SOAP API リファレンス ガイド

Arcserve® ハイ アベイラビリティ

バージョン 18.0

法律上の注意

組み込みのヘルプシステムおよび電子的に配布される資料も含めたこのドキュメント(以下「本書」) はお客様への情報提供のみを目的としたもので、Arcserve により随時、変更または撤回されることがあります。

Arcserve の事前の書面による承諾を受けずに本書の全部または一部を複写、譲渡、変更、開示、修正、複製することはできません。本書は Arcserve が知的財産権を有する機密情報であり、ユーザは (i) 本書に関連する Arcserve ソフトウェアの使用について、Arcserve とユーザとの間で別途締結される契約により許可された以外の目的、または (ii) ユーザとArcserveとの間で別途締結された守秘義務により許可された以外の目的で本書を開示したり、本書を使用することはできません。

上記にかかわらず、本書で取り上げているソフトウェア製品 (複数の場合あり) のライセンスを受けたユーザは、そのソフトウェアに関して社内で使用する場合に限り本書の合理的な範囲内の部数のコピーを作成できます。ただし Arcserve のすべての著作権表示およびその説明を各コピーに添付することを条件とします。

本書を印刷するかまたはコピーを作成する上記の権利は、当該ソフトウェアのライセンスが完全に有効となっている期間内に限定されます。いかなる理由であれ、そのライセンスが終了した場合には、ユーザは Arcserve に本書の全部または一部を複製したコピーを Arcserve に返却したか、または破棄したことを文書で証明する責任を負います。

準拠法により認められる限り、Arcserve は本書を現状有姿のまま提供し、商品性、お客様の使用目的に対する適合性、他者の権利に対する不侵害についての黙示の保証を含むいかなる保証もしません。また、本システムの使用に起因して、逸失利益、投資損失、業務の中断、営業権の喪失、情報の損失等、いかなる損害 (直接損害か間接損害かを問いません) が発生しても、Arcserve はお客様または第三者に対し責任を負いません。Arcserve がかかる損害の発生の可能性について事前に明示に通告されていた場合も同様とします。

本書に記載されたソフトウェア製品は、該当するライセンス契約書に従い使用されるものであり、当該ライセンス契約書はこの通知の条件によっていかなる変更も行われません。

本書の制作者は Arcserve です。

「制限された権利」のもとでの提供 : アメリカ合衆国政府が使用、複製、開示する場合は、FAR Sections 12.212, 52.227-14 及び 52.227-19(c)(1) 及び (2)、及び、DFARS Section252.227-7014(b)(3) または、これらの後継の条項に規定される該当する制限に従うものとします。

Arcserve 製品リファレンス

このマニュアルが参照している Arcserve 製品は以下のとおりです。

- Arcserve® High Availability(HA)
- Arcserve® Replication
- Arcserve® Assured Recovery®
- Arcserve® Content Distribution

関連マニュアル

このマニュアルは、「*Arcserve RHA インストール ガイド*」および「*Arcserve RHA 管理者 ガイド*」と併せてお読みください。Arcserve RHA 18.0 マニュアルのリンクについては、以下を参照してください。

- マニュアル選択メニュー
- リリース ノート

Arcserve へのお問い合わせ

Arcserve サポート チームは、技術的な問題の解決に役立つ豊富なリソースを提供します。重要な製品情報に簡単にアクセスできます。

https://www.arcserve.com/support

Arcserve のサポート：

- Arcserve サポートの専門家が社内で共有しているのと同じ情報ライブラリに直接アクセスできます。このサイトから、弊社のナレッジ ベース(KB) ドキュメントにアクセスできます。ここから、重要な問題やよくあるトラブルについて、製品関連 KB 技術情報を簡単に検索し、検証済みのソリューションを見つけることができます。
- 弊社のライブ チャット リンクを使用して、Arcserve サポート チームとすぐにリアルタイムで会話を始めることができます。ライブ チャットでは、製品にアクセスしたまま、懸念事項や質問に対する回答を即座に得ることができます。
- Arcserve グローバル ユーザ コミュニティでは、質疑応答、ヒントの共有、ベストプラクティスに関する議論、他のユーザとの対話に参加できます。
- サポート チケットを開くことができます。オンラインでサポート チケットを開くと、質問の対象製品を専門とする担当者から直接、コールバックを受けられます。

また、使用している Arcserve 製品に適したその他の有用なリソースにアクセスできます。

製品ドキュメントに関するフィードバックの提供

Arcserve 製品ドキュメントに関してコメントまたは質問がある場合は、こちらまでお問い合わせください。

コンテンツ

第1章: 概要

このセクションには、以下のトピックが含まれます。

SOAP API を使用したプログラミング - フロー

SOAP API を使用して Arcserve RHA を制御するには、クライアントは RHA CS に対する認証を実行し、その後セッションを作成する必要があります。この操作は create_session API 関数を呼び出すことにより実行されます。クライアントがセッションを作成した後は、そのクライアントはこのリファレンス ガイドに記載されている任意の API 関数を呼び出すことができます。タスクが完了したら、クライアントは close_session API を呼び出し、セッションを無効にしてリソースを解放します。

SOAP API の概要

このガイドでは、Arcserve RHA によってエクスポートされた SOAP API について説明します。これらの API を使用すると、レプリケーションおよびハイ アベイラビリティをさまざまな点から管理できます。たとえば、RHA シナリオを制御して、シナリオの実行や停止、シナリオ フェールオーバなどのさまざまな操作を実行できます。<carha> SOAP API を使用するには、統合 Web サービス サポートを含むプログラミング言語 (Visual Basic や C# など) を使用します。

注： このリファレンス ガイドで提供されているサンプル コードは、C# で記述されています。

サンプル コードで使用される SOAP API URL は http://127.0.0.1:8088/ws_man/xosoapapi.asmx です。API を呼び出すときは、このアドレス部をお使いの RHA CS の IP アドレスまたはホスト名で置き換えてください。

このリファレンス ガイドで使用されている規則

Arcserve RHA SOAP API リファレンス ガイドは、以下のテーブルに記述されている引数タイプを使用します。

種類	説明
bool	ブール値
out uint	出力整数
out ulong	出力符号なし長整数
out string	出力文字列
uint	符号なし整数
ulong	符号なし長整数
ushort	符号なし短整数

第1章: SOAP API

このセクションには、以下のトピックが含まれます。

シナリオ管理 API

以下のセクションでは、基本的なシナリオ管理タスクを実行するための API について説明します。

- 作業開始：create_session()
- シナリオの作成：create_scenario_ex()
- ルート ディレクトリの設定：set_root_dir()
- ルート ディレクトリの追加：add_root_dir()
- レプリカの追加：add_replica()
- 複数のレプリカの追加：add_replica_ex()
- シナリオのプロパティの設定：set_scenario_data()
- ホスト プロパティの設定：set_host_data()
- シナリオの開始：run()
- シナリオの停止：stop()
- データ レプリケーションの一時停止：suspend_replication()
- 作業終了：close_session()
- 認証情報の管理：add_credentials_ex()
- Arcserve Backup add_bab_credentials() を使用した統合
- シナリオのプロパティの更新：update_scenario()
- シナリオの削除：remove_scenario()
- シナリオのインポート：import_scenario()
- 実行中のシナリオの同期：synchronize()
- レプリケーションの再開：resume_replication()
- リワインド ブックマークの追加：set_rewind_bookmark()

作業開始：create_session()

create_session API 関数を使用すると、コントロール サービスに対する認証を実行でき、セッション ID が返されます。このセッション ID は、他の API 関数の呼び出し時に引数として渡されます。オープン セッションは、close_session API を使用して無効にすることができます。

引数

create_session API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
user_name	string	ユーザ名。 **例：** MyDomain\Administrator
password	string	ユーザ名のパスワード。 例：<ca>
error_code	out uint	ゼロ値は、API 関数が正常に実行されたことを示します。ゼロ以外の値は、API の失敗を示します。

戻り値

create_session を使用した認証が成功した場合、この関数はセッション ID を含む uint 値を返します。また、error_code 引数が 0 に設定されます。それ以外の場合、MAX uint 値 (0xFFFFFFFF) が返され、error_code 引数には詳細なエラーコードが含まれます。

注： user_name 引数は、<DOMAIN_NAME>\<USER_NAME> という形式で指定する必要があります。たとえば、test_domain\Administrator というように指定します。

例

例 1： Web サービス オブジェクトの作成

```
xosoapapi_c get_mng()
{
xosoapapi_c mng = new xosoapapi_c();
return mng;
}
```

例 2： セッションの作成

```
public bool create_session_example()
{
try
{
uint err = 0;
string user_name = global::api_examples.Properties.Settings.Default.user_name;
```

```
string password = global::api_examples.Properties.Settings.Default.password;
_session_id = get_mng().create_session(user_name, password, out err);
if (_session_id == 0xffffffff)
{
return false;
}
return true;
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

シナリオの作成：create_scenario_ex()

create_scenario_ex API 関数を使用すると、シナリオを作成できます。シナリオの作成後、シナリオの実行、シナリオプロパティの変更、ホストの追加または削除などの操作を実行できます。

引数

create_scenario_ex API 関数には、以下のテーブルで説明されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session 関数 API のコールによって返されたセッション ID。
製品	uint	以下の製品の整数値を指定します。 0 - DR シナリオ 1 - HA シナリオ 2 - CD シナリオ
アプリケーション	uint	以下のアプリケーションの整数値を指定します。 0 - ファイルサーバシナリオ 1- Exchange シナリオ 2 - Oracle シナリオ 3 - 使用されていません 4 - SQL Server シナリオ 5 - IIS シナリオ 6 - コントロールサービスシナリオ 7 - Hyper-v シナリオ 8 - Sharepoint シナリオ 9 - vCenter シナリオ 10 - CRM シナリオ 11- フルシステムシナリオ 12 - カスタマイズシナリオ
is_ar	bool	シナリオがアシュアードリカバリ(AR)をサポートするかどうかを指定します。 ■ true：AR をサポートします ■ false：AR をサポートしません
is_cdp	bool	常に false。使用されていません。
integrate_opt	uint	以下のアプリケーションの実整数値を指定します。 0 - 統合なし

		1 - Backup 2 - <ca> D2D 3 - <ca> Central Applications
group_id	uint	グループ ID。group_id が 0xFFFFFFFF である場合、シナリオはデフォルト シナリオ グループに属します。このグループは、通常「シナリオ」と呼ばれます。
scenario_data	out string	シナリオのデータ。この API をコールした後に、引数には更新されたシナリオ データが含まれます。 **注：** 詳細については、このトピックの例を参照してください。
group_data	out string	グループ データ。

戻り値

この API は、作成されたシナリオ ID を返します。run や stop など他の API では、シナリオ ID が必要です。シナリオ ID がゼロの場合、API は失敗しています。それ以外の場合、ゼロ以外の値は、API が正常に完了したことを示します。

注： この API コールが正常に完了すると、シナリオのスケルトンが作成されます。シナリオのプロパティにはすべてデフォルト値があります。ルート ディレクトリ、ホスト IP などの重要なプロパティは空です。プロパティを満たすために、他の API をコールします。この「リファレンス ガイド」では、これ以降の API について、プロパティを満たす方法を説明します。

例

```
uint product = 0;
uint app = 0;
bool is_ass_rec = false;
bool is_cdp = false;
uint si_opt = 0;
uint group_id = 0xFFFFFFFF;
string scenario_data = "";
string group_data = "";
uint scenario_id = get_mng().create_scenario_ex(_session_id, product, app, is_ass_
rec, is_cdp, si_opt, group_id, out scenario_data, out group_data);
```

ルート ディレクトリの設定：set_root_dir()

この set_root_dir() API 関数によって、ルート ディレクトリのパスを編集できます。

引数

set_root_dir API 関数には、以下のテーブルで説明されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session 関数 API のコールによって返されたセッション ID。
scenario_id	uint	シナリオの ID。
host_index	uint	設定するホストのインデックス。
root_dir_index	uint	ルート ディレクトリのインデックス。ゼロから始まり、ルート ディレクトリが 2 つある場合、インデックスは 0 と 1 になります。
root_dir	string	フォルダ名。 **注：** この引数が必要になるのは、フル システム シナリオの場合のみです。ただし、この API 関数は現在、フル システム シナリオでのルート ディレクトリの設定をサポートしていません。

戻り値

API コールが正常に完了すると、この関数は値として true を返します。それ以外の場合は、false が返されます。

注： create_scenario_ex API は、シナリオのスケルトンのみを作成します。この API をコールして、ルート ディレクトリを設定できます。

例

```
uint master_host_index = 1;

get_mng().set_root_dir(_session_id, scenario_id, master_host_index, 0, "E:/test");
```

ルート ディレクトリの追加：add_root_dir()

この add_root_dir API 関数によって、シナリオに新規ルート ディレクトリを追加できます。

引数

add_root_dir API 関数には、以下のテーブルで説明されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session 関数 API のコールによって返されたセッション ID。
scenario_data	ref string	シナリオのデータ。この API をコールした後に、引数には更新されたシナリオ データが含まれます。
root_directories	string	フォルダ名 。
new_root_dir_index	out uint	新規に作成されたルート ディレクトリのインデックス。

戻り値

API コールが正常に完了すると、この関数は値として true を返します。それ以外の場合は、false が返されます。

例：ルート ディレクトリの追加

```
String scenario_data = get_mng().get_scenario_data(_session_id, scenario_id);
//add root directory
get_mng().add_root_dir(_session_id, ref scenario_data, "c:/test", out new_root_dir_id);
```

レプリカの追加：add_replica()

この add_replica API 関数によって、シナリオに新規レプリカを追加できます。マスタデータを複数のレプリカ ホストへレプリケートする場合、この API をコールして、いくつかのレプリカ ホストを追加できます。

引数

この add_replica API 関数には、以下のテーブルで説明されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session 関数 API のコールによって返されたセッション ID。
scenario_data	ref string	シナリオのデータ。この API をコールした後に、引数には更新されたシナリオ データが含まれます。
host_index	uint	親ホスト インデックス。マスタ ホスト インデックスは常に 1 です。最初のレプリカは、通常 2 です。
new_replica_index	out uint	新規に追加されたレプリカ ホストのインデックス

戻り値

API コールが正常に完了すると、この関数は値として true を返します。それ以外の場合は、false が返されます。

例：レプリカの追加

```
//マスタ ホストに 1 つのレプリカを追加します
get_mng().add_replica(_session_id, ref scenario_data, master_host_index, out new_replica_index);
```

複数のレプリカの追加：add_replica_ex()

この add_replica_ex API 関数によって、シナリオに複数の新規レプリカ ホストを一度に追加できます。

引数

この add_replica_ex API 関数には、以下のテーブルで説明されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session 関数 API のコールによって返されたセッション ID。
scenario_data	ref string	シナリオのデータ。この API をコールした後に、引数には更新されたシナリオ データが含まれます。
host_index	uint	親ホスト インデックス。マスタ ホスト インデックスは常に 1 です。最初のレプリカは、通常 2 です。
host_list	ref string	このホスト リストでは、ホストに関する情報が XML 形式で提供されます。この XML ファイルには、シナリオに追加するホストに関する情報が含まれます。 **注：** 以下のテーブルで画面を確認してください。

たとえば、以下の XML コードには 2 つのホストに関する情報が含まれています。

```
<?xml version="1.0"?>
<object>
  - <object>
        <data type="String" val="155.35.76.155" label="Host"/>
        <data type="String" val="155.35.76.155" label="IP"/>
    </object>
  - <object>
        <data type="String" val="155.35.76.156" label="Host"/>
        <data type="String" val="155.35.76.156" label="IP"/>
    </object>
</object>
```

戻り値

API コールが正常に完了すると、この関数は値として true を返します。それ以外の場合は、false が返されます。

例：複数のレプリカの追加

```
string host_list = "<?xml version=\"1.0\"?><object> <object><data label=\"Host\"
val=\"155.35.76.155\" type=\"String\" /><data label=\"IP\" val=\"155.35.76.155\"
type=\"String\" /> </object><object><data label=\"Host\" val=\"155.35.76.156\"
type=\"String\" /><data label=\"IP\" val=\"155.35.76.156\" type=\"String\" />
</object></object>";
//マスタ ホストの下に 2 つのレプリカを追加します
```

```
get_mng().add_replica_ex(_session_id, ref scenario_data, master_host_index, ref
host_list);
```

シナリオのプロパティの設定：set_scenario_data()

この set_scenario_data API 関数によって、シナリオのプロパティを編集できます。シナリオのほとんどのプロパティは、この API によって更新できます。

引数

この set_scenairo_data API 関数には、以下のテーブルで説明されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session 関数 API のコールによって返されたセッション ID。
scenario_id	uint	シナリオの ID。
property_name	string	ホストのプロパティ名。プロパティには、それぞれ一意の名前があります。
property_value	string	プロパティの値。

戻り値

API コールが正常に完了すると、この関数は値として true を返します。それ以外の場合は、false が返されます。

例：シナリオのプロパティの設定

```
//シナリオ データ プロパティの設定
get_mng().set_scenario_data(_session_id, scenario_id,
"Scenario.ReplicateCompressAttr", "True");
```

以下の表は、共通のシナリオ プロパティ名のリストです。

プロパティ名	説明
Scenario.ScenarioName	シナリオ名
Scenario.BuildShares	ウィンドウ共有の同期
Scenario.SyncADS	NTFS ADS のレプリケート

ホスト プロパティの設定：set_host_data()

この set_scenario_data API 関数によって、ホストのプロパティを編集できます。ホストのほとんどのプロパティは、この API によって更新できます。

引数

この set_host_data API 関数には、以下のテーブルで説明されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session 関数 API のコールによって返されたセッション ID。
host_index	uint	ホストのインデックス。
scenario_id	uint	シナリオの ID。
property_name	string	ホストのプロパティ名。プロパティには、それぞれ一意の名前があります。
property_value	string	プロパティの値。

戻り値

API コールが正常に完了すると、この関数は値として true を返します。それ以外の場合は、false が返されます。

例：ホスト プロパティの設定

```
//ホストのホスト名と IP アドレスを設定します
get_mng().set_host_data(_session_id, scenario_id, master_host_index,
"Scenario.ReplicationTree.ReplNode.CommonHostProps.Host", "master");
get_mng().set_host_data(_session_id, scenario_id, master_host_index,
"Scenario.ReplicationTree.ReplNode.CommonHostProps.IP", "155.35.78.187");
```

以下の表は、共通のマスタ ホスト プロパティ名のリストです。

プロパティ名	説明
Scenario.ReplicationTree.ReplNode.CommonHostProps.Host	マスタ ホスト名
Scenario.ReplicationTree.ReplNode.CommonHostProps.IP	マスタ ホスト IP
Scenario.ReplicationTree.ReplNode.CommonHostProps.Data_IP	マスタ レプリケーション IP アドレス
Scenario.ReplicationTree.ReplNode.CommonHostProps.Port	マスタ ホスト接続ポート番号

Scenario.ReplicationTree.ReplNode.CommonHostProps.SyncScriptBefore	同期前のスクリプトの実行
Scenario.ReplicationTree.ReplNode.CommonHostProps.SyncScriptBefore.Path	スクリプト パス
ReplicationTree.ReplNode.CommonHostProps.SyncScriptBefore.Args	スクリプト引数

シナリオの開始：run()

run API 関数を使用すると、シナリオを実行できます。ベスト プラクティスとしては、この run API をコールする前に、add_credentials_ex API をコールしてマスタ ホストを認証します。

引数

run API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session 関数 API のコールによって返されたセッション ID。
scenario_id	uint	開始するシナリオ ID。
sync_method	uint	同期方法。以下のいずれかのオプションを選択します。 **0** - ファイル レベル同期 **1** - ブロック レベル同期 **2** - ボリューム レベル同期 (FullSystem シナリオの場合のみ)
ignore_same_files	bool	サイズ/時間が同じファイルを無視します。
arc_upt	bool	このパラメータは、Arcserve Backup の統合シナリオのみに適用されます。シナリオが Arcserve Backup の統合シナリオ用でない場合は、このパラメータを false に設定する必要があります。 シナリオが <caab> と統合されている場合は、run API をコールする前に、add_bab_credentials API をコールします。
verification_and_run	uint	予約済み.常に 1 として定義する必要があります。
message	out string	この API が失敗したときに、その理由が含まれます。

戻り値

シナリオが正常に開始されると、true を返します。それ以外の場合、この関数は false を返し、メッセージ引数としてエラーの詳細な説明を提供します。

例

```
public bool run_example()
{
try
{
uint scenario_id = _scenario_id;
//0:File Sync;1:Block Sync;2:Volume Sync
```

```
uint sync_method = 0;
bool ignore_same_files = true;
bool arc_integrated = false;
string message = "";
return get_mng().run(_session_id, scenario_id, sync_method, ignore_same_files, arc_
integrated, 1, out message);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

シナリオの停止：stop()

stop API を使用すると、実行中のシナリオを停止できます。

引数

stop API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	停止するシナリオ ID。
execute_sync	bool	この API を同期または非同期でコールするかどうかを指定します。この引数を true に設定した場合、シナリオが停止されるまで API 関数は値を返しません。それ以外の場合、この関数は値を即座に返します。
why_not_reason	out string	この API が失敗したときに、その理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、why_not_reason 引数で返されるメッセージを確認して、API が失敗した理由を特定してください。

例

```
public bool stop_example()
{
try
{
uint scenario_id = _scenario_id;
string why_not_reason = "";
bool execute_sync = true;
return get_mng().stop(_session_id, scenario_id, execute_sync, out why_not_reason);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

データレプリケーションの一時停止：suspend_replication()

suspend_replication API 関数を使用すると、ホストのレプリケーション操作を一時停止できます。

引数

suspend_replication API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	一時停止するシナリオ ID。
replica_index	uint	シナリオ内のレプリカ ホスト インデックス。通常、レプリカ インデックスの値は 2 です。
execute_sync	bool	この API を同期または非同期でコールするかどうかを指定します。この引数を true に設定した場合、シナリオが完了するまで API 関数は値を返しません。それ以外の場合、この関数は値を即座に返します。
message	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool suspend_replication_example()
{
try
{
uint scenario_id = _scenario_id;
string message = "";
bool execute_sync = true;
uint replica_index = 2;
return get_mng().suspend_replication(_session_id, scenario_id, replica_index,
execute_sync, out message);
}
catch (Exception ex)
{
```

```
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

作業終了：close_session()

close_session API 関数を使用して、コントロール サービスからログアウトできます。コントロール サービスにログインした後、ログアウトするには close_session 引数をコールします。

引数

close_session API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
why_not_reason	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、セッションはクローズされました。戻り値が false の場合、セッションはクローズされませんでした。

例

```
public bool close_session_example()
{
try
{
string why_not_reason = "";
return get_mng().close_session(_session_id, out why_not_reason);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

認証情報の管理：add_credentials_ex()

add_credentials_ex API 関数を使用すると、ホストの認証情報を追加できます。

引数

add_credentials_ex API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session 関数 API のコールによって返されたセッション ID。
scenario_id	uint	認証情報の追加先となるシナリオ ID。
host_name	string	ホストの IP アドレス。 **例：** 155.35.76.44。
port	uint	エンジンのポート番号。通常、ポート番号は 25000 です。
user_name	string	ユーザ名。 **例：** 管理者
password	string	ユーザ名のパスワード。 **例：** Arcserve
domain_name	string	ドメイン名。 **例：** arcserve.com

戻り値

戻り値のタイプは ulong です。戻り値がゼロの場合は、API が正常に完了したことを示します。戻り値がゼロでない場合、API は失敗しました。

例

```
public bool add_credentials_ex_example()
{
try
{
uint scenario_id = _scenario_id;
//Add credential for Master
string host_ip = "155.35.66.138";
uint port = 25000;
string user_name = "administrator";
string password = "caworld";
string domain_name = "155.35.66.138";
ulong res = get_mng().add_credentials_ex(_session_id, scenario_id, host_ip, port,
user_name, password, domain_name);
//Add credential for Replica
host_ip = "155.35.66.142";
```

```
domain_name = "155.35.66.142";
res = get_mng().add_credentials_ex(_session_id, scenario_id, host_ip, port, user_
name, password, domain_name);
return (res == 0);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

Arcserve Backup add_bab_credentials() を使用した統合

add_bab_credentials API を使用すると、<caab> にアクセスするために認証情報を追加できます。

引数

add_bab_credentials API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	Arcserve 認証情報の追加先となるシナリオ ID。
username	string	ユーザ名。 **例：** 管理者
password	string	ユーザ名のパスワード。 **例：** <ca>
async_id	out ulong	この API 関数が非同期でコールされた場合、async_id 引数にはゼロ以外の値が含まれます。この場合、この関数は操作が完了して結果を取得するまで待機します。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。

例

```
public bool add_bab_credentials_example()
{
try
{
uint scenario_id = _scenario_id;
string username = "admin";
string password = "caworld";
ulong async_id = 0;
bool res = get_mng().add_bab_credentials(_session_id, scenario_
id,username,password,out async_id);
return res;
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
```

```
}
return false;
}
```

シナリオのプロパティの更新：update_scenario()

update_scenario API 関数を使用すると、実行時にシナリオデータを更新できます。

引数

update_scenario API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	更新するシナリオ ID。
scenario_data_str	ref string	シナリオに関するデータ。操作が正常に完了すると、新しいシナリオデータが提供されます。
why_not_reason	ref string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値は、XML 形式でシナリオのデータを提供します。

例

```
public bool update_scenario_example()
{
try
{
uint scenario_id = _scenario_id;
string why_not_reason = "";
string scenario_data_str = get_mng().get_scenario_data(scenario_id);
//do some changes for the scenario.
return get_mng().update_scenario(_session_id, scenario_id, ref scenario_data_str, out
why_not_reason);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

シナリオの削除：remove_scenario()

remove_scenario API 関数を使用すると、コントロール サービスからシナリオを削除できます。

引数

remove_scenario API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	実行されるシナリオ ID。
arc_upd	bool	このパラメータは、<caab> の統合シナリオのみに適用されます。
why_not-reason	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool remove_scenario_example()
{
try
{
uint scenario_id = _scenario_id;
bool arc_integrated = false;
string why_not_reason = "";
return get_mng().remove_scenario(_session_id, scenario_id, arc_integrated, out why_
not_reason);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

シナリオのインポート：import_scenario()

import_scenario API 関数を使用すると、シナリオをコントロール サービスにインポートできます。

引数

import_scenario API 関数には、以下のテーブルに記述されている引数が含まれています。

種類	名前	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
group_id	uint	インポートするシナリオ グループの ID。グループ ID が確認できない場合は、「0xFFFFFFFF」に設定します。
scenario_id	out uint	API が正常に完了する時に、シナリオ ID を取得します。
scenario_data	string	シナリオに関するデータ。通常、シナリオ ファイルからシナリオ データ文字列を取得します。このデータは XML 形式です。
why_not_reason	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値は、XML 形式でシナリオのデータを提供します。

例

```
public bool import_scenario_example()
{
try
{
uint scenario_id = 0;
string why_not_reason = "";
string scenario_data = "load the data from a scenario file.";
uint group_id = 0xFFFFFFFF;
return get_mng().import_scenario(_session_id, group_id,scenario_data,out scenario_
id,out why_not_reason);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

実行中のシナリオの同期：synchronize()

synchronize API 関数を使用すると、シナリオのデータを同期できます。

引数

synchronize API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	同期するシナリオ ID。
sync_method	uint	同期方法。以下のいずれかのオプションを選択します。 **0** - ファイルレベル同期 **1** - ブロックレベル同期 **2** - ボリュームレベル同期 (FullSystem シナリオの場合のみ)
ignore_same_files	bool	サイズ/時間が同じファイルを無視します。
execute_sync	bool	この API を同期または非同期でコールするかどうかを指定します。
message	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool synchronize_example()
{
try
{
uint scenario_id = _scenario_id;
string message = "";
bool execute_sync = true;
uint sync_method = 1;
bool ignore_same_files = false;
return get_mng().synchronize(_session_id, scenario_id, sync_method, ignore_same_
files, execute_sync, out message);
}
catch (Exception ex)
```

```
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

レプリケーションの再開：resume_replication()

resume_replication API 関数を使用すると、ホストのレプリケーション操作を再開できます。

引数

suspend_replication API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	再会するレプリケーションの対象となるシナリオ ID。
replica_index	uint	シナリオ内のレプリカ ホスト インデックス。通常、レプリカ インデックスの値は 2 です。これは、レプリケーションが一時停止されるホストです。たとえば、データ変更は、レプリケーション操作が再開するまで、ディスクにデータをコピーせずにスプールに集積されます。
execute_sync	bool	この API を同期または非同期でコールするかどうかを指定します。
message	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool resume_replication_example()
{
try
{
uint scenario_id = _scenario_id;
string message = "";
bool execute_sync = true;
uint replica_index = 2;
return get_mng().resume_replication(_session_id, scenario_id, replica_index,
execute_sync, out message);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
```

```
}
return false;
}
```

リワインド ブックマークの追加：set_rewind_bookmark ()

set_rewind_bookmark API 関数を使用すると、シナリオのブックマークを設定できます。

引数

set_rewind_bookmark API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
scenario_id	string	ブックマークを設定するシナリオ ID。
host_index	uint	常に 1。
bookmark_msg	string	ブックマーク名。
why_not_reason	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool set_rewind_bookmark_example()
{
try
{
string scenario_id = _scenario_id.ToString();
uint host_index = 1;
string why_not_reason = "";
string bookmark_msg = "test bookmark";
return get_mng().set_rewind_bookmark(scenario_id, host_index, bookmark_msg, out
why_not_reason);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

ハイ アベイラビリティ シナリオ管 理 API

以 下 のセクションでは、ハイ アベイラビリティ シナリオを管 理 するための API について説 明します。

このセクションには、以 下 のトピックが含 まれます。

- アシュアード リカバリのトリガ：start_ar()
- ハイ アベイラビリティ シナリオのハートビートの無 効 化 ：stop_is_alive()
- ハイ アベイラビリティ シナリオでスイッチオーバー：switchover()
- ハイ アベイラビリティ シナリオのハートビートの有 効 化 ：start_is_alive()

アシュアード リカバリのトリガ：start_ar()

start_ar API 関数を使用すると、シナリオのアシュアード リカバリ操作 (AR) を実行できます。自動 AR を実行する際に、AR を停止する他の API をコールする必要はありません。AR 操作が完了すると、AR は停止します。手動で AR を実行する場合は、API resume_application をコールして、AR 操作を停止します。

引数

start_ar API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	アシュアード リカバリを実行するシナリオ ID。
replica_index	uint	シナリオ内のレプリカ ホスト インデックス。通常、レプリカ インデックスの値は 2 です。これは、レプリケーションが一時停止されるホストです。
auto_ar	bool	AR を自動または手動で実行します。 ■ True - 自動 ■ False - 手動
execute_sync	bool	この API を同期または非同期でコールするかどうかを指定します。
message	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool start_ar_example()
{
try
{
uint scenario_id = _scenario_id;
string message = "";
bool execute_sync = true;
uint replica_index = 2;
bool auto_ar = true;
```

```
return get_mng().start_ar(_session_id, scenario_id, replica_index, auto_ar, execute_
sync, out message);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

ハイ アベイラビリティ シナリオのハートビートの無効化：stop_is_alive()

Is-alive は、ノードのステータスを識別するために、レプリカ サーバがマスタ サーバに送信する電子信号です。ハイ アベイラビリティ シナリオの実行中に、レプリカ サーバは定期的に電子信号 (ping) をマスタ サーバに送信します。デフォルトでは、ping の間隔は 30 秒です。事前定義された期間 (デフォルトでは 300 秒) の経過後、レプリカ サーバがマスタ サーバに対して ping を実行できない場合、スイッチオーバー イベントをトリガできます。

stop_is_alive API 関数を使用すると、is-alive チェックを一時停止できます。

引数

stop_is_alive API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	is-alive チェックを一時停止するシナリオ ID。
execute_sync	bool	この API を同期または非同期でコールするかどうかを指定します。
err_message	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool stop_is_alive_example()
{
try
{
uint scenario_id = _ha_scenario_id;
string err_messages = "";
bool execute_sync = true;
return get_mng().stop_is_alive(session_id, scenario_id, execute_sync, out err_
messages);
}
catch (Exception ex)
{
```

```
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

ハイ アベイラビリティ シナリオでスイッチオーバー: switchover()

switchover API 関数を使用すると、スイッチオーバー操作を実行できます。

フル システムのハイ アベイラビリティ シナリオで、任意のレプリカ サーバへのスイッチオーバー操作を実行できます。非フェールオーバー レプリカ サーバにスイッチオーバーする場合、スイッチオーバー API を呼び出す前に、execute_action API を呼び出します。

注: execute_action API は、例で記述されています。

引数

switchover API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	スイッチオーバー操作を実行するシナリオ ID。
execute_sync	bool	この API を同期または非同期でコールするかどうかを指定します。
run_reverse_scenario	bool	スイッチオーバー操作が発生した後に、バックワード シナリオを実行または実行しません。
err_message	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

例 1

```
public bool switchover_example()
{
try
{
uint scenario_id = _ha_scenario_id;
string err_messages = "";
bool execute_sync = true;
bool run_reverse_scenario = false;
```

```
return get_mng().switchover(_session_id, scenario_id, execute_sync, run_reverse_
scenario, out err_messages);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

例 2

```
public bool switchover_2nd_example()
{
try
{
set_xcmd_data("switchover", "switchover_index","3" );
uint scenario_id = _ha_scenario_id;
string err_messages = "";
bool execute_sync = true;
bool run_reverse_scenario = false;
return get_mng().switchover(_session_id, scenario_id, execute_sync, run_reverse_
scenario, out err_messages);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

例 3

デフォルトでは、Replication and High Availability は、事前定義済みのフェールオーバ、レプリカ サーバへのスイッチオーバー操作を実行します。フル システムのハイ アベイラビリティ シナリオで、非フェールオーバ レプリカ サーバへのスイッチオーバーを実行できます。ただし、スイッチオーバー API を使用して、非フェールオーバ サーバにスイッチオーバーする場合は、以下の例に示すように、スイッチオーバー API を呼び出す前に、execute_action API を呼び出します。

```
set_xcmd_data("switchover", "switchover_index","3" );
public bool set_xcmd_data(string cmd_name_str,string cmd_data_str,string cmd_
value_str)
{
try
{
string result_data = "";
string action_data;
```

```
XmlDocument doc = new XmlDocument();
XmlNode actions = doc.CreateNode(XmlNodeType.Element, xomngapi.WANSync_
c.xo_actions, "");
XmlNode commonNode = doc.CreateNode(XmlNodeType.Element,
xomngapi.WANSync_c.action_common_lab, "");
XmlAttribute attrSession = doc.CreateAttribute(xomngapi.WANSync_c.action_com_
session_id);
XmlAttribute attrScenario = doc.CreateAttribute(xomngapi.WANSync_c.action_com_
scenario_id);
XmlAttribute attrHostindex = doc.CreateAttribute(xomngapi.WANSync_c.action_com_
host_index);
XmlAttribute attrUsedfor = doc.CreateAttribute(xomngapi.WANSync_c.action_used_
for);
attrSession.Value = xomngapi.WANSync_c.WANSync.session_id.ToString();
attrScenario.Value = this.id.ToString();
attrUsedfor.Value = xomngapi.WANSync_c.action_x_command_data;
commonNode.Attributes.Append(attrSession);
commonNode.Attributes.Append(attrScenario);
commonNode.Attributes.Append(attrHostindex);
commonNode.Attributes.Append(attrUsedfor);
XmlNode xo_cmd = doc.CreateNode(XmlNodeType.Element, xomngapi.WANSync_
c.xo_cmd, "");
XmlAttribute cmd_name = doc.CreateAttribute(xomngapi.WANSync_c.action_cmd_
name);
XmlAttribute cmd_data = doc.CreateAttribute(xomngapi.WANSync_c.action_cmd_
data);
XmlAttribute cmd_value = doc.CreateAttribute(xomngapi.WANSync_c.action_cmd_
value);
cmd_name.Value = cmd_name_str;
cmd_data.Value = cmd_data_str;
cmd_value.Value = cmd_value_str;
xo_cmd.Attributes.Append(cmd_name);
xo_cmd.Attributes.Append(cmd_data);
xo_cmd.Attributes.Append(cmd_value);
actions.AppendChild(commonNode);
commonNode.AppendChild(xo_cmd);
doc.AppendChild(actions);
action_data = doc.OuterXml;
string error;
return get_mng().execute_action(action_data, true, out result_data, out error);
}
catch (System.Exception)
{
return false;
}
```

ハイ アベイラビリティ シナリオのハートビートの有効化：start_is_alive()

Is-alive は、ノードのステータスを識別するために、レプリカ サーバがマスタ サーバに送信する電子信号です。ハイ アベイラビリティ シナリオの実行中に、レプリカ サーバは定期的に電子信号 (ping) をマスタ サーバに送信します。デフォルトでは、ping の間隔は 30 秒です。事前定義された期間 (デフォルトでは 300 秒) の経過後、レプリカ サーバがマスタ サーバに対して ping を実行できない場合、スイッチオーバー イベントをトリガできます。

start_is_alive API 関数を使用すると、is-alive チェックを再開できます。

引数

start_is_alive API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	is-alive チェックを開始するシナリオ ID。
execute_sync	bool	この API を同期または非同期でコールするかどうかを指定します。
err_message	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool start_is_alive_example()
{
try
{
uint scenario_id = _ha_scenario_id;
string err_messages = "";
bool execute_sync = true;
return get_mng().start_is_alive(session_id, scenario_id, execute_sync, out err_
messages);
}
catch (Exception ex)
{
```

```
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

VSS スナップショット管理 API

以下のセクションでは、VSS スナップショットを管理するための API について説明します。

このセクションには、以下のトピックが含まれます。

- VSS スナップショットをレプリカ サーバにマウント：mount_snapshot()
- VSS スナップショットをレプリカ サーバからマウント解除：unmount_snapshot()
- VSS スナップショットをレプリカ サーバから削除：delete_snapshot()
- VSS スナップショットのリストをレプリカ サーバから取得：get_snapshot_list()

VSS スナップショットをレプリカ サーバにマウント : mount_snapshot()

mount_snapshot API 関数を使用すると、VSS スナップショットをレプリカ サーバ上の特定のフォルダにマウントできます。

引数

mount_snapshot API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
host_name	string	エンジンのホスト名
ip_string	string	host_name の IP アドレス。
host_port	ushort	エンジンのポート番号。通常、ポート番号は 25000 です。
mount_path	string	スナップショットをマウントするフォルダ。
snapshot_id	string	VSS スナップショット ID。
why_not_ reason	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool mount_snapshot_example()
{
try
{
string host_name = "155.35.66.142";
string ip_string = "155.35.66.142";
ushort host_port = 25000;
string mount_path = "c:/mount";
string snapshot_id = "{9CFDE664-62D5-4fd8-A304-2B664900B98F}";
string why_not_reason = "";
return get_mng().mount_snapshot(session_id, host_name, ip_string, host_port,
snapshot_id, mount_path, out why_not_reason);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
```

```
}
return false;
}
```

VSS スナップショットをレプリカ サーバからマウント解除：unmount_snapshot()

unmount_snapshot API 関数を使用すると、VSS スナップショットをフォルダからマウント解除できます。

引数

unmount_snapshot API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
host_name	string	エンジンのホスト名
ip_string	string	host_name の IP アドレス。
host_port	ushort	エンジンのポート番号。通常、ポート番号は 25000 です。
snapshot_id	string	VSS スナップショット ID。
why_not_reason	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool unmount_snapshot_example()
{
try
{
string host_name = "155.35.66.142";
string ip_string = "155.35.66.142";
ushort host_port = 25000;
string snapshot_id = "{9CFDE664-62D5-4fd8-A304-2B664900B98F}";
string why_not_reason = "";
return get_mng().unmount_snapshot(session_id, host_name, ip_string, host_port,
snapshot_id, out why_not_reason);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
```

```
}
```

VSS スナップショットをレプリカ サーバから削除：delete_snapshot()

delete_snapshot API 関数を使用すると、VSS スナップショットをレプリカ サーバから削除できます。

引数

delete_snapshot API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
host_name	string	エンジンのホスト名
ip_string	string	host_name の IP アドレス。
host_port	ushort	エンジンのポート番号。通常、ポート番号は 25000 です。
snapshot_id	string	VSS スナップショット ID。
why_not_reason	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool delete_snapshot_example()
{
try
{
string host_name = "155.35.66.142";
string ip_string = "155.35.66.142";
ushort host_port = 25000;
string snapshot_id = "{9CFDE664-62D5-4fd8-A304-2B664900B98F}";
string why_not_reason = "";
return get_mng().delete_snapshot(session_id, host_name, ip_string, host_port,
snapshot_id, out why_not_reason);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
```

```
}
```

VSS スナップショットのリストをレプリカ サーバから取得: get_snapshot_list()

get_snapshot_list API 関数を使用すると、ホストから VSS スナップショットのリストを取得できます。

引数

get_snapshot_list API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
host_name	string	エンジンのホスト名
ip_string	string	host_name の IP アドレス。
host_port	ushort	エンジンのポート番号。通常、ポート番号は 25000 です。
snapshot_list	out string	ボリューム スナップショット リスト。
why_not_reason	out string	この API が失敗する時に、その失敗の理由が含まれます。

戻り値

戻り値のタイプはブールです。戻り値が true の場合、API は正常に完了しています。戻り値が false の場合、API は正常に完了しませんでした。戻り値が false の場合は、メッセージを確認して API が失敗した理由を特定してください。

例

```
public bool get_snapshot_list_example()
{
try
{
string host_name = "155.35.66.142";
string ip_string = "155.35.66.142";
ushort host_port = 25000;
string snapshot_list = "";
string why_not_reason = "";
return get_mng().get_snapshot_list(session_id, host_name, ip_string, host_port, out
snapshot_list, out why_not_reason);
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
```

```
return false;
}
```

統計収集 API

以下のセクションでは、シナリオに関する統計を収集するプロセスを管理するための API について説明します。

このセクションには、以下のトピックが含まれます。

- 拡張シナリオ統計の取得：get_data_ex()
- シナリオの統計の取得：get_scenario_data()

拡張シナリオ統計の取得：get_data_ex()

get_data_ex API 関数を使用すると、以下のようなシナリオ情報をすべて取得できます。

- シナリオ イベント
- シナリオの状態
- 同期およびレプリケーションの統計

パラメータ manager_data は XML 形式の文字列です。このデータにはすべてのシナリオ情報が含まれます。たとえば、シナリオ ステータス(実行中、停止など) 、イベント、シナリオ統計などです。XML 形式の文字列とは、以下のようなものです。

```
<?xml version="1.0"?>
<manager_data>
 - <scenarios>
    + <scenario signature="4636778060728034734" ha_type="Forward" is_arcserve_integrated="False" is_cdp="Fa
      id="1094498606">
    + <scenario signature="4334615870148788711" is_arcserve_integrated="False" is_cdp="False" is_ass_rec="Fal
    + <scenario signature="14684688067413199200" ha_type="Forward" is_arcserve_integrated="False" is_cdp="F
    + <scenario signature="15270013466011305316" ha_type="Forward" is_arcserve_integrated="False" is_cdp="F
      id="43557253">
    + <scenario signature="5773759741404806146" is_arcserve_integrated="False" is_cdp="False" is_ass_rec="Fal
    + <scenario signature="7020398949829650879" is_arcserve_integrated="False" is_cdp="False" is_ass_rec="Fal
    + <scenario signature="7044671085026122361" is_arcserve_integrated="False" is_cdp="False" is_ass_rec="Tru
    + <scenario signature="7920379657132428156" ha_type="Forward" is_arcserve_integrated="False" is_cdp="Fa
      id="3423940998">
   </scenarios>
  + <scenario_groups>
</manager_data>
```

以下のセクションでは、XML 文字列の使用方法を示します。

引数

この get_data_ex API には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenarios_with_statistics	uint	シナリオ ID の配列。シナリオの統計情報を取得します。
last_update_time	ulong	前回更新されたタイムスタンプ。
request_flag	uint	データ タイプをリクエストします。値は以下のようになります。 **1** - シナリオ データ

		2 - cdp データ(使 用されていません) **4** - ホスト管 理 データ **8** - スナップショット データ **15** - 上 記 のすべてのデータ
manager_data	out string	XML 形 式 でのシナリオのデータを返します。

戻り値

戻り値 のタイプはブールです。戻り値 が true の場 合 、コマンドは正 常 に完 了して います。戻り値 が false の場 合 、コマンドは正 常 に完 了しませんでした。戻り値 が false の場 合 は、メッセージを確 認して API が失 敗した理 由 を特 定してください。

例

例 1:

```
public bool get_data_ex_example()
{
try
{
uint[] scenarios_with_statistics = new uint[] { _scenario_id };
uint request_flag = 1;
ulong last_update_time = 0;
string manager_data = "";
bool res = get_mng().get_data_ex(_session_id, scenarios_with_statistics, request_
flag, ref last_update_time, out manager_data);
return res;
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```

例 2:

この関 数 は、get_data_ex() 関 数 によって返 された XML バッファ データ(manager_ data) を解 析します。以 下 の例 は、シナリオの実 行 時 または停 止 時 にシナリオの 状 態 を取 得 する方 法 を示します。

```
string get_scenario_state (string manager_data, string scenario_name_or_id)
```

```
{
string scenario_state = "unknown";
XmlDocument manager_data_doc = new XmlDocument();
manager_data_doc.LoadXml(manager_data);
XmlNode root_node = manager_data_doc.SelectSingleNode(".//manager_data");
if (root_node == null)
{
return scenario_state;
}
//get all the scenario data information
XmlNode scenario_nodes = root_node.SelectSingleNode(".//シナリオ");
if (scenario_nodes == null)
{
return scenario_state;
}
foreach (XmlNode scenario_node in scenario_nodes.ChildNodes)
{
uint scenario_id = 0;
string scenario_name = "";
XmlAttribute id_attr = scenario_node.Attributes["id"];
if (id_attr ! = null)
scenario_id = Convert.ToUInt32(id_attr.Value);
XmlAttribute name_attr = scenario_node.Attributes["name"];
if (name_attr ! = null)
scenario_name = name_attr.Value;
if (scenario_name_or_id ! = scenario_id.ToString() && scenario_name_or_id.ToLower
() ! = scenario_name.ToLower())
continue;
//get the scenario status, running or stopped
foreach (XmlNode node in scenario_node.ChildNodes)
{
//get the scenario state, running or stop
if (0 == string.Compare("state", node.Name, true))
{
if (node.Attributes["val"] ! = null)
{
scenario_state = node.Attributes["val"].Value.ToLower();
}
}
}
}
return scenario_state;
}
```

例 3:

この関数は、get_data_ex() 関数によって返された XML バッファデータ(manager_data) を解析します。以下の例は、シナリオのすべてのイベントを取得する方法を示します。

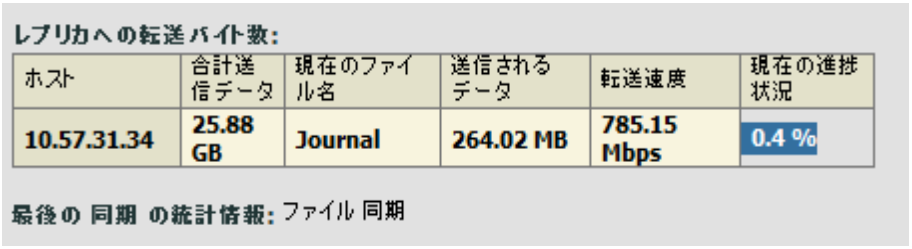
```
void get_scenario_events(string manager_data, string scenario_name_or_id, ref
ArrayList events)
{
XmlDocument manager_data_doc = new XmlDocument();
manager_data_doc.LoadXml(manager_data);
XmlNode root_node = manager_data_doc.SelectSingleNode(".//manager_data");
if (root_node == null)
{
return ;
}
//get all the scenario data information
XmlNode scenario_nodes = root_node.SelectSingleNode(".//シナリオ");
if (scenario_nodes == null)
{
return ;
}
foreach (XmlNode scenario_node in scenario_nodes.ChildNodes)
{
uint scenario_id = 0;
string scenario_name = "";
XmlAttribute id_attr = scenario_node.Attributes["id"];
if (id_attr ! = null)
scenario_id = Convert.ToUInt32(id_attr.Value);
XmlAttribute name_attr = scenario_node.Attributes["name"];
if (name_attr ! = null)
scenario_name = name_attr.Value;
if (scenario_name_or_id ! = scenario_id.ToString() && scenario_name_or_id.ToLower
() ! = scenario_name.ToLower())
continue;
//get the scenario status, running or stopped
foreach (XmlNode node in scenario_node.ChildNodes)
{
//get the scenario state, running or stop
if (0 == string.Compare("gen", node.Name, true))
{
events.Add(new event_data_c(node));
}
}
}
}
```

例 4:

シナリオの同期/レプリケーションの統計を取得するには、scenarios_with_statisticsパラメータを定義します。このパラメータは配列です。複数のシナリオの統計を取得するには、それらのシナリオのIDを配列に追加します。

get_data_exは以下の統計を取得します：

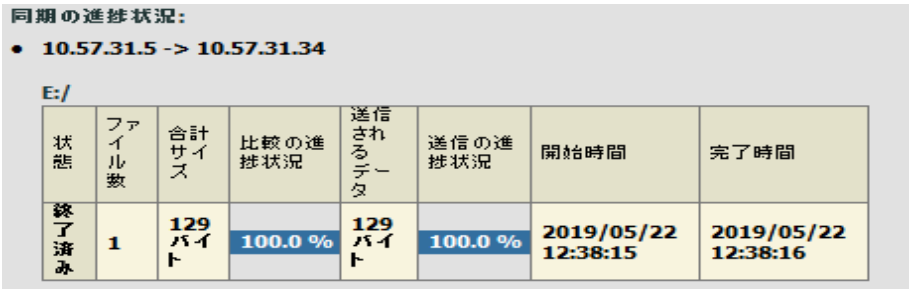
注：以下の画面は、転送および同期の統計情報のプロセスを示します。



レプリカへの転送バイト数:

ホスト	合計送信データ	現在のファイル名	送信されるデータ	転送速度	現在の進捗状況
10.57.31.34	25.88 GB	Journal	264.02 MB	785.15 Mbps	0.4 %

最後の 同期 の統計情報: ファイル 同期



同期の進捗状況:

- 10.57.31.5 -> 10.57.31.34

E:/

状態	ファイル数	合計サイズ	比較の進捗状況	送信されるデータ	送信の進捗状況	開始時間	完了時間
終了済み	1	129 バイト	100.0 %	129 バイト	100.0 %	2019/05/22 12:38:15	2019/05/22 12:38:16

コード

この関数は、get_data_ex() 関数によって返されたXMLバッファデータ(manager_data) を解析します。以下のコードは、転送および同期の統計情報を取得する方法を示します(前の画面を参照してください) 。

```
void get_scenario_sync_statistics(string manager_data, string scenario_name_or_id,
ref ArrayList sync_statistics)
{
XmlDocument manager_data_doc = new XmlDocument();
manager_data_doc.LoadXml(manager_data);
XmlNode root_node = manager_data_doc.SelectSingleNode(".//manager_data");
if (root_node == null)
{
return;
}
//get all the scenario data information
XmlNode scenario_nodes = root_node.SelectSingleNode(".//シナリオ");
if (scenario_nodes == null)
{
return;
}
foreach (XmlNode scenario_node in scenario_nodes.ChildNodes)
{
uint scenario_id = 0;
```

```
string scenario_name = "";
XmlAttribute id_attr = scenario_node.Attributes["id"];
if (id_attr ! = null)
scenario_id = Convert.ToUInt32(id_attr.Value);
XmlAttribute name_attr = scenario_node.Attributes["name"];
if (name_attr ! = null)
scenario_name = name_attr.Value;
if (scenario_name_or_id ! = scenario_id.ToString() && scenario_name_or_id.ToLower
() ! = scenario_name.ToLower())
continue;
//get the scenario status, running or stopped
foreach (XmlNode node in scenario_node.ChildNodes)
{
//get the scenario state, running or stop
if (0 == string.Compare("statistics", node.Name, true))
{
sync_statistics.Add(new host_statistics_c(node));
}
}
}
}
```

例 5：

以下のコードは、シナリオの同期統計情報を取得する方法を示します。

```
ArrayList sync_statistics = new ArrayList();
get_scenario_sync_statistics(manager_data, "FileServer 1", ref sync_statistics);
//show the statistics
foreach (host_statistics_c stat in sync_statistics)
{
//host name
string host_name = stat.host_name;
//transmission statistics
foreach (transfer_to_replica_c trans in stat.trans_to_reps)
{
//handle the transfer data such as speed.
ulong speed = trans.transmission_speed;
}
//sync statistics
foreach (sync_statistics_host_c sync_host in stat.children_hosts)
{
//root directory
foreach(sync_statistics_root_dir_c root_dir in sync_host.sync_root_dirs)
{
//root_dir.total_size
}
}
```

```
}
```

シナリオの統計の取得：get_scenario_data()

get_scenario_data API 関数を使用すると、シナリオ ID を取得できます。

引数

get_scenario_data API 関数には、以下のテーブルに記述されている引数が含まれています。

名前	種類	説明
session_id	uint	create_session API をコールすることで返されたセッション ID。
scenario_id	uint	シナリオ ID。

戻り値

戻り値は、シナリオのデータを提供します。

例

```
public bool get_scenario_data_example()
{
try
{
uint scenario_id = _scenario_id;
string scenario_data_str = get_mng().get_scenario_data(session_id.scenario_id);
return true;
}
catch (Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
return false;
}
```