付資料』で事前に確認してください。

REST API は、DHCP による動的な IP アドレスが割り振られている管理サーバ上では運用できません。管理サーバ（Configuration Manager REST API サーバ）には、固定の IP アドレスを設定してください。

1 台のストレージシステムのコマンドデバイスが複数のゲスト OS にマッピングされ、それらのゲスト OS 上に REST API をインストールしてストレージシステムを管理する場合は、それぞれのゲスト OS は異なる物理サーバ上で動作する必要があります。REST API がインストールされたそれぞれのゲスト OS が同一の物理サーバで動作する場合、ストレージシステムがゲスト OS 間の違いを区別できないことが原因で予期しないエラーが発生する可能性があります。

RAID Manager

管理サーバに RAID Manager がインストールされていない場合、REST API をインストールすると REST API が同梱している RAID Manager が一緒にインストールされます。このとき、RAID Manager のインストール先へのシンボリックリンクが作成されます。

Linux の場合：

ルートディレクトリに、＜*REST API のインストール先*＞/HORCM へのシンボリックリンクが作成されます。

メモ

- REST API に同梱されている RAID Manager がインストールされた環境では、RAID Manager のファイルが使用されていると、REST API のアップグレードインストール、上書きインストール、アンインストールができません。これらの操作を行う場合は、事前に RAID Manager を使用しているプログラムを停止してください。

メモ

REST API に同梱されている RAID Manager がインストールされた環境に対して、RAID Manager を新たに個別でインストールしてその RAID Manager を使用する場合（REST API が使用する RAID Manager を切り替える場合）、事前に REST API のサービスを停止してから、次の操作を実施してください。そのあと、REST API のサービスを起動してください。

Linux の場合：

ルートディレクトリにある HORCM シンボリックリンクを削除してから、RAID Manager を通常の手順でルートディレクトリに新規インストールしてください。

メモ

- REST API は次の場所にインストールされている RAID Manager を使用します。

Linux の場合：/HORCM ディレクトリ

- RAID Manager のコマンドを手動で実行したり、ユーザスクリプトから実行したりする場合、または RAID Manager を使用するプログラムを使用する場合は、REST API サーバがインストールされている管理サーバとは別のマシンで動作させることを推奨します。REST API サーバと同じマシンで動作させるときは、REST API で使用するユーザアカウントと RAID Manager で使用するユーザアカウントは、それぞれ専用のアカウントを使用してください。REST API と RAID Manager で同じユーザアカウントを使用していると、コマンド実行時にアカウントが不当にログアウトされ、コマンドの実行に失敗することがあります。

RAID Manager のインストールおよびアンインストールについては、マニュアル『RAID Manager インストール・設定ガイド』を参照してください。

1.5　REST API のインストール

管理サーバに REST API サーバをインストールします。

インストール時に次の情報を指定するため、事前に確認してください。

- インストール先のパス

  新規インストールの場合に指定します。

- データベースのバックアップ先のパス

  アップグレードインストールまたは上書きインストール中にバックアップを取得する場合に指定します。

1.5.1　REST API のインストール先の条件

REST API のインストール先と、インストール先に指定できるパスの条件について説明します。

REST API のデフォルトのインストール先は次のとおりです。

Linux の場合：

```
/opt/NEC/ConfManager
```

REST API のインストール先をデフォルト以外に変更する場合は、パス長や文字種などの条件を満たすインストール先を用意してください。REST API は、インストール中に指定したインストール先パスの下の ConfManager ディレクトリにインストールされます。

REST API のインストール先に指定できる絶対パスの条件を次に示します。

条件	説明
絶対パスの長さ	64 バイト以内
指定できる文字	**Linux の場合：** A〜Z a〜z 0〜9 _ /

条件	説明
そのほかの条件	**Linux の場合：** • ディレクトリパスの最後にパスの区切り文字（/）を指定しないでください。 • ドライブやネットワークファイルシステムをマウントしたディレクトリーを指定しないでください。

1.5.2　root ユーザで REST API をインストールする（Linux の場合）

インストールメディアを使用して、REST API をインストールします。

メモ

REST API のインストール時は、/tmp および/var/tmp ディレクトリ下のプログラムの実行を制限する noexec オプション設定は実施しないでください。

noexec オプション設定状況は、mount コマンドで確認できます。

REST API のインストールが完了したら、ディレクトリ下のプログラムの実行を制限する設定を実施しても REST API の動作には影響ありません。

前提条件

- root ユーザで管理サーバにログインしていること
- COLUMNS 環境変数が設定されていないこと

  COLUMNS 環境変数が設定されている状態でアップグレードインストールまたは上書きインストールを実行すると、インストールが正常に終了しない可能性があります。
- REST API に同梱されている RAID Manager を使用しているプログラムの停止

  REST API に同梱されている RAID Manager がインストールされた環境では、RAID Manager のファイルが使用されていると、REST API のアップグレードインストール、上書きインストールができません。

操作手順

1. インストールメディアを挿入します。

   自動的にマウントされない場合は、手動でマウントしてください。

   ヒント

   DVD-ROM のマウントパスに指定できる文字は次のとおりです。

   A～Z a～z 0～9 _ /

2. インストーラ（install.sh）が格納されているディレクトリに移動します。

   インストーラは、＜*DVD-ROM のマウントディレクトリ*＞/CM/ConfManager ディレクトリに格納されています。

3. 次のコマンドを実行します。

```
# ./install.sh
```

4. 表示されるメッセージに従って、必要な情報を指定します。

   インストールが完了すると、次のメッセージが表示されます。

```
Configuration Manager REST API installation completed successfully.
```

5. （アップグレードインストールの場合）ストレージシステムの構成情報の更新状態を確認します。

   a. ストレージシステムの構成情報の更新状態を取得する API を実行して、取得した `status` 属性の値を確認します。

   b. アップグレードインストールによって REST API のデータベースが拡張された場合は、`status` 属性の値が `Failed` と表示されます。この場合、エラー情報を確認し、ストレージシステムの構成情報を更新する API を実行します。

関連リンク

ストレージシステムの構成情報の更新状態を取得する（80 ページ）
ストレージシステムの構成情報を更新する（82 ページ）

1.5.3　Amazon Corretto 17 をアップグレードする

インストールされた Amazon Corretto 17 に脆弱性が指摘されている場合、脆弱性に対応された最新バージョンの Amazon Corretto 17 を適用できます。

REST API のアップグレードインストールまたは上書きインストールを実行後は、同梱された Amazon Corretto に変更されます。その場合、同じ手順を実行することで再度最新バージョンを適用できます。

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. Amazon Corretto を任意のフォルダに展開します。

   Linux 版の場合は、展開したファイルに対して実行権限があることを確認してください。

2. REST API のサービスを停止します。
3. REST API のシンボリックリンクの参照先を変更します。

Linux の場合：

```
rm -rf "＜REST API のインストール先＞/base/jdk"
```

```
ln -s "＜Amazon Corretto のパス＞" "＜REST API のインストール先＞/base/jdk"
```

4. REST API のサービスを起動します。

1.5.4　ウィルス検出プログラムおよびプロセス監視ソフトウェアを使用する場合に必要な設定

ウィルス検出プログラムで REST API が使用するファイルにアクセスすると、I/O 遅延やファイル排他などによって障害が発生することがあります。また、プロセス監視ソフトウェアが REST API プロセスを強制終了した場合、REST API は正常に動作しません。

これらを防止するため、REST API の稼働中は、ウィルス検出プログラムのスキャン対象およびプロセス監視ソフトウェアの監視対象から、次のディレクトリーを除外してください。

Linux の場合：

```
＜REST API のインストール先＞/ConfManager
```

```
/HORCM
```

REST API をクラスター環境で運用している場合は、共有ディスク上に作成した REST API 用の共有フォルダーもウィルス検出プログラムのスキャン対象およびプロセス監視ソフトウェアの監視対象から除外してください。

1.6　REST API で使用するポート番号の設定

REST API で使用するポートおよび、ポート番号の設定変更について説明します。

- REST API で使用するポート

  REST API クライアント、管理サーバおよびストレージシステムの通信に使用するポートを説明します。

メモ

- REST API が使用するポート番号と、管理サーバで動作するほかのプログラムが使用するポート番号が競合しないことを確認してください。競合する場合は、どちらかのポート番号を変更してください。
- REST API では、必要な通信を確立するためにファイアウォールの設定を変更しています。

  Linux 環境において、`iptables` および `firewalld` を使用している場合は、REST API により自動で変更されますが、`nftables` を使用している場合は、手動でファイアウォールの設定を変更する必要があります。

- 通信元のマシンと通信先のマシンの間にファイアウォールが設置されている場合は、通信元のポートから通信先のポートに通信できるようにファイアウォールの設定を変更してください。

  Linux の場合、以下のポートが開放されている必要があります。

  - 23450
  - 23451
  - 23452（ループバックの接続だけを許可するように設定してください。）
  - 23453（ループバックの接続だけを許可するように設定してください。）
  - 23454
  - 23455（ループバックの接続だけを許可するように設定してください。）
  - 23459（ループバックの接続だけを許可するように設定してください。）

関連リンク

REST API で使用するポート（9 ページ）

1.6.1　REST API で使用するポート

REST API では、デフォルトで次のポート番号を使用します。

通信元		通信先		説明
マシン	ポート番号	マシン	ポート番号	
REST API クライアント	any/tcp	管理サーバ	23450/tcp	REST API クライアントから REST API サーバに HTTP 通信をするときに使用されます。 通信先のポート番号は変更できます。
			23451/tcp	REST API クライアントから REST API サーバに HTTPS 通信をするときに使用されます。 通信先のポート番号は変更できます。
管理サーバ	any/tcp	管理サーバ（通信元と同じ）	23452/tcp 23453/tcp	REST API サーバの内部通信に使用されます。 通信先のポート番号は変更できます。
			23455/tcp	Linux の場合に、ストレージシステムの構成変更の通知を利用するとき、REST API サーバの内部通信に使用されます。 通信先のポート番号は変更できます。
			23459/tcp	ストレージシステムの構成変更の通知を利用するとき、REST API サーバの内部通信に使用されます。
	any/tcp	ストレージシステム（GUM）	443/tcp	REST API サーバとストレージシステム間の通信で使用されます。

通信元		通信先		説明
マシン	ポート番号	マシン	ポート番号	
ストレージシステム（GUM）	any/tcp	管理サーバ	23454/tcp	REST API サーバがストレージシステムの構成変更の通知を受信するときに使用されます。 通信先のポート番号は変更できます。

関連リンク

HTTPS 通信に使用するポート番号を変更する（10 ページ）
HTTP 通信に使用するポート番号を変更する（11 ページ）
HTTP 通信を無効化する（11 ページ）
HTTP 通信を有効化する（12 ページ）
HTTP 通信および HTTPS 通信に使用するポート番号の設定を初期状態に戻す（13 ページ）
REST API サーバが使用するポート番号を変更する（14 ページ）
ストレージシステムの構成変更の通知を受信するポート番号を変更する（15 ページ）
ストレージシステムの構成変更の通知とは（94 ページ）

1.6.2　HTTPS 通信に使用するポート番号を変更する

REST API クライアントと REST API サーバ間の HTTPS 通信に使用するポート番号を変更します。

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. REST API のサービスを停止します。
2. 次のファイルをテキストエディタで開きます。

   Linux の場合：

   ＜REST API のインストール先＞`/oss/apache/conf/userextra/user-httpd-ssl.conf`

3. `Listen` および `VirtualHost` に指定されているポート番号を変更します。

   指定できる値は、`1`～`65535` です。

   ```
   Listen ＜変更後のポート番号＞
   <VirtualHost _default_:＜変更後のポート番号＞>
   ```

4. ファイルを保存します。

5. REST API のサービスを起動します。

関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）

1.6.3　HTTP 通信に使用するポート番号を変更する

REST API クライアントと REST API サーバ間の HTTP 通信に使用するポート番号を変更します。

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. REST API のサービスを停止します。
2. 次のファイルをテキストエディタで開きます。

   Linux の場合：

   ＜*REST API のインストール先*＞/oss/apache/conf/userextra/user-httpd-port.conf

3. Listen に指定されているポート番号を変更します。

   指定できる値は、1～65535 です。

   ```
   Listen ＜変更後のポート番号＞
   ```

4. ファイルを保存します。
5. REST API のサービスを起動します。

関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）

1.6.4　HTTP 通信を無効化する

REST API クライアントと REST API サーバ間の通信に HTTPS だけを使用する場合は、HTTP 通信を無効化できます。

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. REST API のサービスを停止します。
2. 次のファイルをテキストエディタで開きます。

   Linux の場合：

   ＜*REST API のインストール先*＞/oss/apache/conf/userextra/user-httpd-port.conf

3. Listen 行の先頭に「#」を入力してコメント行にします。

   ```
   # Listen ＜ポート番号＞
   ```

4. ファイルを保存します。
5. REST API のサービスを起動します。

関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）

1.6.5　HTTP 通信を有効化する

REST API クライアントと REST API サーバ間の通信に HTTP を使用する場合は、HTTP 通信を有効化します。

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. REST API のサービスを停止します。
2. 次のファイルをテキストエディタで開きます。

Linux の場合：

＜*REST API のインストール先*＞/oss/apache/conf/userextra/user-httpd-port.conf

3. Listen 行の先頭の「#」を削除します。

```
Listen ＜ポート番号＞
```

4. ファイルを保存します。
5. REST API のサービスを起動します。

関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）

1.6.6　HTTP 通信および HTTPS 通信に使用するポート番号の設定を初期状態に戻す

HTTP 通信および HTTPS 通信に使用するポート番号の設定を初期状態に戻す方法について説明します。

ポート番号の設定に使用するファイルを誤って編集したり、削除したりした場合に、次の手順で初期状態に戻します。

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. REST API のサービスを停止します。
2. 現在の設定ファイルの内容を退避する場合は、次の場所にあるファイルを別の場所にコピーします。

Linux の場合：

＜*REST API のインストール先*＞/oss/apache/conf/userextra

3. 次の場所にある初期状態の設定ファイルを手順 2 の場所にコピーします。

Linux の場合：

＜*REST API のインストール先*＞/oss/apache/conf/userdefault

4. 必要に応じてポート番号の設定をやり直します。
5. REST API のサービスを起動します。

関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）

1.6.7　REST API サーバが使用するポート番号を変更する

REST API サーバが内部での通信に使用するポート番号の変更方法について説明します。

管理サーバに次のポート番号を使用するほかのプログラムがインストールされている場合、REST API サーバの設定を変更してポート番号が競合しないようにします。

- 23452
- 23453
- 23455（Linux の場合）

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. REST API のサービスを停止します。
2. ポート番号 23452 の設定を変更する場合は、次の手順で変更します。
   a. 次のファイルをテキストエディタで開きます。

      Linux の場合：

      ＜REST API のインストール先＞`/data/usercnf/user-api-port.ini`

   b. `-Djetty.port` に指定されているポート番号を変更してファイルを保存します。

      ```
      -Djetty.port=＜変更後のポート番号＞
      ```

   c. 次のファイルをテキストエディタで開きます。

      Linux の場合：

      ＜REST API のインストール先＞`/oss/apache/conf/userextra/user-proxy-path.conf`

   d. `ProxyPass` に指定されているポート番号を変更してファイルを保存します。

```
ProxyPass http://localhost:＜変更後のポート番号＞/restapi disablereuse=on nocanon
```

3. ポート番号 23453 の設定を変更する場合は、次の手順で変更します。
   a. 次のファイルをテキストエディタで開きます。

      Linux の場合：

      ＜*REST API のインストール先*＞/data/usercnf/user-api-port.ini

   b. -DSTOP.PORT に指定されているポート番号を変更してファイルを保存します。

```
-DSTOP.PORT=＜変更後のポート番号＞
```

4. ポート番号 23455 の設定を変更する場合は、setChangeNotificationPort コマンドを実行してポート番号を変更します。

メモ

現在ストレージシステムの構成変更の通知で使用されているポート番号は、setChangeNotificationPort コマンドの実行結果の Internal Port の値で確認できます。

＜*REST API のインストール先*＞/bin/setChangeNotificationPort.sh -get_port

＜*REST API のインストール先*＞/bin/setChangeNotificationPort.sh -set_internal_port ＜*変更後のポート番号*＞

オプション

set_internal_port

ストレージシステムの構成変更の通知で使用されるポート番号を指定します。指定できる値は 1～65535 です。

5. REST API のサービスを起動します。

関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）

1.6.8　ストレージシステムの構成変更の通知を受信するポート番号を変更する

ストレージシステムの構成変更の通知を受信するポート番号の変更方法について説明します。

メモ

現在ストレージシステムの構成変更の通知を受信するために使用しているポート番号は、setChangeNotificationPort コマンドの実行結果の SSL Port の値で確認できます。

Linux の場合：

＜*REST API のインストール先*＞/bin/setChangeNotificationPort.sh -get_port

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. REST API のサービスを停止します。
2. setChangeNotificationPort コマンドを実行してポート番号を変更します。

   Linux の場合：

   ＜*REST API のインストール先*＞/bin/setChangeNotificationPort.sh -set_ssl_port ＜*変更後のポート番号*＞

   オプション

   set_ssl_port

   ストレージシステムの構成変更の通知を受信するポート番号を指定します。指定できる値は 1～65535 です。
3. REST API のサービスを起動します。
4. 変更後のポート番号でストレージシステムの構成変更の通知を受信するために、ストレージシステムの構成変更の通知先を削除してから再度登録します。

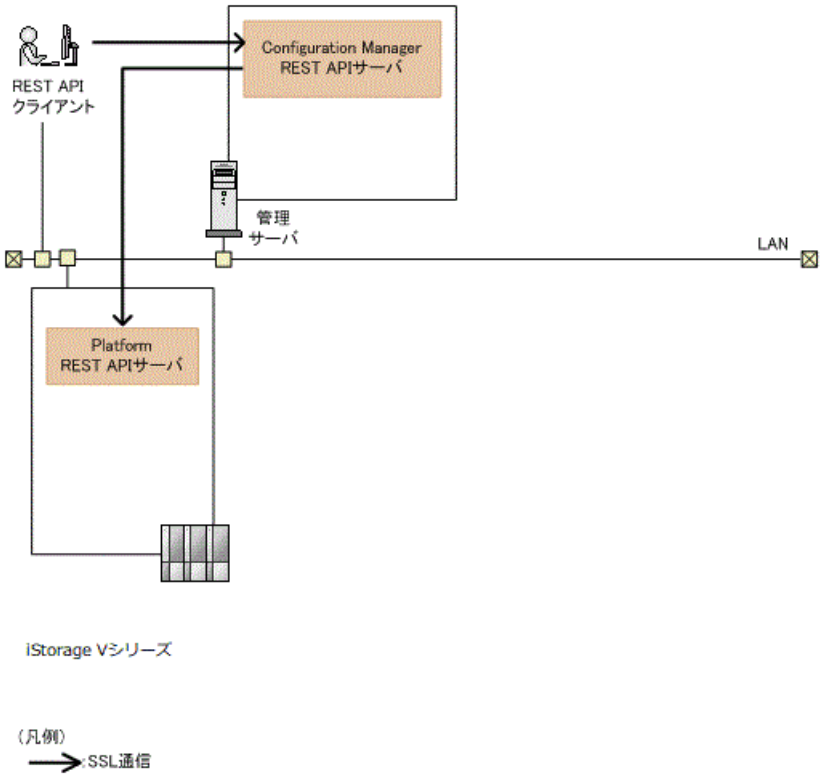
関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）
ストレージシステムの構成変更の通知先の一覧を取得する（96 ページ）
ストレージシステムの構成変更の通知先を登録する（100 ページ）
ストレージシステムの構成変更の通知先を削除する（102 ページ）

1.7　SSL 通信の設定

REST API の SSL 通信の設定について説明します。

REST API クライアントから Configuration Manager REST API サーバへの通信、および Configuration Manager REST API サーバからストレージシステムへの通信について説明します。



REST API クライアントから Configuration Manager REST API サーバへの通信

REST API クライアントから Configuration Manager REST API サーバへの SSL 通信には、Configuration Manager REST API サーバにインストールされているサーバ証明書が使用されます。デフォルトのサーバ証明書は自己署名証明書です。よりセキュリティを高めるためには、別の自己署名証明書または認証局の署名済みの証明書を使用するように変更してください。

秘密鍵とサーバ証明書を作成するには、証明書作成用のプログラム（OpenSSL など）が必要です。OpenSSL を使用する場合は、ホームページ（ http://www.openssl.org/ ）から入手して、インストールしてください。

ヒント

REST API クライアントから Configuration Manager REST API サーバへの SSL 通信にデフォルトのサーバ証明書を使用すると、クライアントプログラムによっては、通信がエラーになる場合があります。クライアントプログラムでエラーを回避するように作成することができます。

クライアントプログラムでエラーを回避するための方法は、プログラム言語によって異なります。

例えば Python では、Requests ライブラリを使用している場合、リクエスト発行時に `verify=False` を指定することでサーバ証明書の検証処理をスキップできます。

メモ

REST API クライアントと Configuration Manager REST API サーバ間の SSL 通信には、TLS バージョン 1.3 または TLS バージョン 1.2 が利用できます。

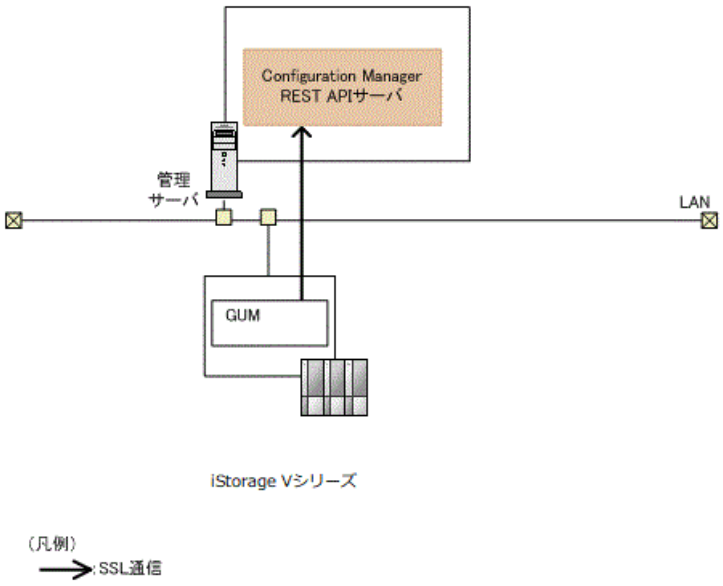
使用できる暗号方式（Cipher Suite）は次のとおりです。

- TLS_AES_128_GCM_SHA256 (0x13,0x01)
- TLS_AES_256_GCM_SHA384 (0x13,0x02)
- TLS_CHACHA20_POLY1305_SHA256 (0x13,0x03)
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xC0,0x30)
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xC0,0x2F)
- TLS_RSA_WITH_AES_256_GCM_SHA384(0x00,0x9D)
- TLS_RSA_WITH_AES_128_GCM_SHA256(0x00,0x9C)

Configuration Manager REST API サーバからストレージシステムへの通信

Configuration Manager REST API サーバと Platform REST API サーバ間で常に SSL 通信が利用されます。ストレージシステムを登録すると、自動的に SSL 通信が有効になります。詳細については、Configuration Manager REST API サーバと Platform REST API サーバ間の SSL 通信路の説明を参照してください。

ストレージシステムから Configuration Manager REST API サーバへの通信について説明します。



ストレージシステムから Configuration Manager REST API サーバへの通信

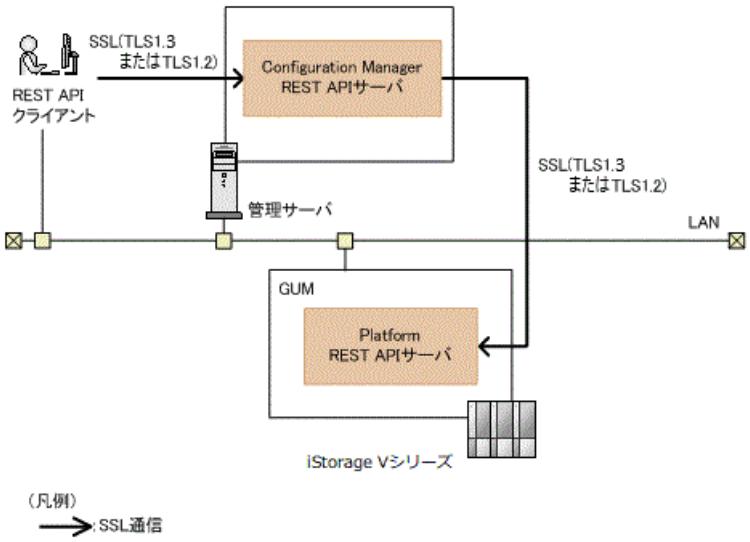
Configuration Manager REST API サーバがストレージシステムの構成変更の通知を受信するときに常に SSL 通信が利用されます。ストレージシステムから Configuration Manager REST API サーバへの SSL 通信には、Configuration Manager REST API サーバにインストールされているサーバ証明書が使用されます。デフォルトのサーバ証明書は自己署名証明書です。よりセキュリティを高めるために、認証局の署名済みの証明書を使用するように変更することもできます。

関連リンク

Configuration Manager REST API サーバと Platform REST API サーバ間の SSL 通信路（19 ページ）
REST API クライアントと REST API サーバ間で SSL 通信するよう設定する（自己署名証明書を使用する場合）（20 ページ）
REST API クライアントと REST API サーバ間で SSL 通信するよう設定する（認証局が発行したサーバ証明書を使用する場合）（23 ページ）

1.7.1　Configuration Manager REST API サーバと Platform REST API サーバ間の SSL 通信路

Configuration Manager REST API サーバから Platform REST API サーバへの SSL 通信路について説明します。



Configuration Manager REST API サーバは、REST API クライアントから受け付けたリクエストを、Platform REST API サーバに送信します。このとき、Configuration Manager REST API サーバと Platform REST API サーバ間で常に SSL 通信が利用されます。

ストレージシステムを登録すると、自動的に SSL 通信が有効になります。

関連リンク

Configuration Manager REST API サーバと Platform REST API サーバ間で SSL 通信するよう設定する（26 ページ）

1.7.2　REST API クライアントと REST API サーバ間で SSL 通信するよう設定する（自己署名証明書を使用する場合）

REST API クライアントと REST API サーバ間で、自己署名証明書を使用して SSL 通信するよう設定する手順を説明します。

⚠ 注意

サーバ証明書および秘密鍵は、REST API サーバにログインしたすべてのユーザがアクセスできます。悪意のあるユーザがログインできないよう、ユーザアカウントの管理には十分に注意してください。

ヒント

Linux の場合、OpenSSL のファイルが手順中のパスとは異なる場所に格納されていることがあります。次の手順では、`openssl` ファイルおよび `openssl.cfg` ファイルが＜*OpenSSL のインストール先*＞`/bin` にある環境を想定しています。

次のコマンドを実行すると、`openssl.cfg` の格納先を確認できます。

＜*OpenSSL のインストール先*＞`/bin/openssl version -a | grep OPENSSLDIR`

操作手順

1. 次のコマンドを実行して、秘密鍵ファイルを作成します。

Linux の場合：

＜*OpenSSL のインストール先*＞`/bin/openssl genrsa -out` ＜*秘密鍵ファイル*＞ `2048`

オプション

`out`

作成する秘密鍵の出力先パスを指定します。出力先パスに同じ名称のファイルがある場合、ファイルが上書きされます。

ヒント

秘密鍵ファイルは、次の場所に `createdServer.key` というファイル名で保存しておくことをお勧めします。

Linux の場合：

＜*REST API のインストール先*＞/oss/apache/conf/ssl.key/createdServer.key

このディレクトリには、REST API が配置した server.key ファイルが格納されています。server.key ファイルを上書きしないでください。

---

2. 次のコマンドを実行して、証明書に書かれる情報を対話形式で入力し、証明書発行要求ファイルを作成します。

Linux の場合：

＜*OpenSSL のインストール先*＞/bin/openssl req -config ＜*OpenSSL のインストール先*＞/bin/openssl.cfg -sha256 -new -key ＜*秘密鍵ファイル*＞ -out ＜*証明書発行要求ファイル*＞

オプション

key

手順 1 で作成した秘密鍵ファイル名を指定します。

out

作成する証明書発行要求の出力先パスを指定します。出力先パスに同じ名称のファイルがある場合、ファイルが上書きされます。

入力する情報を次に示します。

項目	説明
Country Name	（必須）2 文字の国コードを指定します。
State or Province Name	（必須）都道府県名を指定します。
Locality Name	（任意）市区町村名または地域名を指定します。
Organization Name	（必須）組織名を指定します。
Organization Unit Name	（任意）組織の構成単位名を指定します。
Common Name	（必須）REST API サーバの IP アドレスまたはホスト名を入力します。
Email Address	（任意）メールアドレスを指定します。
A challenge password	（任意）証明書を破棄するときに必要になるパスワードを指定します。
An optional company name	（任意）別の組織名を指定します。

3. 次のコマンドを実行して、自己署名証明書ファイルを作成します。

Linux の場合：

＜*OpenSSL のインストール先*＞/bin/openssl x509 -req -sha256 -days ＜*有効日数*＞ -signkey ＜*秘密鍵ファイル*＞ -in ＜*証明書発行要求ファイル*＞ -out ＜*自己署名証明書ファイル*＞

オプション

days

自己署名証明書の有効期間を日数で指定します。

signkey

手順 1 で作成した秘密鍵ファイル名を指定します。

in

手順 2 で作成した証明書発行要求のファイル名を指定します。

out

作成する自己署名証明書の出力先パスを指定します。出力先パスに同じ名称のファイルがある場合、ファイルが上書きされます。

ヒント

自己署名証明書ファイルは、次の場所に createdServer.crt というファイル名で保存しておくことをお勧めします。

Linux の場合：

＜*REST API のインストール先*＞/oss/apache/conf/ssl.crt/createdServer.crt

このディレクトリには、REST API が配置した server.crt ファイルが格納されています。server.crt ファイルを上書きしないでください。

4. REST API のサービスを停止します。
5. user-httpsd-certificate.conf ファイルを編集して、秘密鍵ファイルおよびサーバ証明書ファイル（自己署名証明書ファイル）を設定します。

user-httpsd-certificate.conf ファイルの格納場所

Linux の場合：

＜*REST API のインストール先*＞/oss/apache/conf/userextra/user-httpsd-certificate.conf

user-httpsd-certificate.conf ファイルの SSLCertificateKeyFile および SSLCertificateFile に次の内容を指定します。

SSLCertificateKeyFile

手順 1 で作成した秘密鍵のファイル名を絶対パスで指定します。シンボリックリンクやジャンクションを指定しないでください。

SSLCertificateFile

手順 3 で作成したサーバ証明書（自己署名証明書）のファイル名を絶対パスで指定します。シンボリックリンクやジャンクションを指定しないでください。

user-httpsd-certificate.conf ファイルの指定例

Linux の場合：

```
SSLCertificateKeyFile "/opt/NEC/ConfManager/oss/apache/conf/ssl.key
/createdServer.key"
```

```
SSLCertificateFile "/opt/NEC/ConfManager/oss/apache/conf/ssl.crt/cr
eatedServer.crt"
```

6. REST API のサービスを起動します。

関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）

1.7.3　REST API クライアントと REST API サーバ間で SSL 通信するよう設定する（認証局が発行したサーバ証明書を使用する場合）

REST API クライアントと REST API サーバ間で、認証局発行のサーバ証明書を使用して SSL 通信するよう設定する手順を説明します。

⚠ 注意

サーバ証明書および秘密鍵は、REST API サーバにログインしたすべてのユーザがアクセスできます。悪意のあるユーザがログインできないよう、ユーザアカウントの管理には十分に注意してください。

ヒント

Linux の場合、OpenSSL のファイルが手順中のパスとは異なる場所に格納されていることがあります。次の手順では、openssl ファイルおよび openssl.cfg ファイルが＜*OpenSSL のインストール先*＞/bin にある環境を想定しています。

次のコマンドを実行すると、openssl.cfg の格納先を確認できます。

＜*OpenSSL のインストール先*＞/bin/openssl version -a | grep OPENSSLDIR

操作手順

1. 次のコマンドを実行して、秘密鍵ファイルを作成します。

Linux の場合：

＜*OpenSSL のインストール先*＞/bin/openssl genrsa -out ＜*秘密鍵ファイル*＞ 2048

オプション

out

作成する秘密鍵の出力先パスを指定します。出力先パスに同じ名称のファイルがある場合、ファイルが上書きされます。

ヒント

秘密鍵ファイルは、次の場所に createdServer.key というファイル名で保存しておくことをお勧めします。

Linux の場合：

＜*REST API のインストール先*＞/oss/apache/conf/ssl.key/createdServer.key

このディレクトリには、REST API が配置した server.key ファイルが格納されています。server.key ファイルを上書きしないでください。

2. 次のコマンドを実行して、証明書に書かれる情報を対話形式で入力し、証明書発行要求ファイルを作成します。

Linux の場合：

＜*OpenSSL のインストール先*＞/bin/openssl req -config ＜*OpenSSL のインストール先*＞/bin/openssl.cfg -sha256 -new -key ＜*秘密鍵ファイル*＞ -out ＜*証明書発行要求ファイル*＞

オプション

key

手順 1 で作成した秘密鍵ファイル名を指定します。

out

作成する証明書発行要求の出力先パスを指定します。出力先パスに同じ名称のファイルがある場合、ファイルが上書きされます。

入力する情報を次に示します。

項目	説明
Country Name	（必須）2 文字の国コードを指定します。
State or Province Name	（必須）都道府県名を指定します。
Locality Name	（任意）市区町村名または地域名を指定します。
Organization Name	（必須）組織名を指定します。

項目	説明
Organization Unit Name	（任意）組織の構成単位名を指定します。
Common Name	（必須）REST API サーバの IP アドレスまたはホスト名を入力します。
Email Address	（任意）メールアドレスを指定します。
A challenge password	（任意）証明書を破棄するときに必要になるパスワードを指定します。
An optional company name	（任意）別の組織名を指定します。

3. 手順 2 で作成した証明書発行要求を認証局に送付します。

   認証局へのサーバ証明書の申請は、通常、オンラインでできます。作成した REST API サーバの証明書発行要求を任意の認証局に送信し、電子署名を受けます。

   X.509 PEM 形式のサーバ証明書を発行してもらう必要があります。申請方法については、使用する認証局の Web サイトなどで確認してください。また、証明書の署名アルゴリズムに認証局が対応していることを確認してください。

   メモ

   - 認証局からの返答は保存しておいてください。
   - 認証局が発行する証明書には有効期限があります。期限が切れる前に再発行してもらう必要があります。

     証明書の有効期限を確認するには、次のコマンドを実行してください。

     Linux の場合：

     ＜*OpenSSL のインストール先*＞/bin/openssl x509 -in ＜*サーバ証明書ファイル*＞ -text -noout

   ヒント

   入手したサーバ証明書は、次の場所に createdServer.crt というファイル名で保存しておくことをお勧めします。

   Linux の場合：

   ＜*REST API のインストール先*＞/oss/apache/conf/ssl.crt/createdServer.crt

   このディレクトリには、REST API が配置した server.crt ファイルが格納されています。server.crt ファイルを上書きしないでください。

4. REST API のサービスを停止します。
5. user-httpsd-certificate.conf ファイルを編集して、秘密鍵ファイルおよびサーバ証明書ファイルを設定します。

   user-httpsd-certificate.conf ファイルの格納場所

Linux の場合：

＜*REST API のインストール先*＞/oss/apache/conf/userextra/user-httpsd-certificate.conf

user-httpsd-certificate.conf ファイルの SSLCertificateKeyFile および SSLCertificateFile に次の内容を指定します。

SSLCertificateKeyFile

手順 1 で作成した秘密鍵のファイル名を絶対パスで指定します。シンボリックリンクやジャンクションを指定しないでください。

SSLCertificateFile

手順 3 で認証局から発行されたサーバ証明書のファイル名を絶対パスで指定します。シンボリックリンクやジャンクションを指定しないでください。

user-httpsd-certificate.conf ファイルの指定例

Linux の場合：

```
SSLCertificateKeyFile "/opt/NEC/ConfManager/oss/apache/conf/ssl.key
/createdServer.key"
```

```
SSLCertificateFile "/opt/NEC/ConfManager/oss/apache/conf/ssl.crt/cr
eatedServer.crt"
```

6. REST API のサービスを起動します。

関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）

1.7.4　Configuration Manager REST API サーバと Platform REST API サーバ間で SSL 通信するよう設定する

Configuration Manager REST API サーバと Platform REST API サーバ間で常に SSL 通信が利用されます。

前提条件

- 次のユーザで管理サーバにログインしていること
  - root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. ストレージシステムを登録する API で、ストレージシステムを登録します。

   ストレージシステムを登録すると、Configuration Manager REST API サーバと Platform REST API サーバ間の SSL 通信が自動的に有効になります。

関連リンク

ストレージシステムを登録する（58 ページ）

1.7.5　Configuration Manager REST API サーバとストレージシステム間の SSL 通信に使用する証明書を変更する（構成変更の通知を受信する場合）

Configuration Manager REST API サーバがストレージシステムの構成変更の通知を受信するときに、認証局発行のサーバ証明書を利用して SSL 通信するように設定する手順を説明します。

ストレージシステムの情報検索では、REST API サーバのデータベースに保持しているストレージシステムの構成情報を取得します。ストレージシステムの構成変更の通知を設定しておくと、REST API サーバのデータベースに保持している構成情報を最新の状態に保つことができます。

構成変更の通知では、デフォルトでは自己署名のサーバ証明書が使用されています。セキュリティを向上させるために認証局が発行したサーバ証明書に変更ができます。

⚠ 注意

サーバ証明書および秘密鍵は、REST API サーバにログインしたすべてのユーザがアクセスできます。悪意のあるユーザがログインできないよう、ユーザアカウントの管理には十分に注意してください。

ヒント

Linux の場合、OpenSSL のファイルが手順中のパスとは異なる場所に格納されていることがあります。次の手順では、`openssl` ファイルおよび `openssl.cfg` ファイルが＜*OpenSSL のインストール先*＞`/bin` にある環境を想定しています。

次のコマンドを実行すると、`openssl.cfg` の格納先を確認できます。

＜*OpenSSL のインストール先*＞`/bin/openssl version -a | grep OPENSSLDIR`

前提条件

- 前提条件、操作手順で作成または取得するファイルは、一時保管用ディレクトリを作成し保存してください。手順 5 で使います。

- サーバ証明書を発行する認証局の証明書を入手し、ファイル名を cacert.pem に変更して、一次保管用ディレクトリに保存してください。

メモ

サーバ証明書を発行した認証局が中間認証局の場合は、複数の証明書をテキストエディタで連結させることで、1 つのファイルに複数の証明書を混在させます。この証明書ファイルを cacert.pem として一時保管用ディレクトリに保存してください。

操作手順

1. 次のコマンドを実行して、秘密鍵ファイルを作成します。

Linux の場合：

＜*OpenSSL のインストール先*＞/bin/openssl genrsa -out ＜*秘密鍵ファイル*＞ 2048

オプション

out

作成する秘密鍵の出力先パスを指定します。秘密鍵のファイル名は key.pem としてください。出力先パスに同じ名称のファイルがある場合、ファイルが上書きされます。

2. 次のコマンドを実行して、証明書に書かれる情報を対話形式で入力し、証明書発行要求ファイルを作成します。

Linux の場合：

＜*OpenSSL のインストール先*＞/bin/openssl req -config ＜*OpenSSL のインストール先*＞/bin/openssl.cfg -sha256 -new -key ＜*秘密鍵ファイル*＞ -out ＜*証明書発行要求ファイル*＞

オプション

key

手順 1 で作成した秘密鍵ファイル名を指定します。

out

作成する証明書発行要求の出力先パスを指定します。出力先パスに同じ名称のファイルがある場合、ファイルが上書きされます。

入力する情報を次に示します。

項目	説明
Country Name	（必須）2 文字の国コードを指定します。

項目	説明
State or Province Name	（必須）都道府県名を指定します。
Locality Name	（任意）市区町村名または地域名を指定します。
Organization Name	（必須）組織名を指定します。
Organization Unit Name	（任意）組織の構成単位名を指定します。
Common Name	（必須）REST API サーバの IP アドレスまたはホスト名を入力します。
Email Address	（任意）メールアドレスを指定します。
A challenge password	（任意）証明書を破棄するときに必要になるパスワードを指定します。
An optional company name	（任意）別の組織名を指定します。

3. 手順 2 で作成した証明書発行要求を認証局に送付します。

   認証局へのサーバ証明書の申請は、通常、オンラインでできます。作成した REST API サーバの証明書発行要求を任意の認証局に送信し、電子署名を受けます。

   X.509 PEM 形式のサーバ証明書を発行してもらう必要があります。申請方法については、使用する認証局の Web サイトなどで確認してください。また、証明書の署名アルゴリズムに認証局が対応していることを確認してください。

   認証局の電子署名を受けたサーバ証明書は `cert.pem` というファイル名で一時保管用ディレクトリに保存してください。

   メモ

   - 認証局からの返答は保存しておいてください。
   - 認証局が発行する証明書には有効期限があります。期限が切れる前に再発行してもらう必要があります。

     証明書の有効期限を確認するには、次のコマンドを実行してください。

     Linux の場合：

     ＜OpenSSL のインストール先＞`/bin/openssl x509 -in` *＜サーバ証明書ファイル＞* `-text -noout`

4. REST API のサービスを停止します。

5. 一時保管用ディレクトリに保存されている認証局の証明書ファイル（`cacert.pem`）、秘密鍵ファイル（`key.pem`）、およびサーバ証明書ファイル（`cert.pem`）を次のディレクトリに配置します。

   Linux の場合：

   ＜REST API のインストール先＞`/oss/rabbitmq/etc/rabbitmq/ssl/extra`

6. `advanced.config` ファイルの内容を書き換えます。

   `advanced.config` ファイルは次のディレクトリに配置されています。必要に応じてバックアップをとってください。

Linux の場合：

＜*REST API のインストール先*＞/oss/rabbitmq/etc/rabbitmq

advanced.config ファイルの cacertfile、certfile および keyfile の行を書き換えます。

- 認証局の証明書ファイルの配置先パスは、デフォルトでは次のように定義されています。

```
cacertfile, "＜REST API のインストール先＞/oss/rabbitmq/etc/rabbitmq/ssl/default/cacert.pem"
```

配置先パスの default を extra に書き換えます。default 以外は変更しないでください。

- 秘密鍵ファイルの配置先パスは、デフォルトでは次のように定義されています。

```
certfile, "＜REST API のインストール先＞/oss/rabbitmq/etc/rabbitmq/ssl/default/key.pem"
```

配置先パスの default を extra に書き換えます。default 以外は変更しないでください。

- サーバ証明書ファイルの配置先パスは、デフォルトでは次のように定義されています。

```
keyfile, "＜REST API のインストール先＞/oss/rabbitmq/etc/rabbitmq/ssl/default/cert.pem"
```

配置先パスの default を extra に書き換えます。default 以外は変更しないでください。

7. REST API のサービスを起動します。

関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）

1.8　REST API のサービスの起動と停止

REST API のサービスの起動、停止について説明します。

REST API のサービスには、ConfManagerWebServer、ConfManagerAPIServer、および ConfManagerMessageQueueServer があります。

1.8.1　REST API のサービスの起動

REST API のサービスの起動方法について説明します。

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. 次の操作を実行します。

   Linux の場合：

   次のコマンドを実行します。

   ＜*REST API のインストール先*＞/start.sh

1.8.2　REST API のサービスの停止

REST API のサービスの停止方法について説明します。

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. 次の操作を実行します。

   Linux の場合：

   次のコマンドを実行します。

   ＜*REST API のインストール先*＞/stop.sh

1.8.3　REST API のサービスの稼働状態の確認

REST API のサービスの稼働状態の確認方法について説明します。

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. 次の操作を実行します。

Linux の場合：

次のコマンドを実行します。

＜*REST API のインストール先*＞/status.sh

1.9　REST API のアンインストール

REST API の運用をやめる場合、REST API サーバをアンインストールします。

REST API のアンインストーラを使って、REST API を削除してください。

メモ

次の場合、REST API をアンインストールしても RAID Manager は削除されないで残ります。不要な場合は、RAID Manager をアンインストールしてください。

- 管理サーバに事前に RAID Manager がインストールされていた場合
- 管理サーバの OS が Linux で、REST API をインストールしたあとに RAID Manager をインストールした場合

  この場合、RAID Manager は/HORCM ディレクトリにインストールされた状態になります。

1.9.1　root ユーザで REST API をアンインストールする（Linux の場合）

uninstall.sh を実行して、REST API をアンインストールします。

メモ

REST API のアンインストール時は、/tmp および/var/tmp ディレクトリ下のプログラムの実行を制限する noexec オプション設定は実施しないでください。

noexec オプション設定状況は、mount コマンドで確認できます。

REST API のアンインストールが完了したら、必要に応じてディレクトリ下のプログラムの実行を制限する設定を実施してください。

前提条件

- COLUMNS 環境変数が設定されていないこと

  COLUMNS 環境変数が設定されている状態でアンインストールを実行すると、アンインストールが正常に終了しない可能性があります。

- REST API に同梱されている RAID Manager を使用しているプログラムの停止

  REST API に同梱されている RAID Manager がインストールされた環境では、RAID Manager のファイルが使用されていると、REST API をアンインストールできません。

操作手順

1. root ユーザで管理サーバにログインします。
2. 必要に応じて REST API のデータベースおよび環境設定ファイルをバックアップします。
3. ストレージシステムに登録された構成変更の通知先から、アンインストール対象の REST API サーバの情報を削除するため、ストレージシステムの情報を削除します。
4. ルートディレクトリに移動します。
5. 次のコマンドを実行します。

   ```
   <REST API のインストール先>/inst/uninstall.sh
   ```

6. 表示されたメッセージに従って操作します。

   アンインストールが完了すると、次のメッセージが表示されます。

   ```
   Configuration Manager REST API removal completed successfully.
   ```

関連リンク

ストレージシステムの情報を削除する（64 ページ）
REST API のデータベースおよび環境設定ファイルをバックアップする（89 ページ）

第 2 章
REST API の共通仕様

この章では、REST API でのリソースの指定方法、リクエストとレスポンスの形式および各オブジェクトについて説明します。

2.1　管理対象のリソースの指定

REST API では、管理対象のリソースを URL の形式で指定します。

REST API では、操作の種類ごとにドメインを分けています。URL の形式はドメインごとに異なります。REST API で使用するドメインと、指定する URL の形式を次に示します。

objects ドメイン

REST API の操作対象の個々のオブジェクトに対する操作を定義するドメインです。次に示す形式で URL を指定します。

```
＜プロトコル＞://＜ホスト名＞:＜ポート番号＞/ConfigurationManager/＜バージョン＞/objects/storages/＜ストレージデバイス ID＞
```

views ドメイン

REST API サーバに保持しているストレージシステムの構成情報に対する操作を定義するドメインです。次に示す形式で URL を指定します。

```
＜プロトコル＞://＜ホスト名＞:＜ポート番号＞/ConfigurationManager/＜バージョン＞/views
```

services ドメイン

REST API サーバで提供するサービスを定義するドメインです。サービスとは、複数のオブジェクトに対する一括操作や、REST API サーバの運用や構成変更についての操作を指します。次に示す形式で URL を指定します。

```
＜プロトコル＞://＜ホスト名＞:＜ポート番号＞/ConfigurationManager/＜バージョン＞/＜ストレージデバイス ID＞/services
```

configuration ドメイン

REST API サーバに関する設定を定義するドメインです。次に示す形式で URL を指定します。

```
＜プロトコル＞://＜ホスト名＞:＜ポート番号＞/ConfigurationManager/configuration
```

このマニュアルでは、「*＜プロトコル＞*://*＜ホスト名＞*:*＜ポート番号＞*/ConfigurationManager」をベース URL と表記します。

- プロトコルには、https または http を指定します。セキュリティのため、https を使用することをお勧めします。
- ホスト名には、管理サーバの IP アドレスまたは名前解決のできるホスト名を指定します。
- ポート番号には、REST API サーバとの通信に使用するポート番号を指定します。デフォルトのポート番号は、SSL 通信の場合は 23451、非 SSL 通信の場合は 23450 です。
- バージョンには REST API のバージョンを指定します。現在指定できる値は v1 です。
- ストレージデバイス ID には、「操作対象のストレージシステムの機種ごとの固定値＋ 6 けたのシリアル番号の合計 12 けた」を指定します。

  ストレージシステムの機種ごとの固定値を次に示します。

ストレージシステム	固定値
iStorage V100、iStorage V300	934000

2.2　オブジェクト ID の指定方法

オブジェクト ID は、リソースを一意に識別するための ID です。URL で特定のリソースを指定する場合に使用します。

オブジェクト ID の指定方法には、次の 2 つの方法があります。

- GET 操作を実行して、実行結果からオブジェクト ID を取得する（推奨）。
- 複数の属性値をコンマでつないだ文字列でオブジェクト ID を生成する。

複数の属性値をコンマでつないでオブジェクト ID を生成する場合、RFC3986 に従って REST API クライアントで属性値をエンコードする必要があります。各属性の値をエンコードしたあとで、属性値をコンマでつないだ文字列をオブジェクト ID として指定します。エンコードが必要な代表的な文字を次に示します。

エンコード前	エンコード後
! （感嘆符）	%21
# （番号記号）	%23
$ （ドル記号）	%24
% （パーセント）	%25
& （アンパサンド）	%26
' （アポストロフィ）	%27
(（始め丸括弧）	%28
) （終わり丸括弧）	%29
* （アスタリスク）	%2A

エンコード前	エンコード後
+ （正符号）	%2B
, （コンマ）	%2C
: （コロン）	%3A
; （セミコロン）	%3B
= （等号）	%3D
? （疑問符）	%3F
@ （単価記号）	%40
[（始め角括弧）	%5B
] （終わり角括弧）	%5D

重要

- GET 操作でオブジェクト ID を取得する場合、REST API サーバはエンコード済みの値を返します。GET 操作で取得したオブジェクト ID を別の操作のリクエストに使用する場合は、オブジェクト ID をデコードしないでそのまま使用してください。
- オブジェクトを新規に作成したり属性を変更したりする場合、上記の予約文字を含まないように指定することをお勧めします。

関連リンク

リクエストおよびレスポンスのフォーマット（45 ページ）

2.3　サポートする HTTP メソッド

HTTP では、リソースに対して実行できる操作をメソッドとして定義しています。

REST API では、次に示す HTTP メソッドをサポートしています。

HTTP メソッド	説明	処理方式
GET	オブジェクトの情報を取得する。またはオブジェクトのリストを取得する。	同期
POST	オブジェクトを新規に作成する。	非同期 ただし、次の API は同期処理で実行されます。 • ストレージシステムの登録 • セッションの生成
PUT	オブジェクトの属性や状態を変更する。	非同期 ただし、次の API は同期処理で実行されます。 • ストレージシステム情報の変更
DELETE	オブジェクトを削除する。	非同期 ただし、次の API は同期処理で実行されます。

HTTP メソッド	説明	処理方式
		• ストレージシステム情報の削除 • セッションの破棄

ヒント

HTTP メソッドが PUT の API は、次のメソッドでも実行できます。

- PATCH：オブジェクトの属性や状態を変更する操作
- POST：オブジェクトに対して特定のアクションを実行する操作

REST API の処理方式（同期処理と非同期処理）について説明します。

- 同期処理の場合、処理の実行結果がレスポンスとして返ります。
- 非同期処理の場合、リソースに対する操作はジョブとして登録され、処理を受け付けたことを表す HTTP ステータスコード（202）とともにジョブの情報がレスポンスとして返ります。登録されたジョブは、その後、非同期に実行されます。ジョブの登録に失敗した場合は HTTP ステータスコード（500）が返ります。

  ヒント

  - 非同期処理の操作の場合でも、リクエストヘッダで `Response-Job-Status` に `Completed` を指定すると、ジョブの実行が完了するまで待ってからレスポンスが返ります。

関連リンク

リクエストヘッダ（41 ページ）

2.4　ユーザ認証

ストレージシステムに対する操作を実行する場合、ユーザ認証が必要です。 すべての REST API を実行する際には、ユーザ認証のために Authorization ヘッダを指定します。

REST API では、セッションベースのユーザ認証を行います。 REST API クライアントが REST API サーバにアクセスして操作を開始する際には、必ず最初にセッションを生成します。セッション生成のリクエストでは、ストレージシステムにアクセスするためのユーザ ID とパスワードによる認証を行いますが、セッション生成後は、セッションの情報を Authorization ヘッダに指定し、セッションの情報に基づいて認証を行います。

メモ

- REST API のユーザ認証には、ストレージシステムに登録されたユーザまたはストレージシステムに接続された外部認証サーバと外部認可サーバで管理されているユーザを使用してください。

HA Command Suite 製品など、ほかの製品のユーザや、ほかの製品に接続された外部認証サーバと外部認可サーバで管理されているユーザは、REST API のユーザ認証には使用できません。

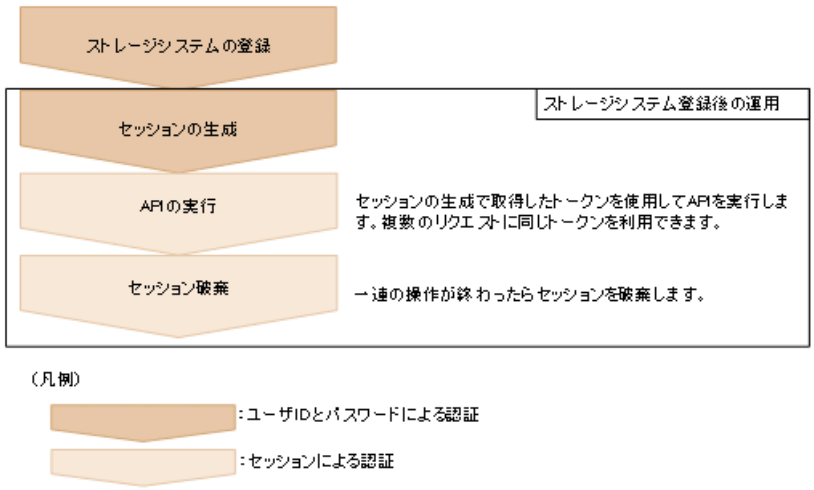
- ストレージシステムと HA Command Suite 製品など、ほかの製品が同じ外部認証サーバと外部認可サーバに接続している場合、REST API のユーザ認証には、ほかの製品で利用するユーザとは別のユーザを新規に作成してください。その場合、次の条件を満たすユーザとしてください。
  - 外部認証ユーザを作成する場合は、このユーザを HA Command Suite 製品に登録しないでください。
  - 外部認可ユーザを作成する場合は、この外部認可グループを HA Command Suite 製品に登録しないでください。

ユーザ認証の使い分けについて

REST API を使用するときは、下記のとおり認証方法を使い分けてください。

- ストレージシステムの登録：ユーザ ID とパスワードによる認証
- セッションの生成：ユーザ ID とパスワードによる認証
- 上記以外の操作：セッションによる認証

REST API の運用に合わせた認証の流れを次に示します。



ユーザ ID とパスワードによる認証

Authorization ヘッダに、次の形式で認証情報を指定します。

```
Authorization: Basic <認証情報>
```

認証情報

ユーザ ID とパスワードをコロン（:）でつないだ文字列を Base64 でエンコードした文字列を指定します。 ストレージシステムのリソースを操作できるユーザアカウントのユーザ ID とパスワードを使用してください。

REST API では、ユーザ ID とパスワードに次の文字を使用できます。

項目	文字数	使用できる文字
ユーザ ID	1～63 文字	次の文字が使用できます。 • 半角英数字 • 次の半角記号 ! # $ % & ' * + - . / = ? @ ^ _ ` { | } ~
パスワード	6 ～63 文字	次の文字が使用できます。 • 半角英数字 • スペースを除くキー入力可能な ASCII 記号 ! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [\ ] ^ _ ` { | } ~

メモ

パスワードポリシーが設定できるストレージシステムの場合、使用できる文字数および文字列は設定されているポリシーに従います。詳細は『システム管理者ガイド』またはご利用のストレージシステムのユーザガイドのパスワードポリシーに関する説明を参照してください。

ユーザ ID が `sample-user`、パスワードが `sample-password` の場合の Authorization ヘッダの例を次に示します。

```
Authorization: Basic c2FtcGxlLXVzZXI6c2FtcGxlLXBhc3N3b3Jk
```

セッションによる認証

Authorization ヘッダに、次の形式でセッションのトークンを指定します。

```
Authorization: Session <トークン>
```

トークン

トークンは、セッションを生成すると返却される認証情報です。この情報を基に、リクエストが認証済みユーザから発行されたかどうかを判定します。

Authorization ヘッダの指定例：

```
Authorization : Session 550e8400-e29b-41d4-a716-446655440000
```

2.5　セッション管理

REST API では、セッションを使用して、複数のリクエストを同一クライアントによる一連の操作として識別します。例えば、あるユーザが同じアカウントを使用して平行に 2 つのクライアントプログラムを実行したい場合は、それぞれ別のセッションを生成する必要があります。それぞれのプログラムは REST API サーバ上でセッションの情報に基づいて識別されます。

REST API クライアントが REST API サーバにアクセスしてストレージシステムの操作を開始する際には、必ず最初にセッションを生成します。セッションを生成すると、クライアントにはセッション ID とトークンが返却されます。以降の操作では、各リクエストの Authorization ヘッダに、認証情報としてトークンを指定します。REST API クライアントからの操作を終了するときは、セッションを削除して、サーバ上に不要なセッションが残らないようにしてください。

セッションの生成

REST API のセッションは、ユーザがセッション生成の API を実行することで生成されます。1 ユーザが複数のセッションを生成できます。使用できるセッションの上限数は、1 ストレージシステム当たり 64 セッションです。

セッションを生成すると、クライアントには次の情報がレスポンスとして返ります。

- セッション ID

  REST API サーバ上でセッションを識別するための ID です。セッションが有効かどうか確認したり、セッションを破棄したりするのに使用します。セッション ID は、セッションを生成したユーザのほか、Administrator ユーザグループ（ビルトイングループ）に属するユーザが参照できます。

- トークン

  リクエストの発行元が特定のユーザであることを識別するための情報です。同一セッションのリクエストであるかどうかを判定するのに使用します。トークンは、セッションを生成したユーザだけが参照できます。

セッションを使用した API の実行

セッションを使用して API を実行するには、リクエストの Authorization ヘッダに認証情報としてトークンを指定します。同じトークンを指定したリクエストは同一セッションによる操作として扱われます。トークンを指定した Authorization ヘッダの指定例を次に示します。

```
Authorization : Session 550e8400-e29b-41d4-a716-446655440000
```

セッションが使用されずに一定時間が経過すると、セッションは自動的に破棄されます（セッションタイムアウト）。セッションタイムアウトまでの経過時間は、そのセッションが生成されてから、または、セッションを指定したリクエストの実行結果が返却されてから経

過した時間です。同期処理中の待ち時間や、非同期処理の API のレスポンス待ち時間は、経過時間にカウントされません。経過時間中にそのセッションを使用したリクエストが発行されると、セッションタイムアウトまでの経過時間はリセットされます。セッションタイムアウトまでの時間はデフォルトで 300 秒（5 分）ですが、セッション生成時に時間を指定することもできます。

継続中の操作のセッションがセッションタイムアウトによって破棄されないようにするには、対象セッションを使用したリクエストを定期的に発行してください。

ヒント

セッションの使用中に、セッションを生成したユーザの情報（ロールやリソースグループなど）が変更された場合は、セッション使用中でも操作に反映されます。セッションを生成したユーザのパスワードが変更された場合、セッションが破棄されることがあります。

セッションの破棄

一連の操作が終了してセッション管理が不要になった場合は、セッションを破棄します。セッションは、生成したユーザと Administrator ユーザグループ（ビルトイングループ）に属するユーザだけが破棄できます。

関連リンク

セッションの一覧を取得する（66 ページ）
特定のセッションの情報を取得する（68 ページ）
セッションを生成する（69 ページ）
セッションを破棄する（71 ページ）

2.6　リクエストヘッダ

REST API でサポートするリクエストヘッダについて説明します。

ヘッダ	指定区分	説明	指定できる値
Accept	任意	レスポンスのメディアタイプを指定するヘッダです。	`*/* (json)` デフォルト値：`*/* (json)`
Content-Type	任意	リクエストボディのメディアタイプを指定するヘッダです。 リクエストボディを指定する場合に Content-Type ヘッダを指定できます。リクエストボディがない場合は指定しても無視されます。	`application/json` デフォルト値：`application/json`
Content-Length	任意	リクエストボディのサイズを指定するヘッダです。 リクエストボディを指定する場合に Content-Length ヘッダを指定できます。クライアントソフトウェ	バイト単位で指定します。 デフォルト値：なし

ヘッダ	指定区分	説明	指定できる値
		アの仕様によっては、自動的に付与されます。	
Authorization	必須	認証情報を指定するヘッダです。 バージョン情報の取得の API およびストレージシステムの一覧取得の API の場合は、指定する必要はありません。 REST API を使用するときは、下記のとおり認証方法を使い分けてください。 • ストレージシステムの登録：ユーザ ID とパスワードによる認証 • セッションの生成：ユーザ ID とパスワードによる認証 • 上記以外の操作：セッションによる認証	次のどちらかの形式で指定します。認証方法は、API によって使い分けてください。 • ユーザ ID とパスワードによる認証 `Basic` ***＜認証情報＞*** 認証情報はユーザ ID とパスワードを Base64 でエンコードした文字列を指定してください。ストレージシステムのリソースを操作できるユーザアカウントのユーザ ID とパスワードを使用してください。 • セッションによる認証 `Session` ***＜トークン＞*** セッション生成時に取得したトークンを指定してください。 デフォルト値：なし
Response-Max-Wait	任意	非同期処理の API を発行する場合に、レスポンスを返すまでの最大待ち時間を指定するヘッダです。REST API サーバが API を受け付けた時点から、指定した時間が経過するとレスポンスが返ります。 最大待ち時間が経過する前に処理が完了した場合は、その時点でレスポンスが返ります。 レスポンスを受け取るまでの時間は、ネットワークの状況や REST API サーバの負荷などの影響を受けるため、指定した最大待ち時間を超えることがあります。最大待ち時間は、これらの影響を考慮して指定してください。	`0`〜`1800` の整数 単位：秒 デフォルト値：なし
Response-Job-Status	任意	非同期処理の API を発行する場合に、レスポンスを返してほしいジョブの状態を指定するヘッダです。指定した状態にジョブが遷移した時点またはエラー終了した時点でレスポンスが返ります。	次のどちらかの形式で指定します。 ＜*ジョブのステータス*＞; または ＜*ジョブのステータス*＞; Job-State=＜*ジョブの状態*＞ デフォルト値：なし

上記以外のヘッダが指定された場合、そのヘッダは無視されます。

Response-Max-Wait と Response-Job-Status は組み合わせて指定できます。両方を指定した場合、どちらかの条件が満たされた時点でレスポンスが返ります。

Response-Max-Wait と Response-Job-Status のどちらも指定しない場合は、ただちにレスポンスが返ります。

関連リンク

ジョブオブジェクト（48 ページ）

2.7　レスポンスヘッダ

REST API サーバが返すレスポンスヘッダについて説明します。

ヘッダ	説明	デフォルト
Content-Type	レスポンスデータのメディアタイプを示します。	application/json;charset=UTF-8
Content-Length	レスポンスデータのサイズを示します。 レスポンスデータのサイズが大きい場合、このヘッダは返却されずに、データを分割して転送することを示す「`Transfer-Encoding: chunked`」が返却されます。	なし
Transfer-Encoding	レスポンスデータの転送時のエンコード形式を示します。 サイズが大きいレスポンスデータを分割して転送する場合に、「`chunked`」が返却されます。	なし

WWW-Authenticate	HTTP ステータスコード 401 が返される場合に、認証が必要であることを示します。	• セッション生成時
		`Basic realm="Block storage"`
		• セッション生成時以外
		`Session realm="Block storage"`

2.8　HTTP ステータスコード

REST API は、処理結果を示すために次に示す標準的な HTTP のステータスコードを使用します。

ステータスコード	説明
200	Success リクエストが適切に処理されたことを示します。 情報取得のリクエストでの取得結果が 0 件の場合も、このステータスコードが返ります。
202	Accepted 非同期処理のリクエストの受け付けが完了したことを示します。
400	Bad request リクエストヘッダ、クエリパラメータ、またはリクエストボディが不正であることを示します。
401	Unauthorized リクエストヘッダに Authorization ヘッダが指定されていない、または Authorization ヘッダに指定された情報での認証に失敗したことを示します。
403	Forbidden

ステータスコード	説明
	操作を実行するために必要な権限がないことを示します。
404	Not found URL で指定したリソースが見つからない、またはリソースに対する Read 権限がないことを示します。
405	Method not allowed URL で指定したリソースに対して、許可されていないメソッドを指定したことを示します。
406	Not acceptable Accept ヘッダに、サポートしていないメディアタイプが指定されたことを示します。
409	Conflict URL で指定したリソースに対して、矛盾した状態や不可能な状態への変更を要求したことを示します。 例：作成済みのリソースと同じ ID のリソースを作成しようとした。
411	Length Required Content-Length ヘッダを指定する必要があることを示します。
412	Precondition failed API を実行するための条件を満たしていないことを示します。
415	Unsupported media type Content-Type ヘッダに、サポートしていないメディアタイプを指定したことを示します。
417	Expectation Failed Expect ヘッダの指定に誤りがあるか、Web サーバが Expect ヘッダに対応していないことを示します。
500	Server error REST API サーバまたは操作対象のストレージシステムで内部エラーが発生したことを示します。
502	Proxy Error REST API サーバからの応答がないか、または不正なレスポンスを受信したことを示します。
503	Service unavailable REST API サーバまたは操作対象のストレージシステムがビジー状態でリクエストを受け付けられないことを示します。 このステータスコードが返る場合は、再度リクエストを実行してください。
504	Gateway Timeout REST API サーバから時間内に応答がないことを示します。

ステータスコードのうち、API 固有の説明があるものについては、各 API のセクションで説明します。

関連リンク

リトライ処理の組み込み（105 ページ）

2.9　リクエストおよびレスポンスのフォーマット

リソースの作成、変更時の属性値の指定、またはリソースの情報取得結果には、JSON のフォーマットを使用します。

POST メソッドでリソースを作成、追加したり 、PUT メソッドでリソースを変更、編集したりする場合、JSON 形式でリソースの属性を指定します。GET メソッドでリソースの情報を取得する場合、レスポンスは JSON 形式で返ります。

サポートする文字コードは UTF-8 です。

リクエストの形式

- string 型の属性に空文字を指定した場合は、その属性の値は空になります。
- string 型以外の属性に空文字を指定した場合は、その属性は指定していないものと見なされます。
- 属性の値には、次の文字が使用できます。

  ```
  A-Z a-z 0-9 , - . : @ _
  ```

  各 API の説明にも使用できる文字が記載されている場合があります。その場合は、各 API の規則に従ってください。

  重要

  - コンマ、コロン、単価記号を含む文字列を指定した場合、オブジェクト ID を生成するときにこれらの記号をエンコードする必要があります。これらの記号は使わないよう運用することをお勧めします。これらの記号をエンコードしないままオブジェクト ID を生成した場合、API を発行する際の URL が不正となりエラーとなる場合があります。
  - 値の先頭にハイフンを指定することはできません。

- URL にバックスラッシュまたはスラッシュは指定しないでください。
- IP アドレスを指定する場合、IPv4 射影アドレスは使用できません。

レスポンスの形式

- API の処理が成功した場合、レスポンスは JSON 形式で返ります。
- 処理が失敗した場合、エラーの内容によっては JSON 形式ではなく HTML 形式でレスポンスが返る場合があります。

  プログラム中で HTTP ステータスコードを基にエラー処理を行う場合には、レスポンスヘッダの Content-Type の値をチェックしてください。

2.10　クエリパラメータ

GET メソッドでオブジェクトを取得する際に、クエリパラメータを指定することで特定の条件で実行結果をフィルタリングできます。

クエリパラメータは、URL の末尾に次の形式で指定します。

?＜*パラメータ*＞=＜*値*＞

複数のパラメータを指定する場合は、&記号でつなぎます。複数のパラメータを指定する場合の例を次に示します。

?＜*パラメータ*＞=＜*値*＞&＜*パラメータ*＞=＜*値*＞...

クエリに指定できるパラメータについては、各 API の説明を参照してください。

パラメータは大文字と小文字が区別されます。各 API で指定できるパラメータ以外を指定した場合、無効なパラメータは指定されなかったものとみなし、有効なパラメータだけで実行結果がフィルタリングされます。

パラメータの値が RFC3986 に定められた予約文字を含む場合は、エンコードした文字列を指定してください。RFC3986 に定められた予約文字については、オブジェクト ID の指定方法を参照してください。

パラメータの値に IP アドレスを指定する場合、IPv4 射影アドレスは使用できません。

関連リンク

オブジェクト ID の指定方法（35 ページ）

2.11　データ型

REST API で指定できるデータの型について説明します。

REST API がサポートするデータ型と対応する JSON のデータ型を次に示します。

データ型	JSON のデータ型	説明
boolean	boolean	`true` または `false` を表す型。 例：`true`
int	number	32 ビットの符号付き整数を表す型。 例：`100`
long	number	64 ビットの符号付き整数を表す型。 例：`1048576`
string	string	任意の文字列を表す型。 例：`"host_group_1"`
ISO8601string	string	ISO 8601 拡張形式（YYYY-MM-DDThh:mm:ssZ）で時刻を表す型。 指定できるタイムゾーンは UTC だけです。

データ型	JSON のデータ型	説明
		例："2015-03-20T09:27:35Z"
link	string	URL のパスを表す型。 link 型は、リソースの URL を示します。例えば、非同期処理のリクエストを発行時にジョブオブジェクトへの URL を link 型で返します。 link 型は、URL からプロトコル、ホスト名、ポート番号を除いた文字列になります。link 型を基に URL を構成する場合は、プロトコル、ホスト名、ポート番号を補って使用してください。

上記のデータ型以外に、JSON 形式の次のデータ型を使用します。

- object 型

  属性と値をコロン（：）でつないだ文字列を{ }で囲む形式です。属性と値のペアが複数ある場合は、コンマで区切ります。

- array 型

  複数の値をコンマで区切った文字列を[ ]で囲む形式です。

2.12　出力形式

API を発行すると、API の処理方式、API の処理種別、実行結果に応じてレスポンスが返ります。

リクエストの処理が成功した場合のレスポンスの出力形式について次に示します。

API の処理方式	API の処理種別	実行結果のステータスコード	出力形式
同期処理	GET （単一のオブジェクトの取得）	200	各 API のレスポンスメッセージの説明を参照
	GET （複数のオブジェクトの取得）	200	データオブジェクト
	GET（Action テンプレート）	200	Action テンプレートオブジェクト
	上記以外	200	各 API のレスポンスメッセージの説明を参照
非同期処理	すべて	202	ジョブオブジェクト

リクエストの処理が失敗した場合は、レスポンスとしてエラーオブジェクトが返ります。

2.13　データオブジェクト

データオブジェクトは、オブジェクトのリストを返すためのオブジェクトです。

objects ドメインの GET 操作でリストを取得する場合

データオブジェクトのスキーマを次に示します。

属性	データ型	説明
data	array	オブジェクトのリスト

views ドメインの GET 操作でリストを取得する場合

データオブジェクトのスキーマを次に示します。

属性	データ型	説明
data	array	リソース情報のリスト
offset	int	データの取得開始位置
count	int	取得したデータの件数
totalCount	int	データの総件数

総件数 100 件のデータについて、51 件目から 10 件のデータを取得した場合の例を次に示します。

```
{
  "data": [
       {
          ...
       },
       ...
  ],
  "offset": 50,
  "count": 10,
  "totalCount": 100
}
```

2.14　ジョブオブジェクト

ジョブオブジェクトは、非同期処理の API を発行したときに返るジョブ情報のオブジェクトです。

ジョブオブジェクトのスキーマを次に示します。

属性	データ型	説明
jobId	long	ジョブのオブジェクト ID
self	link	ジョブの情報にアクセスするための URL
userId	string	ジョブを登録する契機となる API を発行したユーザ ID
status	string	ジョブのステータス 次の値が返ります。 • `Initializing`：ジョブが初期化中であることを示す • `Running`：ジョブが実行中であることを示す • `Completed`：ジョブが実行完了したことを示す
state	string	ジョブの状態 次の値が返ります。 • `Queued`：ジョブがキューイングされた状態を示す

属性	データ型	説明
		• `Started`：ジョブが開始された状態を示す • `Succeeded`：ジョブが成功した状態を示す • `Failed`：ジョブが失敗した状態を示す • `Unknown`：ジョブの状態が不明なことを示す
createdTime	ISO8601string	ジョブが作成された時刻
updatedTime	ISO8601string	ジョブの状態が更新された時刻
completedTime	ISO8601string	ジョブが終了した時刻
request	Request Object	リクエストの情報を保持するオブジェクト
affectedResources	link[]	操作対象のリソースにアクセスするための URL 1 つの API で複数のリソースを操作する場合は、すべての操作対象のリソースの URL が返ります。ジョブが途中で失敗した場合は、処理が完了したことを確認できたリソースの URL だけが返ります。 リソースの削除操作が成功した場合、削除対象のリソースの URL が返ります。この URL にアクセスすると 404 エラーとなり、正常に削除されたことが確認できます。 ジョブの情報を取得する API のレスポンスにも affectedResources が含まれます。この場合、ジョブを登録する契機となった API の操作対象のリソースにアクセスするための URL が返ります。
error	Error Object	エラーの情報を保持するオブジェクト

重要

ジョブの情報の最大保持件数を次に示します。最大保持件数を超えたジョブの情報は、createdTime の古い順に削除されます。

- iStorage V シリーズ：3,000 件

2.15　エラーオブジェクト

エラーオブジェクトは、リクエストの処理に失敗したときに返すエラー情報のオブジェクトです。

API の処理が失敗すると、レスポンスデータとしてエラーオブジェクトが返ります。エラーオブジェクトのスキーマを次に示します。

属性	データ型	説明
errorSource	link	エラーが発生した URL
messageId※	string	メッセージ ID
message	string	エラーメッセージの内容
cause	string	エラーの要因
solution	string	エラーの対処
solutionType	string	エラーの対処の分類 • `RETRY`：リトライで対処可能なエラー • `SEE_ERROR_DETAIL`：エラーメッセージの内容に基づいた対処が必要なエラー

属性	データ型	説明
		この属性に RETRY が返る場合、失敗したリクエストをリトライしてください。 リクエストの内容は、ジョブオブジェクトの request の値で確認できます。
errorCode	object	ストレージシステムのエラーコード ストレージシステムでエラーが発生して、次に示すエラーコードがある場合にだけ値が返ります。 • RAID Manager の SSB1 コードおよび SSB2 コード • RAID Manager のエラーコード • RMI のエラーコード 1 およびエラーコード 2 ストレージシステムのエラーコードは、ストレージシステムの保守に必要になります。
detailCode	string	エラーの詳細な情報 次の形式で表示されます。 *<nnnnnZ>-<TYPE><TYPE ごとの出力形式>* • *<nnnnnZ>* REST API のメッセージ ID を表示します。 - *nnnnn* メッセージの通し番号 - *Z* メッセージの種類 I：Information W：Warning E：Error • *<TYPE>* エラーの種別を表示します。 - 0：REST API サーバ側のエラー - 上記以外：ストレージシステム側のエラー • *<TYPE ごとの出力形式>* *TYPE* の値によって、出力形式が次のように異なります。 - 0 の場合 出力されません。 - 2 の場合 RAID Manager のエラーについて、次の形式で出力されます。 *-< SSB1 コード>-< SSB2 コード>* 詳細については、RAID Manager のマニュアルを参照してください。 - 3 の場合 RAID Manager のエラーコードが出力されます。 詳細については、RAID Manager のマニュアルを参照してください。 - 4 の場合 GUM で発生したエラーについて、次の形式で出力されます。 *-< 部位コード>-< エラーコード>* 詳細については、マニュアル『Storage Navigator メッセージガイド』を参照してください。

注※　属性名は messageID で返ることがあります。

detailCode 属性の出力例を次に示します。

REST API サーバでエラーが発生した場合（メッセージ ID：KART40231-E）

```
"detailCode": "40231E-0"
```

ストレージシステム側でエラーが発生した場合（メッセージ ID：KART30000-E、SSB1 コード：2EDA、SSB2 コード：00EE）

```
"detailCode": "30000E-2-2EDA-00EE"
```

ストレージシステム側でエラーが発生した場合（メッセージ ID：KART30000-E、RAID Manager のエラーコード：EX_INVARG）

```
"detailCode": "30000E-3-EX_INVARG"
```

ストレージシステム側でエラーが発生した場合（メッセージ ID：KART30007-E、部位コード：30762、エラーコード：204092）

```
"detailCode": "30007E-4-30762-204092"
```

関連リンク

リトライ処理の組み込み（105 ページ）

2.16　リクエストオブジェクト

リクエストオブジェクトは、リクエストの情報を保持するためのオブジェクトです。

リクエストオブジェクトのスキーマを次に示します。

属性	データ型	説明
requestUrl	link	非同期処理の API でリクエストした URL URL の文字列が 2048 バイトを超える場合、文字列は途中で省略されます。
requestMethod	string	非同期処理の API でリクエストした HTTP メソッド
requestBody	string	非同期処理の API でリクエストしたリクエストボディ リクエストボディの文字列が 1024 バイトを超える場合、文字列は途中で省略されます。

2.17　Action テンプレートオブジェクト

Action テンプレートオブジェクトは、Action を実行するために必要なリクエストボディのひな型です。GET メソッドで Action テンプレートオブジェクトを取得して、実行したい Action に合わせてテンプレートに値を設定し、リクエストボディに指定して実行します。

Action テンプレートオブジェクトのスキーマを次に示します。

属性	データ型	説明
parameters	object	操作実行時に必要なパラメータ

Action テンプレートには、Action の実行時に指定が必要な属性があらかじめ記載されています。値には、空であることを示す `null` または`[]`が設定されています。これらの属性に値を指定してください。

必要な属性の行を残して、属性値を設定したものをリクエストボディに指定して Action を実行します。

第 3 章
REST API で共通の操作

この章では、セッションの生成やジョブの情報取得など、REST API で共通の操作について説明します。

3.1 バージョン情報を取得する

REST API のバージョン情報を取得します。

実行権限

この API の実行に必要なロールはありません。

リクエストヘッダ

この API は認証されないため、Authorization ヘッダの指定は不要です。

リクエストライン

```
GET ＜ベース URL＞/configuration/version
```

リクエストメッセージ

オブジェクト ID

なし。

クエリパラメータ

なし。

ボディ

なし。

レスポンスメッセージ

ボディ

```
{
  "productName": "Configuration Manager REST API",
```

```
  "apiVersion": "1.57.0"
}
```

属性	型	説明
productName	string	REST API の名称
apiVersion	string	REST API のバージョン

ステータスコード

この操作のリクエストに対する ステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -X
GET https://192.0.2.100:23451/ConfigurationManager/configuration/version
```

関連リンク

HTTP ステータスコード（43 ページ）

3.2　ストレージシステムの一覧を取得する

REST API から操作できるストレージシステムの一覧を取得します。ストレージシステムのストレージデバイス ID やシリアル番号などの情報を確認できます。

実行権限

この API の実行に必要なロールはありません。

リクエストヘッダ

この API は認証されないため、Authorization ヘッダの指定は不要です。

リクエストライン

```
GET ＜ベース URL＞/v1/objects/storages
```

リクエストメッセージ

オブジェクト ID

なし。

クエリパラメータ

なし。

ボディ

なし。

レスポンスメッセージ

ボディ

```
{
  "data": [
    {
      "storageDeviceId" : "934000123458",
      "model" : "iStorage V100",
      "serialNumber" : 123458,
      "ctl1Ip" : "192.0.2.104",
      "ctl2Ip" : "192.0.2.105",
      "targetCtl" : "CTL1"
    }
  ]
}
```

属性	型	説明
storageDeviceId	string	ストレージデバイス ID
model	string	ストレージシステムのモデル名
serialNumber	int	ストレージシステムのシリアル番号
ctl1Ip	string	ストレージシステムのコントローラ 1 の IP アドレス ストレージシステム側で IP アドレスに IPv4 および IPv6 の両方が設定されている場合は、IPv4 アドレスが出力されます。
ctl2Ip	string	ストレージシステムのコントローラ 2 の IP アドレス ストレージシステム側で IP アドレスに IPv4 および IPv6 の両方が設定されている場合は、IPv4 アドレスが出力されます。
targetCtl	string	REST API が操作対象とするコントローラ • `CTL1`：コントローラ 1 • `CTL2`：コントローラ 2

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -X GET https://192.0.2.100:23451/Confi
gurationManager/v1/objects/storages
```

関連リンク

HTTP ステータスコード（43 ページ）

3.3　特定のストレージシステムの情報を取得する

ストレージデバイス ID で指定したストレージシステムについて、詳細な情報を取得します。

実行権限

ストレージ管理者（参照）

リクエストライン

```
GET ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞
```

リクエストメッセージ

オブジェクト ID

ストレージシステムの情報取得で取得した storageDeviceId の値を指定します。

属性	型	説明
storageDeviceId	string	（必須）ストレージデバイス ID

クエリパラメータ

パラメータ	型	フィルタ条件
detailInfoType	string	（任意）取得する詳細情報のタイプ • version ストレージシステム、コントローラ 1、コントローラ 2 のマイクロコードの詳細な情報を追加します。

ボディ

なし。

レスポンスメッセージ

ボディ

```
{
  "storageDeviceId": "934000123456",
  "model": "iStorage V100",
  "serialNumber": 123456,
  "ctl1Ip": "192.0.10.10",
  "ctl2Ip": "192.0.10.11",
  "dkcMicroVersion": "93-05-21/01",
  "communicationModes": [
    {
      "communicationMode": "lanConnectionMode"
    }
  ],
  "isSecure": true,
  "targetCtl": "CTL1",
  "usesSvp": false
}
```

属性	型	説明
storageDeviceId	string	ストレージデバイス ID
model	string	ストレージシステムのモデル名
serialNumber	int	ストレージシステムのシリアル番号
usesSvp	boolean	常に `false` が表示されます。
ctl1Ip	string	ストレージシステムのコントローラ 1 の IP アドレス ストレージシステム側で IP アドレスに IPv4 および IPv6 の両方が設定されている場合は、IPv4 アドレスが出力されます。
ctl2Ip	string	ストレージシステムのコントローラ 2 の IP アドレス ストレージシステム側で IP アドレスに IPv4 および IPv6 の両方が設定されている場合は、IPv4 アドレスが出力されます。
targetCtl	string	REST API が操作対象とするコントローラ • `CTL1`：コントローラ 1 • `CTL2`：コントローラ 2
dkcMicroVersion	string	ストレージシステムのマイクロコードのバージョン
communicationModes	object[]	通信モードの配列 REST API サーバとストレージシステムの通信モードについて次の属性が表示されます。 • communicationMode（string） 通信モード - `fcConnectionMode` - `lanConnectionMode`
isSecure	boolean	REST API サーバとストレージシステム間の SSL 通信の設定 • `true`：SSL 通信が有効

クエリパラメータで `detailInfoType` に `version` を指定して実行すると、ストレージシステムのマイクロコードの詳細情報も取得されます。

```
{
  "storageDeviceId": "934000123456",
  "model": "iStorage V100",
  "serialNumber": 123456,
  "ctl1Ip" : "192.0.10.10",
  "ctl2Ip" : "192.0.10.11",
  "dkcMicroVersion": "88-01-01/82",
  "detailDkcMicroVersion": "88-01-01-60/82",
  "ctl1MicroVersion" : "88-01-01/81",
  "ctl2MicroVersion" : "88-01-01/81",
  "communicationModes": [
    {
      "communicationMode": "lanConnectionMode"
    }
  ],
  "isSecure": true
}
```

属性	型	説明
detailDkcMicroVersion	string	ストレージシステムのマイクロコードのバージョン モデル識別情報を含みます。
ctl1MicroVersion	string	コントローラ 1 の GUM のバージョン コントローラ 1 の GUM に障害が発生している場合は取得されません。
ctl2MicroVersion	string	コントローラ 2 の GUM のバージョン コントローラ 2 の GUM に障害が発生している場合は取得されません。

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X GET https://192.0.2.100:23451/ConfigurationManager/v1/objects/storages/834000123456
```

関連リンク

HTTP ステータスコード（43 ページ）

3.4　ストレージシステムを登録する

REST API からのストレージ運用を開始するために、REST API サーバにストレージシステムの情報を登録します。対象となるストレージシステムの IP アドレスやポート番号などの情報が変更された場合、ストレージシステムの情報を削除してから再度登録します。この API

では、Authorization ヘッダにストレージシステムにアクセスするためのユーザ ID とパスワードを指定してください。指定したユーザ情報はストレージシステムの情報として REST API サーバに登録されます。

重要

- REST API サーバに登録されたユーザ情報は、ストレージシステムの構成情報の更新や、REST API サーバ内の情報を更新するための情報取得時に、REST API サーバによって利用されます。
- REST API サーバにストレージシステムを登録したあとに、登録したユーザのパスワードを REST API や Storage Navigator などで変更した場合、REST API サーバに登録されているストレージシステムの情報には変更内容が反映されません。ストレージシステムの情報を変更する API で、変更後のパスワードを REST API サーバにも登録してください。

実行権限

セキュリティ管理者（参照）またはセキュリティ管理者（参照・編集）

リクエストライン

```
POST　＜ベース URL＞/v1/objects/storages
```

リクエストメッセージ

オブジェクト ID

なし。

クエリパラメータ

なし。

ボディ

- ```
  {
    "ctl1Ip": "192.0.10.10",
    "ctl2Ip": "192.0.10.11",
    "serialNumber": 123456,
    "model": "iStorage V100",
    "changeNotificationSetting": {
      "isNotifiable": true
    }
  }
  ```

属性	型	説明
serialNumber	int	（必須）登録するストレージシステムのシリアル番号
model	string	（必須）登録するストレージシステムのモデル名
  ```

属性	型	説明
		次の値を指定します。 • iStorage V100 • iStorage V300
targetCtl	string	（任意）REST API が操作対象とするコントローラ 次の値を指定します。 • CTL1：コントローラ 1 • CTL2：コントローラ 2 省略した場合、CTL1 が設定されます。
ctl1Ip	string	（任意）ストレージシステムのコントローラ 1 の IP アドレス IPv4 または IPv6 の IP アドレスを指定できます。ストレージシステム側でコントローラ 1 の IP アドレスに IPv4 および IPv6 の両方が設定されている場合は、IPv4 アドレスを指定してください。 省略した場合、値が自動で設定されます。 targetCtl 属性に CTL1 を指定した場合、または省略した場合は、この属性を必ず指定してください。
ctl2Ip	string	（任意）ストレージシステムのコントローラ 2 の IP アドレス IPv4 または IPv6 の IP アドレスを指定できます。ストレージシステム側でコントローラ 2 の IP アドレスに IPv4 および IPv6 の両方が設定されている場合は、IPv4 アドレスを指定してください。 省略した場合、値が自動で設定されます。 targetCtl 属性に CTL2 を指定した場合は、この属性を必ず指定してください。
isSecure	boolean	（任意）REST API サーバとストレージシステム間の SSL 通信の設定 • true：SSL 通信を有効にする 省略した場合、true が設定されます。 通信の安全を確保するため、SSL 通信を有効にする設定だけが実行できます。
changeNotificationSetting	object	（任意）ストレージシステムからの構成変更の通知を受信するための設定 changeNotificationSetting 属性を省略した場合は、isNotifiable 属性を省略したときの動作と同じです。 •（任意）isNotifiable（boolean） ストレージシステムからの構成変更の通知を受信するかどうか この属性は、true を指定することを強く推奨します。 - true：受信する - false：受信しない isNotifiable 属性に true が指定された場合、内部の構成変更の通知テストに失敗するとエラーになります。 省略された場合、isNotifiable 属性に true が指定されたと見なして内部の処理が進みますが、構成変更の通知テストに失敗したときは isNotifiable 属性を false に変更して API は正常終了します。 構成変更の通知を受信するための設定が意図したとおりに設定されているかどうかは、ストレージシステムの

属性	型	説明
		構成変更の通知先の一覧を取得する API を実行して確認します。構成変更の通知先として Configuration Manager REST API サーバが登録されていれば、`true` を指定した場合の設定が完了していると判断できます。

レスポンスメッセージ

ボディ

```
{
  "storageDeviceId": "934000123456",
  "model": "iStorage V100",
  "serialNumber": 123456
}
```

属性	型	説明
storageDeviceId	string	登録したストレージシステムのストレージデバイス ID
model	string	登録したストレージシステムのモデル名
serialNumber	int	登録したストレージシステムのシリアル番号

ステータスコード

この操作のリクエストに対するステータスコードの意味を次に示します。その他のステータスコードについては、HTTP ステータスコードの説明を参照してください。

ステータスコード	メッセージ	説明
409	Conflict	指定したストレージデバイス ID で、ストレージシステムを登録済みです。
412	Precondition Failed	登録できるストレージシステムの最大数を超えるため、指定されたアクションを実行できません。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -u
rest-test:rest-api -X POST --data-binary @./InputParameters.json https://19
2.0.2.100:23451/ConfigurationManager/v1/objects/storages
```

関連リンク

ユーザ認証（37 ページ）
HTTP ステータスコード（43 ページ）
ストレージシステムの一覧を取得する（54 ページ）

3.5　ストレージシステムの情報を変更する

REST API サーバに登録されているストレージシステムのユーザ情報やそれ以外の情報を変更します。

この API は、操作の目的に応じて次のように実行してください。

ユーザ情報を変更する場合

Authorization ヘッダに変更後のユーザ ID とパスワードを指定し、リクエストボディなしで API を実行してください。Authorization ヘッダに指定したユーザ情報が、ストレージシステムのユーザ情報に反映されます。リクエストボディに属性を指定するリクエストとは別のリクエストとして実行してください。

ユーザ情報以外の情報を変更する場合

リクエストボディに属性を指定して実行します。指定した情報だけが変更され、ストレージシステムのユーザ情報は更新されません。

リクエストボディに `targetCtl` 属性を指定する場合（REST API が操作対象とするコントローラを変更する場合）、Authorization ヘッダにユーザ ID とパスワードを指定して API を実行する必要があります。

重要

- 次に示す情報を変更する場合、REST API サーバからストレージシステムの情報を削除してから、対象となるストレージシステムの設定を変更したあとで、再度ストレージシステムの情報を登録します。
  - ストレージシステムの IP アドレスやポート番号などを変更する場合
- REST API サーバにストレージシステムを登録したあとに、登録したユーザのパスワードを REST API や Storage Navigator などで変更した場合、REST API サーバに登録されているストレージシステムの情報には変更内容が反映されません。この API で、変更後のパスワードを REST API サーバにも登録してください。

実行権限

セキュリティ管理者（参照）またはセキュリティ管理者（参照・編集）

リクエストライン

```
PUT ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞
```

この API は PATCH メソッドでも実行できます。

リクエストメッセージ

オブジェクト ID

ストレージシステムの情報取得で取得した storageDeviceId の値を指定します。

属性	型	説明
storageDeviceId	string	（必須）ストレージデバイス ID

クエリパラメータ

なし。

ボディ

```
{
  "targetCtl" : "CTL2"
}
```

属性	型	説明
targetCtl	string	（任意）REST API が操作対象とするコントローラ 次の値を指定します。 • CTL1：コントローラ 1 • CTL2：コントローラ 2

レスポンスメッセージ

ボディ

```
{
  "storageDeviceId" : "934000123456",
  "model" : "iStorage V100",
  "serialNumber" : 123456
}
```

属性	型	説明
storageDeviceId	string	情報を変更したストレージシステムのストレージデバイス ID
model	string	情報を変更したストレージシステムのモデル名
serialNumber	int	情報を変更したストレージシステムのシリアル番号

Action テンプレート

なし。

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -u
rest-test:rest-api -X PUT https://192.0.2.100:23451/ConfigurationManager/v1
/objects/storages/834000123456
```

関連リンク

HTTP ステータスコード（43 ページ）
ストレージシステムの一覧を取得する（54 ページ）
ストレージシステムの構成変更の通知先を登録する（100 ページ）

3.6　ストレージシステムの情報を削除する

ストレージシステムの情報を REST API サーバから削除して、REST API の操作対象から除外します。ストレージシステムの IP アドレスやポート番号などの設定が変更されたり、ストレージシステムが撤去されたりする場合、この API でストレージシステムの情報を REST API サーバから削除したあとで、ストレージシステム側の設定を変更してください。ストレージシステムの情報を REST API サーバから削除する前にストレージシステム側の設定を変更してしまうと、`force` 属性を指定して強制的に削除する必要があります。 この API の処理は数分掛かることがあります。

実行権限

セキュリティ管理者（参照）またはセキュリティ管理者（参照・編集）

リクエストライン

```
DELETE　＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞
```

リクエストメッセージ

オブジェクト ID

ストレージシステムの情報取得で取得した `storageDeviceId` の値を指定します。

属性	型	説明
storageDeviceId	string	（必須）ストレージデバイス ID

クエリパラメータ

なし。

ボディ

なし。

レスポンスメッセージ

ボディ

```
{
  "storageDeviceId" : "934000123456",
  "model" : "iStorage V100",
  "serialNumber" : 123456
}
```

属性	型	説明
storageDeviceId	string	削除したストレージシステムのストレージデバイス ID
model	string	削除したストレージシステムのモデル名
serialNumber	int	削除したストレージシステムのシリアル番号

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -H
"Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X DELETE http
s://192.0.2.100:23451/ConfigurationManager/v1/objects/storages/834000123456
```

ストレージシステムの情報を強制的に削除する場合

この API を実行する前に対象となるストレージシステムの設定が変更されたり、ストレージシステムが撤去されたりした場合、force 属性を指定することでストレージシステムの情報を強制的に削除できます。force 属性を指定する場合、管理サーバからホスト名にローカルホスト（localhost、127.0.0.1 または::1）を指定してこの API を実行してください。force 属性を設定する場合のコード例を次に示します。

```
{
  "force": true
}
```

属性	型	説明
force	boolean	（任意）REST API で保持しているストレージシステムの情報を強制的に削除するかどうかを指定します。 • `true`：ストレージシステムの情報を強制的に削除する • `false`:ストレージシステムの情報を強制的に削除しない 省略した場合、`false` が設定されたと見なされます。

メモ

ストレージシステムの情報を強制的に削除しても、構成変更の通知先として登録された REST API サーバの情報はストレージシステムから削除されません。通知先の情報を削除するには、再度ストレージシステムを登録してから、通知先を削除する API を実行してください。

関連リンク

HTTP ステータスコード（43 ページ）
ストレージシステムの一覧を取得する（54 ページ）
ストレージシステムの構成変更の通知先を削除する（102 ページ）

3.7　セッションの一覧を取得する

REST API サーバ上の有効なセッションの一覧を取得します。この操作は、Administrator ユーザグループ（ビルトイングループ）に属するユーザだけが実行できます。

実行権限

Administrator ユーザグループ（ビルトイングループ）

リクエストライン

```
GET ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞/sessions
```

リクエストメッセージ

オブジェクト ID

なし。

クエリパラメータ

なし。

ボディ

なし。

レスポンスメッセージ

ボディ


```
{
  "data": [
    {
      "sessionId": 8,
      "userId": "rest-user",
      "ipAddress": "192.0.2.100",
      "createdTime": "2015-09-14T01:02:24Z",
      "lastAccessedTime": "2015-09-14T01:02:24Z"
    },
    {
      "sessionId": 6,
      "userId": "api-user",
      "ipAddress": "192.0.2.100",
      "createdTime": "2015-09-14T00:59:58Z",
      "lastAccessedTime": "2015-09-14T00:59:58Z"
    },
    {
      "sessionId": 5,
      "userId": "admin-user",
      "ipAddress": "192.0.2.100",
      "createdTime": "2015-09-14T00:59:53Z",
      "lastAccessedTime": "2015-09-14T00:59:53Z"
    }
  ]
}
```


属性	型	説明
sessionId	int	セッション ID
userId	string	セッションを生成したユーザ ID
ipAddress	string	セッションを生成した REST API クライアントの IP アドレス REST API クライアントから別のサーバを経由して REST API サーバにアクセスしている場合は、クライアントの IP アドレスと経由したサーバの IP アドレスを連結した文字列（REST API サーバが受信した X-Forwarded-For ヘッダの内容）が出力されます。
createdTime	ISO8601string	セッションが生成された時刻
lastAccessedTime	ISO8601string	セッションが最後に使用された時刻

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -H
"Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X GET https:/
/192.0.2.100:23451/ConfigurationManager/v1/objects/storages/836000123456/se
ssions/
```

関連リンク

HTTP ステータスコード（43 ページ）

3.8　特定のセッションの情報を取得する

セッション ID を指定して、REST API サーバ上で有効なセッションの情報を取得します。リクエストの Authorization ヘッダには、セッションのトークンを指定してください。

実行権限

ストレージ管理者（参照）

リクエストライン

```
GET ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞/sessions/＜オブジェクト ID＞
```

リクエストメッセージ

オブジェクト ID

セッション生成時に取得した sessionId の値を指定します。

属性	型	説明
sessionId	int	（必須）セッション ID

クエリパラメータ

なし。

ボディ

なし。

レスポンスメッセージ

ボディ

```
{
  "token": "97c13b80-8244-4b36-bc21-03026205fa64",
  "sessionId": 9
}
```

属性	型	説明
sessionId	int	セッション ID
token	string	トークン

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -H
"Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X GET https:/
/192.0.2.100:23451/ConfigurationManager/v1/objects/storages/836000123456/se
ssions/9
```

関連リンク

HTTP ステータスコード（43 ページ）
セッションの一覧を取得する（66 ページ）

3.9　セッションを生成する

セッションを生成して、REST API サーバでセッション管理を行います。1 ストレージシステム当たり、最大 64 セッションを生成できます。最大セッション数を超えると HTTP ステータスコード（503）が返ります。この場合は、しばらくしてから再度リクエストを実行してください。

実行権限

ストレージ管理者（参照）

リクエストライン

```
POST ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞/sessions
```

リクエストメッセージ

オブジェクト ID

なし。

クエリパラメータ

なし。

ボディ

セッションタイムアウトまでの時間を指定する場合のコード例を次に示します。

```
{
  "aliveTime": 5
}
```

属性	型	説明
aliveTime	long	（任意）セッションタイムアウトまでの時間（秒） 1～300 の値を指定します。※ 省略した場合、300 が指定されたと見なされます。
authenticationTimeout	long	（任意）認証処理でのタイムアウトまでの時間（秒） 外部認証サーバでユーザ認証している場合に指定します。 ストレージシステムの外部認証の設定に合わせて変更してください。 1～900 の値を指定します。 省略した場合、120 が指定されたと見なされます。

注※

- 実際にセッションタイムアウトするまでの時間は、-20 秒から+5 秒の誤差があります。セッション の無効を示す KART40047-E で API の実行に失敗した場合、セッションを再度生成して API を実 行してください。
- 下記の条件に該当する場合、aliveTime 属性に 20 秒未満を指定してセッションを生成したあとに API を実行すると、装置の負荷状況によっては API の実行に失敗することがあります。 aliveTime 属性には 20 秒以上を指定してください。
  - iStorage V シリーズかつ REST API サーバの通信モードが fcConnectionMode の場合

レスポンスメッセージ

ボディ

```
{
  "token": "b74777a3-f9f0-4ea8-bd8f-09847fac48d3",
  "sessionId": 3
}
```

属性	型	説明
sessionId	int	セッション ID セッションを管理するための ID です。
token	string	トークン リクエストの発行元が特定のユーザであることを識別するための情報です。

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -u
rest-test:rest-api -X POST https://192.0.2.100:23451/ConfigurationManager/v
1/objects/storages/836000123456/sessions/ -d ""
```

関連リンク

ユーザ認証（37 ページ）
セッション管理（40 ページ）
HTTP ステータスコード（43 ページ）
セッションの一覧を取得する（66 ページ）
リトライ処理の組み込み（105 ページ）

3.10　セッションを破棄する

不要になったセッションを破棄します。リクエストの Authorization ヘッダには、破棄するセッションのトークンを指定してください。

実行権限

ストレージ管理者（参照）

リクエストライン

```
DELETE ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞/sessions/＜オブジェクト ID＞
```

リクエストメッセージ

オブジェクト ID

セッション生成時に取得した sessionId の値を指定します。Administrator ユーザグループ（ビルトイングループ）に属するユーザは、セッション情報取得で取得した sessionId の値を指定できます。

属性	型	説明
sessionId	int	（必須）セッション ID

クエリパラメータ

なし。

ボディ

```
{
  "force": true
}
```

属性	型	説明
force	boolean	ほかのユーザが生成したセッションも強制的に破棄するかどうかを指定します。この属性は、Administrator ユーザグループ（ビルトイングループ）に属するユーザだけが指定できます。 • true：ほかのユーザのセッションも強制的に破棄する • false：自分が生成したセッションだけを破棄する 省略した場合、false が指定されたと見なされます。

レスポンスメッセージ

ボディ

なし。

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -H
"Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X DELETE --da
ta-binary @./InputParameters.json https://192.0.2.100:23451/ConfigurationMa
nager/v1/objects/storages/836000123456/sessions/1
```

関連リンク

HTTP ステータスコード（43 ページ）
セッション管理（40 ページ）
セッションの一覧を取得する（66 ページ）

3.11　ジョブの情報の一覧を取得する

ユーザが REST API から投入したジョブの情報の一覧を取得します。ストレージ管理者（システムリソース管理）のロールを持つユーザグループに属するユーザの場合、登録されているすべてのジョブについて情報を取得できます。ジョブの情報は発行した API の内容を確認したり、ストレージシステムで発生した問題の原因を特定する情報として利用したりします。

実行権限

この API の実行に必要なロールはありません。対象となるストレージシステムに認証が通るユーザで発行します。

リクエストライン

```
GET ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞/jobs
```

リクエストメッセージ

オブジェクト ID

なし。

クエリパラメータ

クエリパラメータを指定しない場合、ユーザが参照できるジョブ情報のうちジョブの投入時刻が新しいものから 100 件を取得します。

パラメータ	型	フィルタ条件
startCreatedTime	ISO8601string	（任意）取得するジョブ投入時刻の始点を YYYY-MM-DDThh:mm:ssZ 形式で指定します。 指定された時刻以降（指定時刻を含む）に投入されたジョブ情報を取得します。
endCreatedTime	ISO8601string	（任意）取得するジョブ投入時刻の終点を YYYY-MM-DDThh:mm:ssZ 形式で指定します。 指定された時刻より前（指定時刻を含まない）に投入されたジョブ情報を取得します。
count	int	（任意）取得するジョブの件数を 1～100 の値で指定します。

パラメータ	型	フィルタ条件
		指定された件数を上限としてジョブ情報を取得します。 省略した場合、100 が指定されます。
status	string	（任意）取得するジョブの状態（Status）として、次の値を指定します。 • Initializing：初期化中 同時に state を指定する場合、state には Queued を指定します。 • Running：実行中 同時に state を指定する場合、state には Started を指定します。 • Completed：実行完了 同時に state を指定する場合、state には Succeeded、Failed、Unknown のどれかを指定します。
state	string	（任意）取得するジョブの状態（State）として、次の値を指定します。 • Queued：ジョブがキューイングされた状態 • Started：ジョブが開始された状態 • Succeeded：ジョブが成功した状態 • Failed：ジョブが失敗した状態 • Unknown：ジョブの状態が不明

例として、2015/05/01 08:00:00 以降 2015/05/31 23:59:59 以前のジョブ情報のうち正常終了したものをジョブ投入時刻の新しいものから 30 件を上限として取得する場合を次に示します。

```
?startCreatedTime=2015-05-01T08:00:00Z&endCreatedTime=2015-05-31T23:59:5
9Z&count=30&state=Succeeded
```

ボディ

なし。

レスポンスメッセージ

ボディ

```
{
  "data": [
    {
      "jobId": 2,
      "self": "/ConfigurationManager/v1/objects/storages/934000123456/jo
bs/2",
      "userId": "rest-test",
      "status": "Completed",
      "state": "Succeeded",
      "createdTime": "2021-04-04T05:05:51Z",
      "updatedTime": "2021-04-04T05:05:51Z",
      "completedTime": "2021-04-04T05:05:51Z",
      "request": {
        "requestUrl": "/ConfigurationManager/v1/objects/storages/9340001
23456/remote-storages/934000123457",
```

```
        "requestMethod": "DELETE",
        "requestBody": "{\"isMutualDeletion\":false}"
      },
      "affectedResources": [
        "/ConfigurationManager/v1/objects/storages/934000123456/remote-s
torages/934000123457"
      ]
    },
    {
      "jobId": 1,
      "self": "/ConfigurationManager/v1/objects/storages/934000123456/jo
bs/1",
      "userId": "rest-test",
      "status": "Completed",
      "state": "Succeeded",
      "createdTime": "2021-04-04T05:06:01Z",
      "updatedTime": "2021-04-04T05:06:05Z",
      "completedTime": "2021-04-04T05:06:05Z",
      "request": {
        "requestUrl": "/ConfigurationManager/v1/objects/storages/9340001
23456/remote-storages",
        "requestMethod": "POST",
        "requestBody": "{\"storageDeviceId\":\"934000123457\",\"restServ
erIp\":\"192.0.2.101\",\"restServerPort\":443,\"isMutualDiscovery\":fals
e}"
      },
      "affectedResources": [
        "/ConfigurationManager/v1/objects/storages/934000123456/remote-s
torages/934000123457"
      ]
    }
  ]
}
```


属性	型	説明
data	object[]	ユーザが REST API から作成したジョブの情報（ジョブオブジェクト） 1 度に最大で 100 件のジョブ情報を取得します。

ジョブオブジェクトのスキーマについては、ジョブオブジェクトの説明を参照してください。

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -H
"Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X GET https:/
/192.0.2.100:23451/ConfigurationManager/v1/objects/storages/836000123456/jo
bs
```

関連リンク

HTTP ステータスコード（43 ページ）
ジョブオブジェクト（48 ページ）

3.12　特定のジョブの情報を取得する

ジョブ ID を指定して、ユーザが非同期 API から投入したジョブの情報を任意のタイミングで取得します。ストレージ管理者（システムリソース管理）のロールを持つユーザグループに属するユーザの場合、他ユーザが投入したジョブについてもジョブ情報が取得できます。取得した情報からジョブの状態を確認します。

実行権限

この API の実行に必要なロールはありません。対象となるストレージシステムに認証が通るユーザで発行します。

リクエストライン

```
GET ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞/jobs/＜オブジェクト ID＞
```

リクエストメッセージ

オブジェクト ID

非同期 API のレスポンスメッセージまたはジョブ一覧から取得した jobId を指定します。

属性	型	説明
jobId	long	（必須）ジョブのオブジェクト ID

クエリパラメータ

なし。

ボディ

なし。

レスポンスメッセージ

ボディ

```
{
  "jobId": 1,
  "self": "/ConfigurationManager/v1/objects/storages/934000123456/jobs/1
",
  "userId": "rest-test",
  "status": "Completed",
  "state": "Succeeded",
  "createdTime": "2021-04-04T05:06:01Z",
  "updatedTime": "2021-04-04T05:06:05Z",
  "completedTime": "2021-04-04T05:06:05Z",
  "request": {
    "requestUrl": "/ConfigurationManager/v1/objects/storages/93400012345
6/remote-storages",
    "requestMethod": "POST",
    "requestBody": "{\"storageDeviceId\":\"934000123457\",\"restServerIp
\":\"192.0.2.101\",\"restServerPort\":443,\"isMutualDiscovery\":false}"
  },
  "affectedResources": [
    "/ConfigurationManager/v1/objects/storages/934000123456/remote-stora
ges/934000123457"
  ]
}
```

ジョブオブジェクトのスキーマについては、ジョブオブジェクトの説明を参照してください。

ステータスコード

この操作のリクエストに対するステータスコードの意味を次に示します。そのほかのステータスコードについては、HTTP ステータスコードの説明を参照してください。

ステータスコード	メッセージ	説明
404	Not Found	• 指定したジョブ ID に該当する情報がない • 指定したジョブ ID に該当するジョブは API 発行ユーザが投入したものではない

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -H
"Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X GET https:/
/192.0.2.100:23451/ConfigurationManager/v1/objects/storages/836000123456/jo
bs/3
```

関連リンク

HTTP ステータスコード（43 ページ）

ジョブオブジェクト（48 ページ）

第 4 章
ストレージシステムの情報検索

この章では、REST API で実行するストレージシステムのリソースの情報検索について説明します。情報検索は、同時に 1 台のストレージシステムに対して実行できます。

4.1　REST API サーバのデータベースを最新にする方法

REST API サーバのデータベースを最新にする方法には、次の 2 つの方法があります。

- ストレージシステムの構成変更通知を利用したデータベースの更新

  ストレージシステムの構成が変更されたときは、ストレージシステムから REST API サーバに構成変更が通知されます。通知された情報をもとに、REST API サーバのデータベースが自動的に更新されます。

  対象のストレージシステムについて、構成変更の通知を受信できるかどうかを、ストレージシステムの構成変更の通知先の一覧を取得して確認してください。ストレージシステムの構成変更の通知先が登録されていない場合は、ストレージシステムの構成変更の通知先を登録する API を実行してください。

- ストレージシステムの構成情報を更新する API を利用したデータベースの更新

  ストレージシステムの構成情報を更新する API を実行すると、対象となるストレージシステムから構成情報を取得して、REST API サーバに保持しているストレージシステムの構成情報を更新できます。

メモ

- 保守員によるストレージシステムの構成変更が行われた場合、REST API サーバのデータベースに構成変更の情報が反映されません。情報検索の API を実行する前に、ストレージシステムの構成情報を更新する API を実行してください。
- 構成変更の通知を受信していても、次のようなリソースの状態を取得する場合、REST API サーバのデータベースの情報がストレージシステムの最新の情報と一致しないことがあります。情報検索の API を実行する前に、ストレージシステムの構成情報を更新する API を実行してください。
  - 容量や使用率など、リソースの使用状況によって値が変動する情報
  - プールの状態など、ユーザの操作に関わらず自動的に変動する情報

関連リンク

ストレージシステムの構成情報を更新する（82 ページ）
ストレージシステムの構成変更の通知先の一覧を取得する（96 ページ）

ストレージシステムの構成変更の通知先を登録する（100 ページ）

4.2　ストレージシステムの構成情報の更新

REST API サーバがデータベースに保持しているストレージシステムの構成情報を更新したり、更新状態を確認したりする操作について説明します。

4.2.1　ストレージシステムの構成情報の更新状態を取得する

REST API サーバで保持しているストレージシステムの構成情報の更新状態を取得します。取得した `status` 属性や `lastSucceededTime` 属性の結果から構成情報の更新が必要かどうかを確認できます。

実行権限

この API の実行に必要なロールはありません。

リクエストライン

```
GET ＜ベース URL＞/v1/views/refresh-statuses
```

リクエストメッセージ

クエリパラメータ

指定できるクエリパラメータについては、情報検索で使用するクエリパラメータの説明を参照してください。

ボディ

なし。

レスポンスメッセージ

ボディ

```
{
  "data" : [ {
    "refreshStatus" : {
      "storageDeviceId" : "934000010051",
      "status" : "Succeeded",
      "lastSucceededTime" : "2017-12-25T01:21:03Z",
      "lastStartedTime" : "2017-12-25T01:18:00Z"
```

```
    }
  }, {
    "refreshStatus" : {
      "storageDeviceId" : "934000010057",
      "status" : "Succeeded",
      "lastSucceededTime" : "2017-12-25T01:24:48Z",
      "lastStartedTime" : "2017-12-25T01:22:32Z"
    }
  } ],
  "offset" : 0,
  "count" : 2,
  "totalCount" : 2
}
```

属性	型	説明
refreshStatus	object	構成情報の更新ステータス情報 • storageDeviceId（string） ストレージデバイス ID • status（string） 構成情報の更新状態 - `Queued`：構成情報の更新処理がキューイングされた状態 - `Started`:構成情報の更新処理が開始された状態 - `Succeeded`：構成情報の更新処理が成功した状態 - `Failed`：構成情報の更新処理が失敗した状態、または構成情報の更新が必要な状態※1 - `PartiallyNotUpdated`：構成情報の更新処理が一部失敗した状態※2 • lastSucceededTime（ISO8601string） 構成情報の更新処理が成功した最終時刻 • lastStartedTime（ISO8601string） 構成情報の更新処理が開始した最終時刻※3 • error（object） エラーの情報を保持するオブジェクト `status` 属性の値が `Failed` や `PartiallyNotUpdated` の場合に取得できます。

注※1　アップグレードインストールによって REST API のデータベースが拡張された場合は、ストレージシステムの構成情報を更新する必要があります。ストレージシステムの構成情報を更新する API を実行してください。

注※2　ストレージシステムの構成変更の通知を受信したり、REST API でストレージシステムの構成を変更したりしたタイミングで、構成情報の更新処理が失敗した場合に表示されます。

注※3　すでに REST API サーバで構成情報の更新が実行中のストレージシステムに対して、ストレージシステムの構成情報を更新する API を実行すると、リクエストが無視

されます。この場合、lastStartedTime 属性の値はストレージシステムの構成情報を更新する API を実行した時刻より前になるときがあります。

デフォルトソートキー

refreshStatus の storageDeviceId 属性

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Authorization:Session b74777a3-f9f
0-4ea8-bd8f-09847fac48d3" -X GET https://192.0.2.100:23451/ConfigurationMan
ager/v1/views/refresh-statuses
```

関連リンク

HTTP ステータスコード（43 ページ）

4.2.2　ストレージシステムの構成情報を更新する

対象となるストレージシステムから構成情報を取得して、REST API サーバに保持しているストレージシステムの構成情報を更新します。

ヒント

次のような場合には、この API を利用してストレージシステムの構成情報を更新してください。

- マイクロコードを更新した場合（新しい構成情報を取得できる場合があります）
- REST API をアップグレードした場合（REST API サーバが更新されるので、新しい構成情報を取得できる場合があります）

実行権限

セキュリティ管理者（参照）またはセキュリティ管理者（参照・編集）

リクエストライン

```
PUT ＜ベース URL＞/v1/views/actions/refresh/invoke
```

この API は POST メソッドでも実行できます。

リクエストメッセージ

クエリパラメータ

なし。

ボディ

```
{
  "parameters": {
    "storageDeviceId": "934000010051"
  }
}
```

属性	型	説明
storageDeviceId	string	（必須）ストレージデバイス ID 対象となるストレージシステムのストレージデバイス ID を 1 つだけ指定します。

レスポンスメッセージ

ボディ

```
{
  "refreshStatus" : {
    "storageDeviceId" : "934000010051",
    "status" : "Started",
    "lastSucceededTime" : "2016-03-23T04:45:03Z",
    "lastStartedTime" : "2016-03-23T07:21:07Z"
  }

}
```

属性	型	説明
refreshStatus	object	構成情報の更新ステータス情報 • storageDeviceId（string） ストレージデバイス ID • status（string） 構成情報の更新状態 - `Queued`：構成情報の更新処理がキューイングされた状態 - `Started`:構成情報の更新処理が開始された状態 - `Succeeded`：構成情報の更新処理が成功した状態 - `Failed`：構成情報の更新処理が失敗した状態 - `PartiallyNotUpdated`：構成情報の更新処理が一部失敗した状態※1 • lastSucceededTime（ISO8601string）

属性	型	説明
		構成情報の更新処理が成功した最終時刻 • lastStartedTime（ISO8601string） 構成情報の更新処理が開始した最終時刻※2 • error（object） エラーの情報を保持するオブジェクト `status` 属性が `Failed` や `PartiallyNotUpdated` の場合に取得できます。

注※1　ストレージシステムの構成変更の通知を受信したり、REST API でストレージシステムの構成を変更したりしたタイミングで、構成情報の更新処理が失敗した場合に表示されます。

注※2　すでに REST API サーバで構成情報の更新が実行中のストレージシステムに対して、ストレージシステムの構成情報を更新する API を実行すると、リクエストが無視されます。この場合 `lastStartedTime` 属性の値はストレージシステムの構成情報を更新する API を実行した時刻より前になるときがあります。

Action テンプレート

なし。

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept: application/json" -H "Content-Type:application/json" -H
 "Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X PUT --data
-binary @./InputParameters.json https://192.0.2.100:23451/ConfigurationMana
ger/v1/views/actions/refresh/invoke
```

関連リンク

HTTP ステータスコード（43 ページ）

付録 A. REST API サーバの通信モードの変更

REST API の処理速度は、REST API サーバとストレージシステム間の接続方法に依存します。ここでは、REST API サーバとストレージシステム間の接続方法を変更し、REST API サーバの通信モードを変更することによって、REST API の処理速度を向上する方法について説明します。

A.1 REST API サーバの通信モードの変更とは

REST API サーバとストレージシステム間の接続方法を変更することによって、処理速度を上げることができます。接続方法を変更する場合は、REST API の通信モードを変更する必要があります。REST API がサポートしている通信モードについて説明します。

REST API の運用を開始する時点では、REST API サーバが配置されている管理サーバとストレージシステムは、LAN で接続されています。このデフォルトの接続方法の場合の通信モードを lanConnectionMode と呼びます。

処理性能を向上したい場合、ファイバーチャネルまたは iSCSI によってストレージシステムに直接接続したサーバから API を実行する通信方式に変更します。この方式を In-Band 方式と呼びます。REST API を In-Band 方式で実行するには、REST API サーバの通信モードを変更する必要があります。

- fcConnectionMode

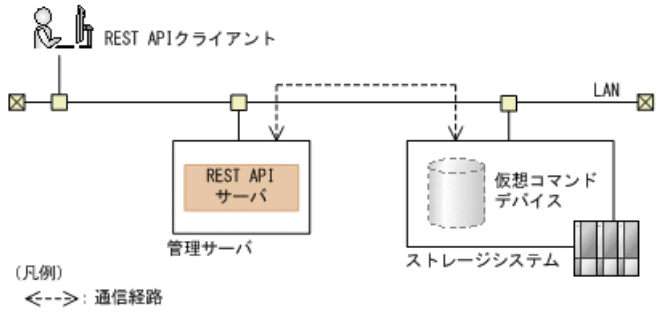
  REST API サーバが配置されている管理サーバとストレージシステムをファイバーチャネル接続または iSCSI 接続する場合の通信モードです。

通信モードを変更するには、ストレージシステムやサーバに必要な設定をしてから、通信方式を変更する API を実行してください。

次に、それぞれの通信モードについて説明します。

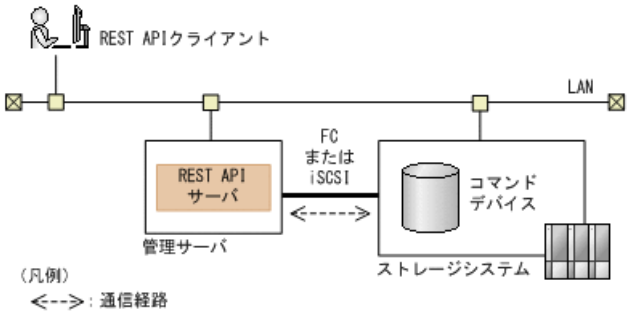
lanConnectionMode

デフォルトの通信モードです。REST API サーバから発行された命令は、ストレージシステム上の仮想コマンドデバイスを経由して実行されます。lanConnectionMode の場合の通信経路の例を次に示します。



fcConnectionMode

REST API サーバが配置されている管理サーバとストレージシステムを、ファイバーチャネル接続または iSCSI 接続する場合の通信モードです。fcConnectionMode の場合の通信経路の例を次に示します。



A.2 REST API サーバの通信モードを変更するための設定

REST API サーバの通信モードを変更するために、あらかじめ必要な設定について説明します。

fcConnectionMode に変更する場合に必要な設定

管理サーバとストレージシステムをファイバーチャネル接続または iSCSI 接続したあと、次の手順でストレージシステム上にコマンドデバイスを作成して、管理サーバ に割り当てます。

1. コマンドデバイスを作成します。次の条件で作成してください。

   条件に従わない場合、REST API の操作が期待した結果を得られないことがあります。

   - ユーザ認証の設定：有効
   - セキュリティの設定：無効
   - リソースグループ：meta_resource

- デバイスグループの情報認証の設定：無効

REST API がインストールされた管理サーバに割り当てられたコマンドデバイスが複数ある場合、そのすべてのコマンドデバイスに必要な設定を行わないと、fcConnectionMode での API の実行に失敗する可能性があります。

2. Storage Navigator で、作成したコマンドデバイスに管理サーバへのパスを割り当てます。

   コマンドデバイスにパスを割り当てる方法については、マニュアル『システム構築ガイド』を参照してください。

3. iSCSI 接続の場合は、管理サーバで iSCSI イニシエータを設定します
4. 管理サーバで、コマンドデバイスが割り当てられていることを確認します。

コマンドデバイスを割り当てたあとにコマンドデバイスの設定を変更した場合は、REST API のサービスを再起動してください。

A.3 REST API サーバの通信モードを変更する

ストレージシステムやサーバに必要な設定をしたあとで、REST API サーバの通信モードを変更します。

実行権限

ストレージ管理者（初期設定）

リクエストライン

```
PUT ＜ベース URL＞/v1/＜ストレージデバイス ID＞/services/communication-mode/actions/change/invoke
```

この API は POST メソッドでも実行できます。

リクエストメッセージ

オブジェクト ID

なし。

クエリパラメータ

なし。

ボディ

```
{
  "parameters": {
    "communicationModes": [
      {
        "communicationMode":"fcConnectionMode"
      }
    ]
  }
}
```

属性	型	説明
communicationModes	object[]	（必須）通信モードの配列 communicationMode に、fcConnectionMode または lanConnectionMode のどちらか 1 つを指定できます。各通信モードには次の属性を指定します。 • communicationMode（string） （必須）通信モード fcConnectionMode、lanConnectionMode のどれかを指定します。

レスポンスメッセージ

ボディ

ジョブオブジェクトを返します。affectedResources 以外の属性については、ジョブオブジェクトの説明を参照してください。

属性	説明
affectedResources	通信モードを変更したストレージシステムの URL

Action テンプレート

なし。

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept: application/json" -H "Content-Type:application/json" -H
 "Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X PUT --data
-binary @./InputParameters.json https://192.0.2.100:23451/ConfigurationMana
ger/v1/836000123456/services/communication-mode/actions/change/invoke
```

付録 B. バックアップとリストア

REST API のデータベースおよび環境設定ファイルのバックアップ、リストアについて説明します。

B.1 REST API のデータベースおよび環境設定ファイルをバックアップする

REST API のデータベースおよび環境設定ファイルのバックアップ方法について説明します。

ディスク障害が発生した場合、バックアップしておいたデータベースおよび環境設定ファイルをリストアすることで、バックアップ時の REST API サーバと同じ状態で稼働を継続できます。データベースおよび環境設定ファイルは、定期的にバックアップしてください。

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. REST API のサービスを停止します。
2. 次のファイルを任意の格納先に手動でコピーして、バックアップを取得します。

バックアップが必要なファイル	ファイルの格納場所
REST API のデータベース	**Linux の場合：** ＜*REST API のインストール先*＞/data/db ディレクトリ以下のファイル • restapi.sqlite.db • search.sqlite.db
プロパティファイル※	**Linux の場合：** ＜*REST API のインストール先*＞/data/properties/StartupV.properties
内部通信用のポートの設定ファイル	**Linux の場合：** ＜*REST API のインストール先*＞/data/usercnf/user-api-port.ini
HTTP 通信または HTTPS 通信用のポートの設定ファイル 秘密鍵およびサーバ証明書の設定ファイル	**Linux の場合：** ＜*REST API のインストール先*＞/oss/apache/conf/userextra ディレクトリ以下のファイル • user-httpd-port.conf • user-httpd-ssl.conf • user-httpsd-certificate.conf • user-proxy-path.conf

バックアップが必要なファイル	ファイルの格納場所
内部通信用のポートの設定ファイル	
ユーザが作成・取得した秘密鍵ファイルおよびサーバ証明書ファイル	ユーザが格納した任意のディレクトリ
Web サーバ用の設定ファイル	**Linux の場合：** ＜*REST API のインストール先*＞/oss/apache/conf/httpd.conf
ストレージシステムの構成変更の通知を受信するための設定ファイル	**Linux の場合：** ＜*REST API のインストール先*＞/oss/rabbitmq/etc/rabbitmq ディレクトリ以下のファイル • rabbitmq-env.conf • advanced.config • .erlang.cookie

注※　ファイルが存在しない場合、バックアップは不要です。

3. REST API のサービスを起動します。

関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）

B.2　REST API のデータベースおよび環境設定ファイルをリストアする

REST API のデータベースおよび環境設定ファイルのリストア方法について説明します。

前提条件

- REST API のデータベース（データベースファイルが存在する場合）および環境設定ファイルがバックアップしてあること
- バックアップ元とリストア先のホスト名、IP アドレスおよび OS が同じであること
- バックアップ元とリストア先の REST API のバージョンが同じであること
- 次のユーザで管理サーバにログインしていること
  - root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. REST API のサービスを停止します。
2. バックアップしておいたファイルを、リストア先に上書きします。

   秘密鍵ファイルおよびサーバ証明書ファイルの格納先をリストア時に変更したい場合は、`user-httpsd-certificate.conf` ファイルに記載されている秘密鍵ファイルおよびサーバ証明書ファイルのパスも合わせて変更してください。
3. ストレージシステムの構成変更の通知を利用する場合、次のコマンドを実行して、ストレージシステムの構成変更の通知を利用するための任意の文字列を設定します。

   Linux の場合：

   ＜REST API のインストール先＞`/bin/setChangeNotificationSecret.sh` *＜任意の文字列＞*

   任意の文字列は次の文字を使用して、32 文字以内で設定してください。

   `A～Z a～z 0～9 - _`
4. REST API のサービスを起動します。
5. ストレージシステムの構成変更の通知を利用する場合、登録されている通知先の情報を更新するため、ストレージシステムの構成変更の通知先を削除してから登録し直します。

   通知先が登録されていないときは、リストア対象の REST API サーバの情報をストレージシステムの構成変更の通知先に登録してください。

関連リンク

REST API のサービスの起動（30 ページ）
REST API のサービスの停止（31 ページ）
ストレージシステムの構成変更の通知先の一覧を取得する（96 ページ）
ストレージシステムの構成変更の通知先を登録する（100 ページ）
ストレージシステムの構成変更の通知先を削除する（102 ページ）

付録 C. トラブルシューティング

REST API サーバで障害が発生した場合の対処方法について説明します。メッセージまたはログファイルを参照して、障害の要因を特定し、対処してください。

C.1 障害発生時に採取が必要な情報

障害要因を特定できない場合や、障害を回復できない場合には、次の情報を採取して、障害対応窓口に連絡してください。

- REST API の保守情報（必須）

  RestTI コマンドを実行して採取してください。

- ストレージシステムにある REST API インターフェースのログ

  ストレージシステムのダンプファイルを採取してください。

  ストレージシステムのダンプファイルの採取方法については、マニュアル『システム管理者ガイド』または『HA Device Manager - Storage Navigator ユーザガイド』を参照してください。

関連リンク

REST API の保守情報を取得する（92 ページ）

C.2 REST API の保守情報を取得する

RestTI コマンドを実行して、管理サーバの保守情報や REST API の保守情報を取得します。

⚠ 注意

保守情報の取得中に表示されるダイアログは閉じないでください。

前提条件

次のユーザで管理サーバにログインしていること

- root ユーザ（Linux の root ユーザでインストールした場合）

操作手順

1. 次のコマンドを実行します。

Linux の場合：

＜REST API のインストール先＞/SupportTools/CollectTool/RestTI.sh -dir *＜保守情報の格納先＞*

オプション

dir

採取した保守情報を格納するディレクトリを絶対パスで指定します。

パスには一部の特殊文字を除いた ASCII 印字可能文字コードを指定できます。指定できない特殊文字を次に示します。Linux の場合は、パス中に空白は指定できません。

\ / : , ; * ? " < > | $ % & ' `

ただし、パスの区切り文字として、Linux の場合はスラッシュ（/）を使用できます。

操作結果

採取した保守情報は、ConfManager_log.jar ファイルに出力されます。

付録 D. ストレージシステムの構成変更の通知

ここでは、ストレージシステムの構成変更を通知する機能の概要と、その機能を利用して REST API のデータベースを更新する方法について説明します。

D.1 ストレージシステムの構成変更の通知とは

ストレージシステムの構成変更の通知の仕組みと、通知を利用して Configuration Manager REST API のデータベースを自動的に更新する方法について説明します。

構成変更の通知の概要

ストレージシステムの構成変更の通知とは、Storage Navigator や REST API などからストレージシステムの構成が変更された場合に、構成変更が発生したことをストレージシステムから管理サーバに通知する機能です。Configuration Manager REST API サーバは、受信した変更通知を基に、対象のリソースの情報をストレージシステムから取得して、情報検索で使用する REST API のデータベースを自動的に更新します。ユーザがストレージシステムの構成情報を更新する API を手動で実行しなくても、Configuration Manager REST API のデータベースを最新の状態に保てます。

構成変更の通知を受信するための設定

構成変更の通知を受信するには、管理サーバが構成変更の通知先としてストレージシステムに登録されている必要があります。

構成変更の通知先は、ストレージシステムを登録する際に同時に登録されます。あとから構成変更の通知先を登録することもできます。

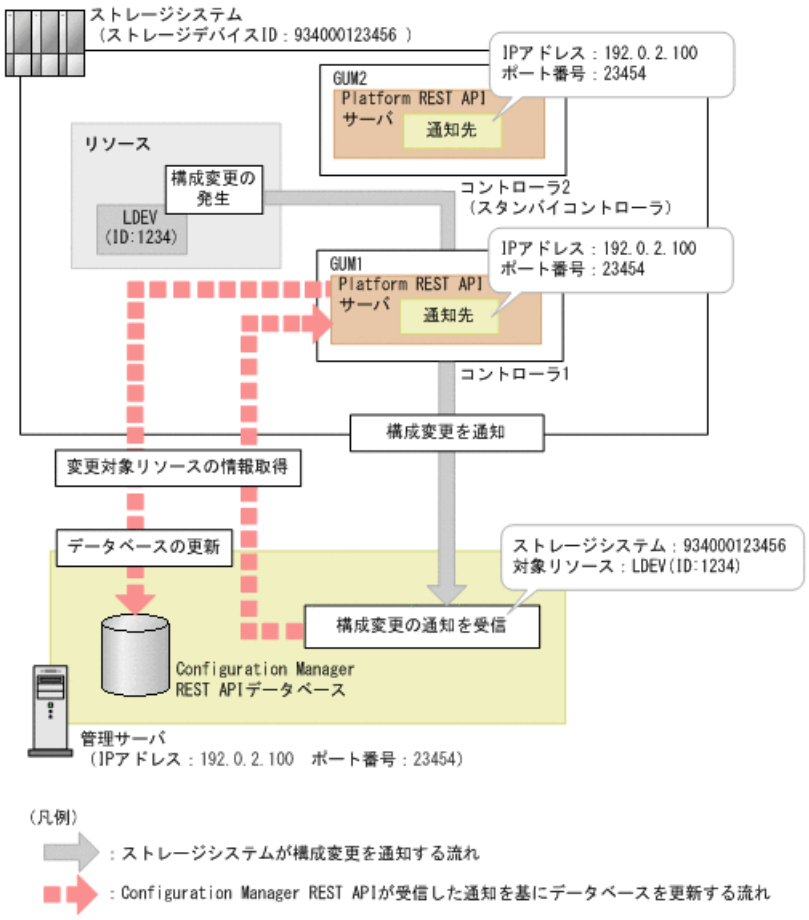
構成変更の通知先

構成変更の通知先情報はコントローラ 1 側の GUM 上とコントローラ 2 側の GUM 上の両方に同時に登録されます。通知先の上限数は、1 コントローラ当たり 4 件です。

D.1.1 構成変更の通知の処理の流れ

ストレージシステムの構成変更の通知を受信して、Configuration Manager REST API のデータベースを更新する流れを説明します。

構成変更の通知の処理の流れの例を次に示します。



ストレージシステムの構成変更の通知先情報はコントローラ 1 側とコントローラ 2 側の両方に登録されています。ストレージシステムの構成変更の通知は、コントローラ 1 側またはコントローラ 2 側のどちらか一方から通知されます。図の例では、コントローラ 1 側から構成変更が通知されています。Configuration Manager REST API サーバは、受信した変更通知を基に、対象のリソースの情報を Platform REST API サーバから取得して、情報検索で使用する REST API のデータベースを自動的に更新します。

メモ

コントローラ 1 側の通知先情報またはコントローラ 2 側の通知先情報のどちらか一方しか登録されていないと、Configuration Manager REST API では、構成変更の通知を正しく受信できません。

構成変更の通知先の登録

ストレージシステムの構成変更の通知を受信するためには、コントローラ 1 側とコントローラ 2 側の両方の通知先をストレージシステムに登録する必要があります。ストレージシステムを登録する API またはストレージシステムの構成変更の通知先を登録する API を実行すると、コントローラ 1 側とコントローラ 2 側の構成変更の通知先情報がそれぞれ別のオブジェクト ID で一度に登録されます。

ヒント

構成変更の通知先を登録したあとで、Configuration Manager REST API サーバの IP アドレスやポート番号が変更された場合は、通知先を再度登録してください。

構成変更の通知先の削除

ストレージシステムの構成変更の通知が不要になった場合は、ストレージシステムに登録した構成変更の通知先を削除して、ストレージシステムから通知が送信されないようにします。

ストレージシステムの情報を削除する API を実行すると、コントローラ 1 側とコントローラ 2 側の構成変更の通知先も同時に削除されるため、そのあとに通知先を削除する API を実行する必要はありません。

通知先を削除する API を実行する場合は、コントローラ 1 側の通知先とコントローラ 2 側の通知先をそれぞれ別のリクエストで削除する必要があります。

メモ

ストレージシステムの情報を削除する API を実行しないで次の操作をした場合は、ストレージシステムに通知先として登録された REST API サーバの情報が削除されずに残るため、通知先を削除する API を実行する必要があります。通知先を削除する API で、コントローラ 1 側の通知先とコントローラ 2 側の通知先をそれぞれ削除してください。

- ストレージシステムの設定を変更したり、ストレージシステムを撤去したりした場合
- Configuration Manager REST API をアンインストールした場合

D.2 ストレージシステムの構成変更の通知先の一覧を取得する

ストレージシステムに登録されている構成変更の通知先の一覧を取得します。運用中の Configuration Manager REST API サーバが通知先として登録されているかどうかを、IP アドレスとポート番号から確認できます。

メモ

- コントローラ 1 側とコントローラ 2 側の両方の構成変更の通知先情報を取得できます。

実行権限

ストレージ管理者（参照）

リクエストライン

```
GET ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞/change-notification-settings
```

リクエストメッセージ

オブジェクト ID

なし。

クエリパラメータ

なし。

ボディ

なし。

レスポンスメッセージ

ボディ

出力例を次に示します。

```
{
  "data" : [ {
    "receiverId" : "changeNotification-ffbac078-8cf0-483f-817d-184a5e812621",
    "receiverIp" : "192.0.2.100",
    "receiverPort" : 23454,
    "ctl" : "CTL1"
  }, {
    "receiverId" : "changeNotification-ffbac078-8cf0-483f-817d-184a5e812623",
    "receiverIp" : "192.0.2.101",
    "receiverPort" : 23454,
    "ctl" : "CTL1"
  }, {
    "receiverId" : "changeNotification-ffbac078-8cf0-483f-817d-184a5e812622",
    "receiverIp" : "192.0.2.100",
    "receiverPort" : 23454,
    "ctl" : "CTL2"
  }, {
    "receiverId" : "changeNotification-ffbac078-8cf0-483f-817d-184a5e812624",
    "receiverIp" : "192.0.2.101",
    "receiverPort" : 23454,
    "ctl" : "CTL2"
  } ]
}
```

属性	型	説明
receiverId	string	構成変更の通知先情報のオブジェクト ID
receiverIp	string	構成変更の通知先の IP アドレス
receiverPort	int	構成変更の通知先のポート番号
ctl	string	構成変更の通知先情報が登録されているコントローラ • CTL1：コントローラ 1 • CTL2：コントローラ 2

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -H
"Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X GET https:/
/192.0.2.100:23451/ConfigurationManager/v1/objects/storages/834000123456/ch
ange-notification-settings
```

関連リンク

HTTP ステータスコード（43 ページ）

D.3 ストレージシステムの構成変更の特定の通知先を取得する

ストレージシステムに登録されている構成変更の通知先の情報を、通知先のオブジェクト ID を指定して取得します。指定したオブジェクト ID の通知先が存在するかどうかを確認できます。

メモ

- コントローラ 1 側の構成変更の通知先の情報とコントローラ 2 側の構成変更の通知先の情報は、それぞれ別のリクエストで取得する必要があります。

実行権限

ストレージ管理者（参照）

リクエストライン

```
GET ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞/change-notification-settings/＜オブジェクト ID＞
```

リクエストメッセージ

オブジェクト ID

ストレージシステムの構成変更の通知先の一覧取得で取得した receiverId の値を指定します。

属性	型	説明
receiverId	string	（必須）構成変更の通知先情報のオブジェクト ID

クエリパラメータ

なし。

ボディ

なし。

レスポンスメッセージ

ボディ

出力例を次に示します。

```
{
  "receiverId" : "changeNotification-ffbac078-8cf0-483f-817d-184a5e812621",
  "receiverIp" : "192.0.2.100",
  "receiverPort" : 23454,
  "ctl" : "CTL1"
}
```

属性	型	説明
receiverId	string	構成変更の通知先情報のオブジェクト ID
receiverIp	string	構成変更の通知先の IP アドレス
receiverPort	int	構成変更の通知先のポート番号
ctl	string	構成変更の通知先情報が登録されているコントローラ • CTL1：コントローラ 1 • CTL2：コントローラ 2

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -H
"Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X GET https:/
/192.0.2.100:23451/ConfigurationManager/v1/objects/storages/834000123456/ch
ange-notification-settings/changeNotification-ffbac078-8cf0-483f-817d-184a5
e812615
```

関連リンク

HTTP ステータスコード（43 ページ）
ストレージシステムの構成変更の通知先の一覧を取得する（96 ページ）

D.4 ストレージシステムの構成変更の通知先を登録する

Configuration Manager REST API サーバに登録済みのストレージシステムに、構成変更の通知先を登録します。

メモ

- コントローラ 1 側とコントローラ 2 側の構成変更の通知先情報がそれぞれ別のオブジェクト ID で一度に登録されます。

ヒント

すでに通知先が登録されている場合に、Configuration Manager REST API サーバの IP アドレスやポート番号が変更されたときは、通知先を再度登録してください。

実行権限

セキュリティ管理者（参照）またはセキュリティ管理者（参照・編集）

リクエストライン

```
POST ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞/change-notifica
tion-settings
```

リクエストメッセージ

オブジェクト ID

なし。

クエリパラメータ

なし。

ボディ

なし。

レスポンスメッセージ

ボディ

出力例を次に示します。

```
{
  "receiverId" : "changeNotification-ffbac078-8cf0-483f-817d-184a5e812621",
  "standbyReceiverId" : "changeNotification-ffbac078-8cf0-483f-817d-184a5e812622",
  "ctl" : "CTL1"
}
```

属性	型	説明
receiverId	string	構成変更の通知先情報のオブジェクト ID
standbyReceiverId	string	構成変更の通知先情報のオブジェクト ID（スタンバイコントローラ側）
ctl	string	`receiverId` 属性の値がどちらのコントローラの情報であるかを示します。 • `CTL1`：コントローラ 1 • `CTL2`：コントローラ 2

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -H "Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X POST --data-binary @./InputParameters.json https://192.0.2.100:23451/ConfigurationManager/v1/objects/storages/834000123456/change-notification-settings
```

関連リンク

HTTP ステータスコード（43 ページ）
ストレージシステムの情報を変更する（62 ページ）
ストレージシステムの構成変更の通知先の一覧を取得する（96 ページ）

D.5 ストレージシステムの構成変更の通知先を削除する

ストレージシステムに登録した構成変更の通知先を削除します。

メモ

- コントローラ 1 側の構成変更の通知先の情報とコントローラ 2 側の構成変更の通知先の情報は、それぞれ別のリクエストで削除する必要があります。

実行権限

セキュリティ管理者（参照）またはセキュリティ管理者（参照・編集）

リクエストライン

```
DELETE ＜ベース URL＞/v1/objects/storages/＜ストレージデバイス ID＞/change-notification-settings/＜オブジェクト ID＞
```

リクエストメッセージ

オブジェクト ID

ストレージシステムの構成変更の通知先の一覧取得で取得した `receiverId` の値を指定します。

属性	型	説明
receiverId	string	（必須）構成変更の通知先情報のオブジェクト ID

クエリパラメータ

なし。

ボディ

なし。

レスポンスメッセージ

ボディ

```
{
  "receiverId": "changeNotification-ffbac078-8cf0-483f-817d-184a5e812616
"
}
```

属性	型	説明
receiverId	string	削除した通知先情報のオブジェクト ID

ステータスコード

この操作のリクエストに対するステータスコードについては、HTTP ステータスコードの説明を参照してください。

コード例

```
curl -v -H "Accept:application/json" -H "Content-Type:application/json" -H
"Authorization:Session b74777a3-f9f0-4ea8-bd8f-09847fac48d3" -X DELETE http
s://192.0.2.100:23451/ConfigurationManager/v1/objects/storages/834000123456
/change-notification-settings/changeNotification-ffbac078-8cf0-483f-817d-18
4a5e812615
```

関連リンク

HTTP ステータスコード（43 ページ）
ストレージシステムの構成変更の通知先の一覧を取得する（96 ページ）

付録 E.　Configuration Manager のバージョン

対象製品のバージョン、REST API のバージョン、およびストレージシステムのマイクロコードのバージョンの対応について説明します。

E.1　Configuration Manager バージョン対応表

対象製品のバージョンと REST API のバージョン、および対象製品でサポートした機能をすべて使用するために必要なストレージシステムのマイクロコードのバージョンの対応を次の表に示します。なお、ご利用のマイクロコードが次の表に示すより古いバージョンの場合、REST API の一部の機能が使用できません。

対象製品のバージョン	Configuration Manager REST API のバージョン※1	ストレージシステムのマイクロコードのバージョン
Configuration Manager 11.0.4	1.57.0	**iStorage V シリーズ** 93-07-27-XX
Configuration Manager 11.0.3	1.54.0	**iStorage V シリーズ** 93-07-25-XX
Configuration Manager 11.0.0	1.40.0	**iStorage V シリーズ** 93-07-23-XX
Configuration Manager 10.9.2	1.36.0	**iStorage V シリーズ** 93-06-81-XX
Configuration Manager 10.9.1	1.34.0	**iStorage V シリーズ** 93-06-81-XX
Configuration Manager 10.9.0	1.33.0	**iStorage V シリーズ** 93-06-42-XX
Configuration Manager 10.7.0	1.27.0	**iStorage V シリーズ** 93-06-21-XX

注※1　REST API のバージョンを確認するには、バージョン情報を取得する API を実行してください。

関連リンク

バージョン情報を取得する（53 ページ）

付録 F.　リトライ処理の組み込み

REST API を使用したスクリプトにリトライ処理を実装する上で、考慮すべき点について説明します。

F.1　リトライ処理の組み込み

REST API を使用してクライアントプログラムを作成する場合に、意図したとおりにプログラムが実行できるよう、リトライ処理を適切に組み込んでください。そのあとに、本番環境に近いシステム構成で、クライアントプログラムを検証することをお勧めします。

リトライの条件

ネットワークやサーバへの一時的な負荷の集中が原因で、REST API の実行に失敗することがあります。このような場合、失敗したリクエストをリトライする処理を実装することで、処理を継続できる可能性があります。リトライで対処できるエラーかどうかは、HTTP ステータスコード、およびエラーオブジェクトなどのレスポンスから判定できます。リトライで対処ができるエラーの条件を説明します。

次のどれかの場合にリトライすることができます。

- HTTP ステータスコードに 503（Service unavailable）が返る
- ジョブが失敗したときに、エラーオブジェクトの solutionType に RETRY が返る
- REST API を長時間使用しているときに以下のレスポンスが返る

  1. HTTP ステータスコードが 500 で、HTML 形式のレスポンスボディが返る

     この現象は、1 分～2 分ほど続きます。

  2. セッションを破棄する API 以外の実行時に HTTP ステータスが 200 で、空のレスポンスボディが返る

     この後、1.の現象が発生します。

  この場合、HTTP ステータスが 500 で HTML 形式のレスポンスが返らなくなるまで API を再実行してください。

- エラーオブジェクトに、特定のメッセージ ID やストレージシステムのエラーコードが返る

  リトライはエラーオブジェクトの内容で判定します。エラーオブジェクトは、次の方法で取得できます。

  - API リクエスト発行時のレスポンス
  - ジョブの情報を取得する API のレスポンス

エラーオブジェクト	メッセージ ID	ストレージシステムのエラーコード		
		SSB1	SSB2	errorCode
API リクエスト発行時のレスポンス	KART00003-E KART00006-E KART30003-E KART30090-E KART30095-E KART30096-E KART30097-E KART40042-E KART40049-E KART40051-E KART40052-E	-	-	-
ジョブの情報を取得する API のレスポンス	KART30000-E	2E11	2205	-
	KART30008-E KART30072-E	-	-	-

（凡例）

-：該当なし

クライアントプログラムの検証

本番環境に近いシステム構成でクライアントプログラムの検証をします。

次の観点で検証してください。

- リトライ条件を組み込んだクライアントプログラムが動作するか。
- クライアントプログラムが期待している実行時間内に完了するか。

検証結果が期待通りではなかった場合、下記の観点でプログラムを見直してください。

- クライアントの API のリトライ回数、および、リトライ時間を調整してください。
- API の同時実行数を減らして、クライアントの API のリトライ頻度を低減してください。

関連リンク

HTTP ステータスコード（43 ページ）
エラーオブジェクト（49 ページ）
ジョブの情報の一覧を取得する（73 ページ）

F.2 リトライ処理のコード例

API 実行時に、リトライ処理を組み込むためのコード例について説明します。

リトライ処理のコード例

API（function_xxx）の実行に失敗した場合に、返却されるステータスコードやエラーオブジェクトのエラーメッセージがリトライ条件に該当するかどうかを判定します。このサンプルコードでは、HTTP ステータスコードに 503（Service unavailable）が返るときと、エラーオブジェクトのメッセージ ID でリトライを判断する条件のときに、リトライするよう記載しています。リトライ条件に合致した場合は、指定したリトライ上限回数や間隔で API をリトライします。

```
retry_error_list = ["KART30095-E", "KART30096-E", "KART30003-E", "KART30090
-E", ...]
retry_count = 1

from requests.structures import CaseInsensitiveDict

while True:
    url = block_storage_api.function_xxx()
    r = requests.post(url, headers=headers, auth=USER_CREDENTIAL,
                      verify=False)
    if r.status_code == http.client.OK or r.status_code == http.client.ACCE
PTED:
        # Succeeded #
        print("Succeeded")
        break
    else:
        # failed #
        error_code = CaseInsensitiveDict(r.json())["messageId"]
        if r.status_code == http.client.SERVICE_UNAVAILABLE or error_code i
n retry_error_list:
            if retry_count > MAX_RETRY_COUNT:
                raise Exception("Timeout Error! "
                                "Operation was not completed.")
            print("Retry API")
            time.sleep(WAIT_TIME)
            retry_count += 1
        else:
            raise requests.HTTPError(r)
```

Succeeded

API の実行に成功した場合の処理です。

このコード例では、ステータスコードが 200（http.client.OK）と 202（http.client.ACCEPTED）の両方を定義しています。処理を実装する際には、API の処理方式（同期処理と非同期処理）に合わせて定義してください。

failed

API の実行に失敗した場合に、リトライを行う処理です。

ステータスコードが 503（http.client.SERVICE_UNAVAILABLE）、またはエラーメッセージが retry_error_list に定義したメッセージ ID に合致した場合、指定したリトライ上限回数（MAX_RETRY_COUNT）や間隔（WAIT_TIME）で API をリトライします。リトラ

イ上限回数や間隔は、ご利用のシステム設計や運用に合わせて適切に設定してください。

付録 G.　このマニュアルの参考情報

このマニュアルを読むに当たっての参考情報を示します。

G.1　このマニュアルでの表記

このマニュアルでは、製品名を次のように表記しています。

表記	製品名
JDK	Java Development Kit
Storage Navigator	次の製品の表記です。 • HA Device Manager - Storage Navigator
VMware	VMware ®
VMware ESXi	VMware vSphere ® ESXi ™
iStorage V シリーズ	次の製品を区別する必要がない場合の表記です。 • iStorage V100 • iStorage V300

G.2　このマニュアルで使用している略語

このマニュアルで使用する英略語を次に示します。

略語	正式名称
ALU	Administrative Logical Unit
ALUA	Asymmetric Logical Unit Access
API	Application programming interface
CHAP	Challenge Handshake Authentication Protocol
DKC	DisK Controller
CLPR	Cache Logical Partition
CVS	Custom Volume Size
DCR	Dynamic Cache Residency
DP	Dynamic Provisioning
FC	Fibre Channel
FCoE	Fibre Channel over Ethernet
GUM	Gateway for Unified Management
HBA	host bus adapter
DT	Dynamic Tiering
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
I/O	input/output

略語	正式名称
ID	identifier
IOPH	Input Output Per Hour
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
iSCSI	Internet Small Computer System Interface
JSON	JavaScript Object Notation
LDEV	logical device
LU	logical unit
LUN	logical unit number
LUSE	logical unit size expansion
MCU	Main Control Unit
MLC	Multiple Level Cell
OS	operating system
P-VOL	primary volume
PEM	Privacy Enhanced Mail
RAID	Redundant Array of Independent Disks
RCU	Remote Control Unit
REST	Representational State Transfer
RFC	Request for Comments
RI	Read Intensive
SAS	Serial Attached SCSI
SCM	Storage Class Memory
SIM	Service Information Message
SLC	Single Level Cell
SLU	Subsidiary Logical Unit
S-VOL	secondary volume
SSD	Solid-State Drive
SSID	Storage System ID
SSL	Secure Sockets Layer
TLS	Transport Layer Security
URL	Uniform Resource Locator
UUID	Universally Unique Identifier
V-VOL	virtual volume
WWN	World Wide Name

G.3 KB（キロバイト）などの単位表記について

1KB（キロバイト）、1MB（メガバイト）、1GB（ギガバイト）、1TB（テラバイト）、1PB（ペタバイト）は、それぞれ 1KiB（キビバイト）、1MiB（メビバイト）、1GiB（ギビバイト）、1TiB（テビバイト）、1PiB（ペビバイト）と読み替えてください。

1KiB、1MiB、1GiB、1TiB、1PiB は、それぞれ 1,024 バイト、1,024KiB、1,024MiB、1,024GiB、1,024TiB です。

索引

A

Action テンプレートオブジェクト.....51
Authorization ヘッダ..... 37

G

GUM..... 1

H

HTTP ステータスコード.....43
HTTP 通信
　無効化..... 11
　有効化..... 12
HTTP メソッド.....36

J

JSON.....45,46

R

REST API クライアント..... 1
REST API サーバ
　更新状態の取得..... 80
　構成情報の更新..... 82
REST API サービス
　起動..... 30
　停止..... 31
RestTI コマンド.....92

S

SSL 通信.....20,23,62

U

URL の形式.....34

あ

アンインストール
　Linux.....32
　REST API..... 32
インストール
　Linux.....6
　REST API..... 5
　前提プログラム..... 3
インストール先.....5
エラーオブジェクト.....49
オブジェクト ID..... 35

か

クエリパラメータ
　objects ドメイン.....46
構成変更の通知
　通知先の削除..... 94
　通知先の登録..... 94

さ

サーバ証明書.....23
自己署名証明書.....20
システム構成.....1
出力形式.....47
情報検索
　更新状態の取得..... 80
　構成情報の更新..... 82
ジョブ.....48
ジョブオブジェクト.....48
ストレージデバイス ID..... 34
セッション.....40

た

データオブジェクト.....47
データ型.....46
同期処理.....36

は

バックアップ.....89
非同期処理.....36
ベース URL.....34
ポート番号.....8,9
保守情報.....92

ら

リクエストオブジェクト.....51
リクエストヘッダ.....41
リストア.....90
レスポンスヘッダ.....43

iStorage V100/V300
HA Command Suite
Configuration Manager
REST API リファレンスガイド

IV-UG-212-001-08

2025 年 10 月 第 8 版 発行

日本電気株式会社

© NEC Corporation 2021-2025