テクニカル レポート

NetApp Manageability SDKセキュリティ強化ガイド

NetApp
Swati Godha
2023年5月| TR-4945

概要

このテクニカル レポートでは、組織が情報システムの機密性、整合性、可用性について定めたセキュリティ目標に沿ってNetApp® ONTAP® 9を導入するためのガイダンスと設定を記載します。

<<本レポートは機械翻訳による参考訳です。公式な内容はオリジナルである英語版をご確認ください。>>

目次

表一覧

© 2023 NetApp, Inc. All rights reserved.

図一覧

 © 2023 NetApp, Inc. All rights reserved.

はじめに

現在、進化を続ける脅威から最も価値のある資産であるデータと情報を保護するため、組織は今までに経験したことのない課題に直面しています。日々進化する脅威や脆弱性はますます洗練され、難読化やスパイ技術も巧妙化しているため、システム管理者にはデータや情報のセキュリティにプロアクティブに対処することが求められています。このガイドは、セキュリティ部門のオペレータや管理者に対し、NetAppソリューションの中核をなす機密性、整合性、可用性を活用した支援を提供することを目的としています。

NMSDKパッケージの整合性の検証

ここでは、NMSDKバイナリの整合性を検証する2つの方法、チェックサムの検証と署名の検証について説明します。

チェックサムの確認

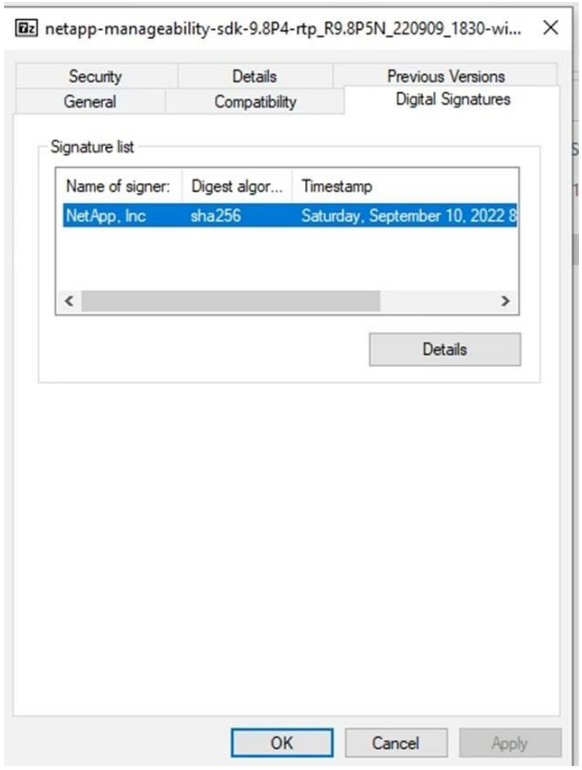
チェックサムは、NMSDKのダウンロードページで提供されています。ユーザーは、ダウンロードしたパッケージのチェックサムを、NMSDK ダウンロードページで提供されているチェックサムと照合する必要があります。

シグネチャの検証

Windowsでの署名の確認

NetApp Support SiteからNMSDKをダウンロードした後、ユーザーはexeの署名を確認できます。確認するには、ダウンロードしたexeファイルを右クリックして[プロパティ]を開きます。[プロパティ]ダイアログの[デジタル署名]タブで、署名者の名前がNetApp Incと表示され ます(図1)。

図1）Windowsで署名を確認する



© 2023 NetApp, Inc. All rights reserved.

Linuxでのシグネチャの検証

Red Hat Enterprise Linux（RHEL）の製品zipに加えて、NetAppは製品ダウンロードページでコード署名証明書をお客様と共有しています。コード署名証明書から、次の公開鍵を抽出できます。

```
#> openssl x509 -pubkey -noout -in <code_signing_certificate.pem> <pubkey.pub>
example:
#> openssl x509 -pubkey -noout -in csc-prod-NMSDK-LINUX.pem > csc-prod-NMSDK-LINUX.pub
```

公開鍵を使用して、次の製品郵便番号の署名を確認できます。

```
#> openssl dgst -sha256 -verify <public key> -signature <signature file> <Binary>
example:
#> openssl dgst -sha256 -verify csc-prod-NMSDK-LINUX.pub -signature netapp-manageability-sdk-
9.8P6-linux.zip.sig netapp-manageability-sdk-9.8P6-linux.zip
Verified OK => response
```

ONTAPとの接続の確立

ユーザ名とパスワード

NMSDKを使用してNetApp ONTAP® APIをテストする場合、ONTAPに接続するにはONTAPのユーザー名とパスワードが必要です。ONTAPのパスワードポリシーには、数字、小文字、大文字、特殊文字、およびパスワードの長さの要件を組み合わせたものが含まれています。

証明書ベースのAPIアクセス

ONTAPへのアクセスにNetApp Manageability SDK APIを使用する場合は、ユーザIDとパスワードによる認証の代わりに、証明書ベースの認証を使用する必要があります。

自己署名証明書を生成してONTAPにインストールするには、次の手順を実行します。

1. OpenSSLを使用して、次のコマンドを実行して証明書を生成します。

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout test.key -out test.pem -subj
"/C=US/ST=NC/L=RTP/O=NetApp/CN=cert_user"
```

   このコマンドは、`test.pem` という名前のパブリック証明書と、`key.out`という名前の秘密鍵を生成します。共通名CNはONTAPのユーザIDに対応します。

2. Privacy Enhanced Mail（PEM）形式のパブリック証明書の内容をONTAPにインストールします。次のコマンドを実行し、プロンプトが表示されたら証明書の内容を貼り付けます。

```
security certificate install -type client-ca -vserver ontap9-tme-8040
```

3. ONTAPがSSL経由のアクセスをクライアントに許可し、APIアクセスに使用するユーザIDを定義できるようにします。

```
security ssl modify -vserver ontap9-tme-8040 -client-enabled true
security login create -user-or-group-name cert_user -application ontapi -authmethod cert -role
admin -vserver ontap9-tme-8040
```

4. 次の例では、証明書認証を使用したAPIアクセスが使用できるようにユーザID`cert_user_` が有効になっています。ONTAPのバージョンを表示するための、`cert_user` を使用した簡単なManageability SDK Pythonスクリプトは、次のようになります。

```
#!/usr/bin/python

import sys
sys.path.append("/home/admin/netapp-manageability-sdk-9.5/netapp-manageability-
sdk9.5/lib/python/NetApp")
from NaServer import *

cluster = "ontap9-tme-8040"
```


© 2023 NetApp, Inc. All rights reserved.

```
transport = "HTTPS"
port = 443
style = "CERTIFICATE"
cert = "test.pem"
key = "test.key"

s = NaServer(cluster, 1, 30)
s.set_transport_type(transport)
s.set_port(port) s.set_style(style)
s.set_server_cert_verification(0)
s.set_client_cert_and_key(cert, key)

api = NaElement("system-get-version")
output = s.invoke_elem(api)
if (output.results_status() == "failed"):
  r = output.results_reason()
  print("Failed: " + str(r))
  sys.exit(2)

ontap_version = output.child_get_string("version")
print ("V: " + ontap_version)

The output of the script displays the ONTAP version.
./version.py
V: NetApp Release 9.5RC1: Sat Nov 10 05:13:42 UTC 2018
```

Active IQ Unified Managerとの接続の確立

NMSDKを使用してActive IQ Unified Manager APIをテストする場合、Active IQ Unified Managerに接続するにはActive IQ Unified Managerのユーザ名とパスワードが必要です。Active IQ Unified Managerのパスワードポリシーには、数字、小文字、大文字、特殊文字、およびパスワードの長さの要件を組み合わせたものが含まれています。

詳細：

監査ログ

監査ログには、ONTAP API呼び出し履歴が記録されます。にあります。`/etc/log/auditlog`

Active IQ Unified Managerの場合、呼び出し履歴はにあります。 `<dfm-server- installdirectory>/ log/`

EMSログ

EMSログは、ONTAP API処理情報を提供します。にあります。`/etc/log/ems`

syslog

Syslogでは、ONTAP APIエラーメッセージが表示されます。にあります。`/etc/messages`

DFMサーバログ

DFMサーバのログには、Active IQ Unified Managerコアパッケージの障害に対するActive IQ Unified Manager APIに関するエラーログ情報が記録されます。このログはにあります `<dfm-server-installdirectory>/log/`。

© 2023 NetApp, Inc. All rights reserved.

NMSDKロギング

NMSDKの9.8P6バージョンは、クライアントマシン上のすべてのCおよびJava APIから最小限のログをキャプチャします。次に、アンマウント処理の例を示します。

```
2023-04-25 17:08:24.032 [application:DEBUG]: [6572:0x27e8]: na_server_invoke:IN    API: system-get-version
2023-04-25 17:08:24.082 [application:DEBUG]: [6572:0x27e8]: na_server_invoke_elem invoked
2023-04-25 17:08:24.127 [application:DEBUG]: [6572:0x27e8]: na_server_invoke_elem_http:IN     on host=<host_IP>,look_up_host=1,major=<major_version>,minor=<minor_version>,style=0,
2023-04-25 17:08:25.096 [application:DEBUG]: [6572:0x27e8]: na_server_invoke_elem_http:OUT
2023-04-25 17:08:25.146 [application:DEBUG]: [6572:0x27e8]: na_server_invoke:OUT    API: system-get-version
```

Linuxプラットフォーム

環境変数を使用したログパスのカスタマイズ：

Linuxプラットフォームでは、nmsdkログフォルダのデフォルトパスは **/var/log/nmsdk/です。**このフォルダパスは、NMSDK_LOG_LOCATIONという名前の環境変数をユーザー定義の場所に設定することによってカスタマイズされます。Linuxシェルセッションでのコマンドの例を以下に示します。この場所に必要なすべての読み取り/書き込み権限があることを確認してください。

setenv NMSDK_LOG_LOCATION/var/customlocation/nmsdk/

または

NMSDK_LOG_LOCATION=/var/customlocation/nmsdk/をエクスポート

上記の手法は環境変数を設定することであるため、このコマンドを入力したLinuxシェルセッションでのみ動作します。すべてのセッションで環境変数を保持する必要がある場合は、rootユーザが.bash_profileファイルと.bashrcファイルに次の例の行を追加します。

file.bash_profile

nmsdk_log_location=/var/customlocation/nmsdk/export
nmsdk_log_location

file.bashrc内

NMSDK_LOG_LOCATION=/var/customlocation/nmsdk/をエクスポート

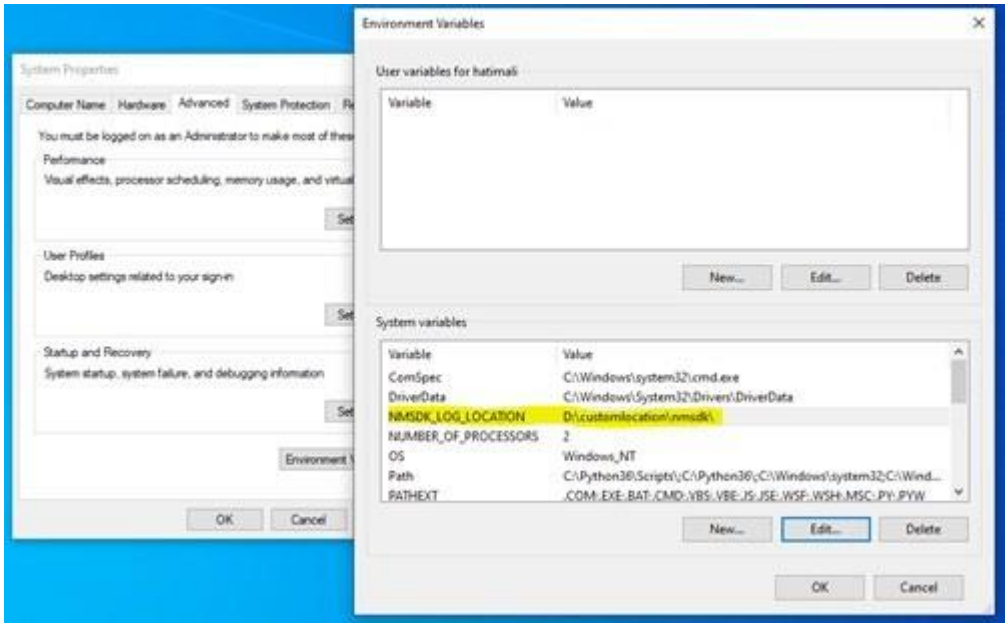
Windowsプラットフォーム

環境変数を使用したログパスのカスタマイズ：

Windowsプラットフォームでは、nmsdkログフォルダのデフォルトパスはC:\nmsdk\です。このフォルダパスは、NMSDK_LOG_LOCATIONという名前のシステム環境変数をユーザー定義の場所に設定することによってカスタマイズされます。この場所に必要なすべての読み取り/書き込み権限があることを確認してください。

© 2023 NetApp, Inc. All rights reserved.

図2）環境変数- NMSDK_LOG_LOCATION



NMSDKログローテーションおよび整合性スクリプト

NMSDKログのローテーションと整合性は、特定のフォルダの場所にあるNMSDK APIから生成されたログを自動的にローテーションする手法です。この手法では、ディスクストレージを効率的に使用するためにログを圧縮し、不正な変更をチェックします。

ログローテーションおよび整合性スクリプトの実装に使用されるユーティリティがいくつかあります。これらについては、後のセクションで詳しく説明します。

導入

ログのローテーションと整合性のために作成されたスクリプトは、LinuxとWindowsの両方のプラットフォームに実装されています。ログローテーションが実行されると、auth.logファイルはログファイルの整合性に関する情報で更新されます。ログファイルが改ざんされると、その情報がログに記録されます。次に、アンマウント処理の例を示します。

```
The log file nmsdk.log.5.gz for respective checksum file nmsdk.log.5.gz.cksum is deleted or
tampered
Checksum mismatch for nmsdk.log.4.gz
```

Linuxプラットフォーム

Linuxプラットフォームでスクリプトをテストするために使用される構成とユーティリティは次のとおりです。

オペレーティングシステム：

- Red Hat Enterprise Linux Server 7.7
- Firmware Version: 3.8.6
- Firmware Version: 3.6.8
- sha256sum（GNU coreutils）バージョン8.22

© 2023 NetApp, Inc. All rights reserved.

注: 上記のインストール済みパッケージの場所のパスはすべて、PATH環境変数で定義する必要があります。

ログローテーションおよび整合性スクリプトは、次の3つのファイルで構成されます。3つのファイルはすべて同じフォルダに配置されます。

1. logrotatescript.config
2. generateLogFileChecksum.py
3. verifyLogIntegrity.py

「logrotatescript.config」ファイルは、ログのローテーション時に「generateLogFileChecksum.py」と「verifyLogIntegrity.py」という名前の2つのPythonファイルを呼び出す設定ファイルです。以下は「logrotatescript.config」ファイルの青写真です。この構成ファイルは、Linuxテキスト形式で記述する必要があります。

```
NMSDK_LOG_FILE_PATH_PLACE_HOLDER>
{
        size <NMSDK_LOG_FILE_SIZE_PLACE_HOLDER>
        rotate <NMSDK_LOG_FILE_ROTATION_COUNT_PLACE_HOLDER>
        compress
        delaycompress
        missingok
        notifempty

        firstaction
                echo firstaction IN > /dev/null
                python3
<NMSDK_PROPERTY_FOLDER_PATH_PLACE_HOLDER>verifyLogIntegrity.py $1 > /dev/null
                echo firstaction OUT > /dev/null
        endscript
        lastaction
                echo lastaction IN > /dev/null
                python3
<NMSDK_PROPERTY_FOLDER_PATH_PLACE_HOLDER>generateLogFileChecksum.py $1 >
/dev/null
                echo lastaction OUT > /dev/null
        endscript
}
```

Windowsプラットフォーム

Windowsプラットフォームでスクリプトをテストするために使用される構成とユーティリティは次のとおりです。オペレーティングシステム：

- Windows 10 Enterpriseバージョン21H2
- Microsoft .NET Framework 4.7.2以降
- logrotateバージョン0.0.0.18（リンク：https://sourceforge.net/projects/logrotatewin/からインストール）
- Firmware Version: 3.6.8

注: 上記のインストール済みパッケージの場所のパスはすべて、PATH環境変数で定義する必要があります。

© 2023 NetApp, Inc. All rights reserved.

ログローテーションおよび整合性スクリプトは、次の3つのファイルで構成されます。3つのファイルはすべて同じフォルダに配置されます。

1. logrotatescript.config
2. generateLogFileChecksum.py
3. verifyLogIntegrity.py

「logrotatescript.config」ファイルは、ログのローテーション時に「generateLogFileChecksum.py」と「verifyLogIntegrity.py」という名前の2つのPythonファイルを呼び出す設定ファイルです。以下は「logrotatescript.config」ファイルの青写真です。この構成ファイルは、Linuxテキスト形式で記述する必要があります。

```
NMSDK_LOG_FILE_PATH_PLACE_HOLDER>
{
       size <NMSDK_LOG_FILE_SIZE_PLACE_HOLDER>
       rotate <NMSDK_LOG_FILE_ROTATION_COUNT_PLACE_HOLDER>
       compress
       delaycompress
       missingok
       notifempty

       prerotate
              echo prerotate IN > nul
              python.exe
<NMSDK_PROPERTY_FOLDER_PATH_PLACE_HOLDER>verifyLogIntegrity.py %1 > nul
              echo prerotate OUT > nul
       endscript
       postrotate
              echo postrotate IN > nul
              python.exe
<NMSDK_PROPERTY_FOLDER_PATH_PLACE_HOLDER>generateLogFileChecksum.py %1 > nul
              echo postrotate OUT > nul
       endscript
}
```

このファイルでは4つのプレースホルダが使用されており、デフォルト値が設定されています。カスタマイズする必要がある場合は、それらをそれぞれの値に置き換える必要があります。

- **<NMSDK_LOG_FILE_PATH_PLACE_HOLDER> :** このプレースホルダは、ログローテーション用にnmsdk.logファイルの完全修飾ファイルパスに置き換える必要があります。また、その整合性もチェックする必要があります。
- **<NMSDK_LOG_FILE_SIZE_PLACE_HOLDER> :** このプレースホルダは、ログファイルのサイズがこのしきい値サイズに達したときにログローテーションが行われるログファイルのしきい値サイズに置き換える必要があります。ログファイルは、サイズがバイトより大きくなった場合にのみローテーションされます。sizeに続けてkを指定すると、サイズはキロバイト単位とみなされます。Mを使用する場合のサイズはメガバイト単位で、Gを使用する場合のサイズはギガバイト単位です。したがって、サイズ100、サイズ100k、サイズ100m、サイズ100gはすべて有効です。
- **<NMSDK_LOG_FILE_ROTATION_COUNT_PLACE_HOLDER> :** このプレースホルダは、そのファイルの回転数に置き換える必要があります。ログファイルは、削除または上書きされるまでのローテーション回数です。カウントが0の場合、古いバージョンはローテーションではなく削除されます。
- **<NMSDK_property_folder_path_place_holder>:** このプレースホルダは、nmsdkプロパティ/構成/スクリプトファイルが存在する完全修飾フォルダパスに置き換える必要があります。

© 2023 NetApp, Inc. All rights reserved.

参考資料

参考資料については、次のリンクを参照してください。

- https://linux.die.net/man/8/logrotate
- https://sourceforge.net/p/logrotatewin/wiki/LogRotate/
- https://sourceforge.net/projects/logrotatewin/
- https://github.com/plecos/logrotatewin

NMSDKユーティリティの使用

NMSDKには、APIのテストに役立つ2つのユーティリティ（apitestとZExplore Development Interface（ZEDI））が用意されています。

適量

Apitestは、APIをテストするためのコマンドラインユーティリティです。このユーティリティは、初心者レベルのAPIユーザに適しています。C、Java、Perl、C#、VB.NETで利用できます。 Windows、Python、RubyのPowerShellです。UNIX系の環境では、apitestはC、Java、Perl、Python、Rubyで利用できる。

コマンド - `ruby apitest.rb 10.236.156.69 admin netapp1! system-get-version`

かぎります。

```
<results status="passed">
        <build-timestamp>1661323269</build-timestamp>
        <is-clustered>true</is-clustered>
        <version>NetApp Release Yellowdog_9.12.1: Wed Aug 24 06:41:09 UTC 2022</version>
        <version-tuple>
                <system-version-tuple>
                                <generation>9</generation>
                                 <major>12</major>
                                 <minor>1</minor>
                </system-version-tuple>
        </version-tuple>
</results>
```

ZExplore開発インターフェイス

ZExploreは、Active IQ Unified Managerコアパッケージ用のONTAP APIとNetApp Active IQ®Unified Manager APIをテストするためのUIです。ZEDIはJava Runtime Environment（JRE）1.6.0以降をサポート

図3）ZEDI

© 2023 NetApp, Inc. All rights reserved.

セキュリティ関連のクラスと機能の詳細

表1）概要を使用したセキュリティ関連クラスの詳細

名前：	説明
NaServer	Data ONTAP 6.4以降を実行するNetAppストレージシステムへの管理接続をカプセル化します。また、Active IQ Unified Manager（旧OnCommand Unified Manager）との接続の確立にも使用できます。クエリまたはコマンドを表すNaElementオブジェクトを作成し、を使用して `invoke_elem()` ストレージシステムまたはActive IQ Unified Managerに送信できます。
NaElement	XML要素の1レベルをカプセル化します。要素は任意にネストすることができます。XMLタグに対応する名前、属性（結果にのみ使用）、値（常に文字列）、ネストされたタグ付きアイテムに対応する子があります。NaElementsを使用してONTAPI API呼び出しを呼び出す手順については、前述のNaServer概要を参照してください。
ARCFOUR	データストリームの暗号化のためにARCFOUR（疑わしいRC4アルゴリズム）を実装するユーティリティクラス。RC4はRSA Data Security, Inc.の企業秘密であったが、1994年に匿名でメーリングリストに公開された。流出したバージョンが実際にRC4であったという証拠はなく、RC4は商標であるため、"ARCFOUR"と呼ばれている。
ベース16	文字配列をBase16文字列にエンコードし、再度デコードします。
承認情報	証明書ベースの認証に必要なキーストアと信頼ストアの詳細に関する情報が含まれます。
Base64	Base64表記との間でエンコードおよびデコードを行います。
HttpRequest	HTTP要求を表します。
shttpc.c	このHTTPクライアントは、TCP/SSL用に機能します。

表2）Python言語の関数の詳細

API名	説明	名前：
__init（self、server、major_version、minor_version）	サーバーへの新しい接続を作成します。使用する前に、スタイルをに設定する `hosts.equiv` か、ユーザ名（常に `root` 現在の状態）とパスワードをで設定する必要があり `set_admin_user()` ます。	NaServer
set_style（self、style）	• 原因サーバにログインを渡し、ユーザ名とパスワードを使用したHTTPシンプル認証を使用します。 • `hosts.equiv` Filer上のファイルを使用してアクセス権を決定するホストを渡します（ `root` その場合は、ユーザ名を指定する必要があります）。 • 証明書を渡して、Unified Managerサーバで証明書ベースの認証を使用します。 **注:** style=certificateの場合、証明書を使用して、ユーザ名とパスワードを必要とせずにサーバに接続しようとするクライアントを認証できます。この形式は、転送タイプを内部的にHTTPSに設定します。サーバのIDを正しく認証するには、サーバの証明書の検証が必要です。この形式を使用すると、サーバ証明書の検証がデフォルトで有効になり、サーバ証明書の検証では常にホスト名の検証が有効になります。	NaServer

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
	を使用して、（ホスト名を含む）サーバ証明書の検証を無効にすることができます `set_server_cert_verification()`。	
get_style（self）	認証形式を取得します。	NaServer
set_admin_user（self、user、password）	管理者のユーザ名とパスワードを設定します。現時点では、`user` 常にそうでなければなりません `root`。	NaServer
set_server_type（self、server_type）	filer、dfm、またはocumのいずれかのキーワードを指定して、サーバがストレージシステム（Filer）かActive IQ Unified Managerかを指定します。も使用する場合 `set_port()`は、 `set_port()` このルーチンを呼び出した後にを呼び出します。 デフォルト値はFILERです。	NaServer
get_server_type（self）	このサーバ接続が適用されるサーバのタイプを取得します。	NaServer
set_transport_type（self、scheme）	デフォルトの転送タイプを上書きします。現在の有効な転送タイプはHTTPとHTTPSです。	NaServer
get_transport_type（self）	この接続に使用されるトランスポートを取得します。	NaServer
set_port（self、port）	このサーバのデフォルトポートを上書きします。また電話をかける場合は `set_server_type()`、電話をかける前に電話をかける必要があり `set_port()`ます。	NaServer
get_port（self）	リモートサーバに使用されているポートを取得します。	NaServer
use_https（self）	HTTPSが有効かどうか	NaServer
set_client_cert_and_key（self、cert_file、key_file）	証明書を使用するサーバによるクライアント認証に必要なクライアント証明書とキーファイルを設定します。キーファイルが定義されていない場合、証明書ファイルがキーファイルとして使用されます。	NaServer
set_ca_certs（self、ca_file）	このアプリケーションによって信頼され、サーバ証明書の検証に使用される認証局（CA）の証明書を指定します。	NaServer
set_server_cert_verification（self、enable）	クライアントによるサーバ証明書の検証を有効または無効にします。形式がcertificateに設定されている場合、サーバ証明書の検証はデフォルトで有効になります。ホスト名（CN）の検証は、サーバ証明書の検証中に有効になります。 `set_hostname_verification()` APIを使用して、ホスト名の検証を無効にできます。	NaServer
is_server_cert_verification_enabled（self）	サーバ証明書の検証を有効にするかどうかを指定します。有効になっている場合はTrueを返し、有効になっていない場合はFalseを返します。	NaServer
set_hostname_verification（self、enable）	サーバ証明書の検証中のホスト名検証を有効または無効にします。ホスト名（CN）の検証は、サーバ証明書の検証中にデフォルトで有効になっています。	NaServer
is_hostname_verification_eが有効（self）	ホスト名の検証を有効にするかどうかを指定します。有効になっている場合はTrueを返し、有効になっていない場合はFalseを返します。	NaServer
child_add_string_encrypted（self、name、value、key=None）	と同じ `child_add_string`ですが、 `value key` 現在のオブジェクトに要素を追加する前にで暗号化します。 これは、現時点で特定の鍵交換操作にのみ使用されます。クライアントとサーバーの両方がの値を知ってい `key` て、このルーチンとそのコンパニオンを使用することに同意する必要があります `child_get_string_encrypted()`。 指定したキーがNoneの場合は、デフォルトのキーが使用されます。	NaElement

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
child_get_string_encrypted（self、name、key=None）	という名前の子の値を取得し namekey 、で復号化してから返します。 指定したキーがNoneの場合は、デフォルトのキーが使用されます。	NaElement
toEncodedString（self）	&、<、>などの特殊文字で埋め込まれた文字列をエンコードします。これは主に、&、<、>などの特殊文字が埋め込まれた文字列値をAPIに渡す場合に便利です。 例：server.invoke("qtree- create、"qtree"、"abc<qt0"、"volume"、"vol0"）	NaElement

表3）Ruby言語の関数の詳細

API名	説明	名前：
initialize（server、major_version、minor_version）	サーバーへの新しい接続を作成します。使用する前に、スタイルをに設定する hosts.equiv か、ユーザ名（常に root 現在の状態）とパスワードをで設定する必要があり set_admin_user()ます。	NaServer
set_style（スタイル）	• 原因サーバにログインを渡し、ユーザ名とパスワードを使用したHTTP簡易認証を使用します。 • hosts.equiv Filer上のファイルを使用してアクセス権を決定するホストを渡します（root その場合は、ユーザ名を指定する必要があります）。 • Unified Managerサーバで証明書ベースの認証を使用するための証明書を渡します。 **注:** $STYLE=CERTIFICATEの場合、証明書を使用して、ユーザー名とパスワードを必要とせずにサーバーに接続しようとするクライアントを認証できます。この形式は、転送タイプを内部的にHTTPSに設定します。サーバのIDを正しく認証するには、サーバの証明書の検証が必要です。 **注：** サーバ証明書の検証はこの形式を使用してデフォルトで有効になっており、サーバ証明書の検証では常にホスト名の検証が有効になります。を使用して、（ホスト名を含む）サーバ証明書の検証を無効にすることができます set_server_cert_verification()。	NaServer
get_style()	認証形式を取得します。	NaServer
set_admin_user（ユーザ、パスワード）	管理者のユーザ名とパスワードを設定します。現時点では、user 常にそうでなければなりません root。	NaServer
set_server_type（server_type）	filer、dfm、またはocumのいずれかのキーワードを渡して、サーバがストレージシステム（Filer）サーバかActive IQ Unified Manager（旧OnCommand Unified Manager）サーバかを示します。 も使用する場合 set_port()は、 set_port() このルーチンを呼び出した後にを呼び出します。 デフォルト値はFILERです。	NaServer
get_server_type()	このサーバ接続環境サーバのタイプを取得します。	NaServer

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
set_transport_type（スキーム）	デフォルトの転送タイプを上書きします。現在の有効な転送タイプはHTTPおよびHTTPSです。	NaServer
get_transport_type ()	この接続に使用されるトランスポートを取得します。	NaServer
ポート：	このサーバのデフォルトポートを上書きします。また電話をかける場合は `set_server_type()`、電話をかける前に電話をかける必要があり `set_port()` ます。	NaServer
get_port ()	リモートサーバに使用されているポートを取得します。	NaServer
use_https ()	HTTPSが有効かどうか	NaServer
set_server_cert_verification（イネーブル）	クライアントによるサーバ証明書の検証を有効または無効にします。形式がcertificateに設定されている場合、サーバ証明書の検証はデフォルトで有効になります。ホスト名（CN）の検証は、サーバ証明書の検証中は常に有効になります。	NaServer
is_server_cert_verification_enabled ()	サーバ証明書の検証を有効にするかどうかを指定します。有効になっている場合はTrueを返し、有効になっていない場合はFalseを返します。	NaServer
set_client_cert_and_key（cert_file、key_file=nil、key_passwd=nil）	証明書を使用するサーバによるクライアント認証に必要なクライアント証明書とキーファイルを設定します。キーファイルが定義されていない場合は、証明書ファイルがキーファイルとして使用されます。	NaServer
set_ca_certs（ca_file）	このアプリケーションによって信頼され、サーバ証明書の検証に使用されるCAの証明書を指定します。	NaServer
set_hostname_verification（イネーブル）	サーバ証明書の検証中のクライアントによるホスト名検証を有効または無効にします。	NaServer
is_hostname_verification_enabled()	ホスト名の検証を有効にするかどうかを指定します。有効になっている場合はTrueを返し、有効になっていない場合はFalseを返します。	NaServer
child_add_string_encrypted（name、value、key=nil）	と同じ `child_add_string`ですが、 `value key` 現在のオブジェクトに要素を追加する前にで暗号化します。 これは現在、特定の鍵交換操作にのみ使用されています。クライアントとサーバーの両方がの値を知ってい `key` て、このルーチンとそのコンパニオンを使用することに同意する必要があります `child_get_string_encrypted()`。 指定したキーがNoneの場合は、デフォルトのキーが使用されます。	NaElement
child_get_string_encrypted（name、key=nil）	という名前の子の値を取得し `namekey`、で復号化してから返します。 指定したキーがNoneの場合は、デフォルトのキーが使用されます。	NaElement
toEncodedString（self）	&、<、>などの特殊文字で埋め込まれた文字列をエンコードします。これは主に、&、<、>などの特殊文字が埋め込まれた文字列値をAPIに渡す場合に便利です。 例：server.invoke("qtree-create","qtree","abc<qt0","volume","vol0"）	NaElement

© 2023 NetApp, Inc. All rights reserved.

表4）Perl言語の関数の詳細

API名	説明	名前：
新規（$filer、$majorversion、$minorversion）	Filerへの新しい接続を作成します `$filer`。 使用する前に、スタイルをに設定する `hosts.equiv` か、ユーザ名（常に `root` 現在の状態）とパスワードをで設定する必要があり `set_admin_user()` ます。	NaServer
YLE（$style）	• 原因サーバにログインを渡し、ユーザ名とパスワードを使用したHTTP簡易認証を使用します。 • `hosts.equiv` Filer上のファイルを使用してアクセス権を決定するホストを渡します（その場合、ユーザ名はrootである必要があります）。 • Unified Managerサーバで証明書ベースの認証を使用するための証明書を渡します。 **注:** $STYLE=certificateの場合、証明書を使用して、ユーザ名とパスワードを必要とせずにサーバに接続しようとするクライアントを認証できます。この形式は、転送タイプを内部的にHTTPSに設定します。サーバのIDを正しく認証するには、サーバの証明書の検証が必要です。この形式を使用すると、サーバ証明書（ホスト名を含む）の検証がデフォルトで有効になります。を使用してサーバ証明書（ホスト名を含む）の検証を無効にすること `set_server_cert_verification()` も、を使用してホスト名の検証のみを無効にすること `set_hostname_verification()` もできます。	NaServer
get_style()	認証形式を取得します。	NaServer
set_admin_user（$user、$password）	管理者のユーザ名とパスワードを設定します。現時点では、`$user` 常にそうでなければなりません `root`。	NaServer
set_server_type（$type）	filer、dfm、またはocumのいずれかのキーワードを渡して、サーバがONTAPストレージシステムとActive IQ Unified Manager（旧OnCommand Unified Manager）サーバのどちらであるかを示します。 も使用する場合 `set_port()` は、 `set_port()` このルーチンを呼び出した後にを呼び出します。 デフォルト値はFILERです。	NaServer
get_server_type()	このサーバ接続が適用されるサーバのタイプを取得します。	NaServer
set_transport_type（$scheme）	デフォルトの転送タイプを上書きします。現在の有効な転送タイプはHTTPとHTTPSです。	NaServer
get_transport_type ()	この接続に使用されるトランスポートを取得します。	NaServer
set_port（$port）	このサーバのデフォルトポートを上書きします。また電話をかける場合は `set_server_type()`、電話をかける前に電話をかける必要があり `set_port()` ます。	NaServer
get_port ()	リモートサーバに使用されているポートを取得します。	NaServer
use_https ()	HTTPSが有効かどうか	NaServer
set_server_cert_verification ()	クライアントによるサーバ証明書の検証を有効または無効にします。形式がcertificateに設定されている場合、サーバ証明書の検証はデフォルトで有効になります。ホスト名の検証は、サーバ証明書の検証中にデフォルトで有効になっています。	NaServer

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
is_server_cert_verification_enabled ()	サーバ証明書の検証を有効にするかどうかを指定します。有効になっている場合は1を返し、有効になっていない場合は0を返します。	NaServer
set_client_cert_and_key()	証明書を使用するサーバによるクライアント認証に必要なクライアント証明書とキーファイルを設定します。キーファイルが定義されていない場合、証明書ファイルがキーファイルとして使用されます。	NaServer
set_ca_certs()	このアプリケーションによって信頼され、リモートサーバ証明書の検証に使用されるCAの証明書を指定します。	NaServer
set_hostname_verification ()	サーバ証明書の検証中のクライアントによるホスト名検証を有効または無効にします。	NaServer
is_hostname_verification_enabled()	ホスト名の検証を有効にするかどうかを指定します。有効になっている場合は1を返し、有効になっていない場合は0を返します。	NaServer
verify_server_certificate ()	サーバ証明書の共通名を特定のホスト名と照合して検証するサブルーチン。このサブルーチンは `undef` 成功した場合に戻ります。	NaServer
set_sslv3（$enable）	HTTPS転送で使用するSSLv3プロトコルを有効または無効にします。デフォルトでは、SSLv3プロトコルは無効になっています。	NaServer
is_sslv3()	HTTPS転送でSSLv3プロトコルを使用できるかどうかを指定します。SSLv3が有効な場合は1を返し、有効でない場合は0を返します。	NaServer
child_add_string_encrypted（$name、$value、$key）	と同じ `child_add_string`ですが、 `$value $key` 現在のオブジェクトに要素を追加する前にで暗号化します。 **注:** これは現在、特定の鍵交換操作にのみ使用されています。クライアントとサーバーの両方がの値を知ってい `$key` て、このルーチンとそのコンパニオンを使用することに同意する必要があります `child_get_string_encrypted()`。 指定したキーがの場合は、デフォルトのキーが使用され `undef`ます。	NaElement
child_get_string_encrypted（$name、$key）	子の値を取得し `$name` 、で復号化し `$key` てから返します。 指定したキーがの場合は、デフォルトのキーが使用され `undef`ます。	NaElement
toEncodedString()	&、<、>などの特殊文字で埋め込まれた文字列をエンコードします。これは主に、&、<、>などの特殊文字が埋め込まれた文字列値をAPIに渡す場合に便利です。 例：$server->invoke（"qtree-create"、"qtree"、"abc <qt0"、volume、"vol0"）；	NaElement

表5）Java言語の関数の詳細

API名	説明	名前：
setApiVersion（int majorVersion、int minorVersion）	要求のAPIバージョンを設定します。	NaServer

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
同期されたvoid setKeepAliveEnabled（boolEAN有効）	HTTPキープアライブをオンまたはオフにします。デフォルトはoffです。キープアライブが有効になっている場合は、`close()` このオブジェクトの使用が終了したらを呼び出す必要があります。	NaServer
同期されたvoid setPort (int TcpPort)	サーバでAPI呼び出しに使用するTCPポートを設定します。 ONTAP API通信に使用されるストレージシステムのデフォルトポートは次のとおりです。 • HTTP転送：80 • HTTPS転送：443 Active IQ Unified Manager 5.2以前のサーバでAPI通信用にデフォルトで使用されるポートは次のとおりです。 • HTTP転送：8080 • HTTPS転送：8488 Active IQ Unified Manager 6.0以降のサーバでAPI通信用にサポートされているポートは次のとおりです。 • HTTPS転送：443 **注:** サーバーにデフォルトのポート設定がある場合は、上記のポート番号をこのAPIへの入力として使用します。	NaServer
GETPORT ()	サーバでのAPI呼び出しに使用されるTCPポートを取得します。	NaServer
getServerType()	サーバタイプを取得します。これは、`SERVER_TYPE_`値	NaServer
同期されたvoid setServerType（int serverType)	指定したサーバタイプに適したURIとTCPポートを設定します。	NaServer
getTransportType()	転送タイプを取得します。これは、`TRANSPORT_TYPE_`値	NaServer
同期されたvoid setTransportType (int transportType)	転送タイプを設定します。転送タイプを設定したら、を呼び出して、`setPort()` その転送タイプに使用するポート番号を設定します。	NaServer
setAdminUser（文字列ログイン、文字列パスワード）	APIが呼び出されたときの認証に使用するログインとパスワードを設定します。	NaServer
setStyle（intスタイル）	以降のAPI認証の認証形式を設定します。	NaServer
getStyle()	現在の認証形式を取得します。	NaServer
setKeyStore（String keystore、String keyStorePasswd）	クライアント証明書とキーが存在するキーストアファイルの場所を設定します。	NaServer
setKeyStoreType (int keyStoreType)	クライアント証明書のキーストアタイプを設定します。	NaServer
enableServerCertVerification ()	クライアントによるサーバ証明書の検証をイネーブルにします。このメソッドは、ホスト名の検証を内部的に有効にします。認証形式がに設定されている場合、サーバ証明書の検証はデフォルトで有効になります `STYLE_CERTIFICATE`。 `setTrustStore()` メソッドを使用して、サーバの認証に必要な信頼されたCAの証明書を含む信頼ストアを指定します。	NaServer

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
	を使用して、ホスト名の検証を無効にできます `disableHostnameVerification()`。 **注:** `TRANSPORT_TYPE_HTTPS`このメソッドを呼び出す前に、転送タイプをHTTPS()に設定する必要があります。	
disableServerCertVerification ()	クライアントによるサーバ証明書の検証を無効にします。このメソッドは、ホスト名の検証を内部的に無効にします。 **注:** `TRANSPORT_TYPE_HTTPS`このメソッドを呼び出す前に、転送タイプをHTTPS()に設定する必要があります。	NaServer
isServerCertVerificationEnabled()	クライアントがサーバ証明書の検証を有効にするかどうかを指定します。	NaServer
enableHostnameVerification ()	サーバ証明書の検証中に、クライアントによるホスト名の検証をイネーブルにします。ホスト名の検証は、認証形式がに設定されている場合はデフォルトで有効になります `STYLE_CERTIFICATE`。 ホスト名の検証により、クライアントが接続するホスト名が、SSL接続の一部としてサーバが送り返す証明書のホスト名（CN名）と一致していることが確認されます。 ホスト名を検証するには、アプリケーションがNaServerコンストラクタでサーバに接続するIPアドレスではなく、ホスト名（CN名）を指定する必要があります。 この方法を使用する前に、サーバ証明書の検証を有効にする必要があります。	NaServer
disableHostnameVerification ()	サーバ証明書の検証中のクライアントによるホスト名検証をディセーブルにします。	NaServer
isHostnameVerificationEnabled()	サーバ証明書の検証中に、クライアントがホスト名の検証を有効にするかどうかを指定します。	NaServer
setTrustStore（文字列信頼ストア）	サーバの認証に必要な信頼できるCAの証明書が格納されている信頼ストアのデフォルトの場所を設定します。	NaServer
enableTLS()	HTTPS転送経由の接続で使用するTLSプロトコルを有効にします。 **注:** デフォルトでは、TLSプロトコルは有効になっています。	NaServer
disableTLS()	HTTPS転送で使用するTLSプロトコルを無効にします。 **注:** デフォルトでは、TLSプロトコルは有効になっています。	NaServer
isTLSEnabled()	HTTPS転送でTLSプロトコルを使用できるようにするかどうかを指定します。	NaServer
enableSSLv3()	HTTPS転送で使用するSSLv3プロトコルをイネーブルにします。 **注：** デフォルトでは、SSLv3プロトコルは無効になっています。	NaServer
disableSSLv3()	HTTPS転送で使用するSSLv3プロトコルを無効にします。 **注：** デフォルトでは、SSLv3プロトコルは無効になっています。	NaServer
isSSLv3Enabled()	HTTPS転送でSSLv3プロトコルを使用できるかどうかを指定します。	NaServer

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
setEncryptedContent（String gコンテンツ）	指定された値を暗号化し、暗号化された値をこの要素のコンテンツとして設定します。	NaElement
addNewEncryptedChild（文字列名、文字列の内容）	コンテンツに含まれるデータを暗号化します。指定された名前の `ant_sty;encrypted` 内容を持つ新しい子要素を追加します。	NaElement
getChildEncryptContent（Stリング名）	指定された名前の子を検索し、その文字列コンテンツを復号化して、復号化されたコンテンツを返します。`null` 指定した名前の子要素が見つからない場合に返します。	NaElement
getEncryptContent()	この要素のコンテンツの値を復号化して返します。	NaElement
初期化	暗号化エンジンを初期化します。 **注:** このメソッドは、一意の暗号化または復号化操作の前に呼び出します。	ARCFOUR
crypt（char[]入力、char[]出力）	暗号化または復号化します。これは、データのストリームを暗号化するために複数回呼び出すことができます。 **注:** `init()` 一意の暗号化または復号化操作の前に必ずを呼び出してください。	ARCFOUR
encryptAndEncode（文字列入力）	暗号化とBase16の両方がデフォルトキーを使用して文字列をエンコードする便利なメソッド。	ARCFOUR
decodeAndDecrypt（文字列入力）	`encryptAndEncode()` デフォルトのキーを使用してエンコードを元に戻し、から暗号化する便利な方法。	ARCFOUR
エンコード（char[]バイト）	Base16を使用して一部のバイトをエンコードします。	ベース16
decode（文字列テキスト）	Base16文字の文字列をバイトにデコードします。	ベース16
setKeyStore（String keystoreFile、String keyStorePasswd、String keyPasswd）	クライアント証明書とキーが格納されているキーストアの詳細を設定します。	承認情報
setKeyStoreType（文字列keyStoreType）	クライアントキーストアファイルのタイプを設定します。	承認情報
setTrustStore (string trustStoreFile)	サーバ認証に必要な信頼ストアファイルを設定します。	承認情報
getKeyStoreFile()	クライアントキーストアファイルを取得します。	承認情報
getKeyStorePasswd()	クライアントキーストアのパスワードを取得します。	承認情報
getKeyPasswd()	クライアントの秘密鍵パスワードを取得します。	承認情報
getKeyStoreType()	クライアントキーストアファイルのタイプを取得します。	承認情報
getTrustStoreFile()	サーバ信頼ストアファイルの場所を取得します。	承認情報
getTrustStoreType()	サーバ信頼ストアファイルのタイプを取得します。	承認情報
sethost（文字列ホスト）	サーバ認証に必要なホスト名を設定します。	承認情報
getHost()	サーバ認証に必要なホスト名を取得します。	承認情報
enableCBA ()	証明書ベースの認証メカニズムをイネーブルにします。	承認情報
disableCBA ()	証明書ベースの認証メカニズムをディセーブルにします。	承認情報
isCBAEnabled()	このフラグは、証明書ベースの認証メカニズムを使用するかどうかを示します。	承認情報

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
enableServerCertVerification ()	クライアントによるサーバ証明書の検証をイネーブルにします。	承認情報
disableServerCertVerification ()	クライアントによるサーバ証明書の検証をディセーブルにします。	承認情報
isServerCertVerificationEnabled()	サーバ証明書の検証を使用するかどうかを示すフラグ。	承認情報
enableHostnameVerification ()	サーバ証明書の検証中に、クライアントによるホスト名の検証をイネーブルにします。	承認情報
disableHostnameVerification ()	サーバ証明書の検証中のクライアントによるホスト名検証をディセーブルにします。	承認情報
isHostnameVerificationEnabled()	サーバ証明書の検証中に、クライアントがホスト名の検証を有効にするかどうかを指定します。	承認情報
比較（AuthInfo A1、AuthInfo A2）	は、キーストアおよび信頼ストアのプロパティに従って、指定された認証情報オブジェクトを比較します。 **注意:** 両方のプロパティが一致する場合はTrueを返します。	承認情報
encodeObject（java.io.Serializable serializableObject）	オブジェクトをシリアル化し、そのシリアル化されたオブジェクトのBase64エンコードバージョンを返します。 **注意:** オブジェクトをシリアル化できない場合、または別のエラーがある場合、メソッドは戻ります。	Base64
encodeBytes（byte[] source、int off、int len、booleanブレークライン）	バイト配列をBase64表記にエンコードします。	Base64
encodeString (文字列、ブールブレークライン)	Base64表記の文字列を75文字のBase64文字ごとに改行してエンコードします。 **注意:** 文字列をエンコードする必要があるのは、多くの英語以外の言語など、ASCII以外の文字が含まれている場合だけです。	Base64
readFile（文字列ファイル、ブールエンコード）	ファイルを読み取り、エンコードまたはデコードします。	Base64
WriteFile(byte[] data、文字列ファイル、ブールエンコード)	バイト配列を指定どおりにエンコードまたはデコードしてファイルに書き込みます。	Base64
encodeFromFile (String rawfile)	Base64がファイルをエンコードし、エンコードされた文字列を返す、またはエラーが発生した場合の単純なヘルパーメソッド。	Base64
decodeFromFile（文字列エンファイル）	Base64がファイルをデコードし、デコードされたデータを返す、またはエラーが発生した場合の単純なヘルパーメソッド。	Base64
encodeToFile(byte[]rawdata, 文字列ファイル)	Base64がデータをファイルにエンコードする単純なヘルパーメソッド。	Base64
decodeToFile(byte[]encdata, 文字列ファイル)	Base64がデータをファイルにデコードする単純なヘルパーメソッド。	Base64
decode（文字列s）	Base64表記からデータをデコードします。	Base64
decodeToString（文字列s）	Base64表記からデータをデコードし、文字列として返します。 **注意:** これは、<code> new String(decode(s))</code>を呼び出すのと同じです。	Base64

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
decodeToObject(String encodedObject)	Base64データをデコードし、内のJavaオブジェクトをデシリアライズしようとします。null エラーが発生した場合はを返します。	Base64
デコード（byte[] source、int off、int len）	Base64コンテンツをバイト配列形式でデコードし、デコードされたバイト配列を返します。	Base64
getPath()	このHTTP要求のURIパスを取得します。	HttpRequest
setBasicAuthorization（Stringユーザ名、文字列パスワード）	HTTPベーシック認証用のヘッダーを追加します。	HttpRequest
HttpRequest読み取り（InputStream入力）	ソケットからデータを読み取り、新しいHTTP要求を作成します。	HttpRequest

表6）.NET言語の関数の詳細

API名	説明	名前：
SetApplicationName（文字列applicationName）	NMSDKコアAPIを使用するクライアントアプリケーションの名前を設定します。	NaServer
GetApplicationName()	NMSDKコアAPIを使用しているクライアントアプリケーションの名前を取得します。	NaServer
SetAdminUser（文字列login、文字列password）	指定されたサーバコンテキストのユーザ名とパスワードを設定します。認証形式はです。 LOGIN_PASSWORD **注:** 指定するユーザーには、サーバーに対する管理者権限が必要です。	NaServer
ポート	サーバでのAPI呼び出しに使用するTCPポートを取得します。設定することも可能です。 ONTAP API通信に使用されるストレージシステムのデフォルトポートは次のとおりです。 • HTTP転送：80 • HTTPS転送：443 Active IQ Unified Managerサーバ5.2以前のデフォルトポートは次のとおりです。 • HTTP転送：8080 • HTTPS転送：8488 Active IQ Unified Managerサーバ6.0以降のデフォルトポートは次のとおりです。 • HTTPS転送：443 **注:** サーバーにデフォルトのポート設定がある場合は、上記のポート番号をこのAPIへの入力として使用します。	NaServer
NaServer.server_type ServerType	サーバタイプを取得します。設定することも可能です。サーバタイプは SERVER_TYPE 値 値は次のいずれかです。 • NetAppストレージ・システムに接続するFiler。 • DFMを使用してActive IQ Unified Managerサーバ5.2以前に接続します。	NaServer

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
	• Active IQ Unified Managerサーバに接続するためのOCUM 6.0以降 • NetAppホスト・エージェントに接続するエージェント 注： デフォルトのサーバタイプはfilerです。	
NaServer.auth_styleスタイル	認証形式（ AUTH_STYLE 値の1つ）を取得します。設定することも可能です。 サポートされる値は次のとおりです。 • ログインパスワード • RPC • ホステスキヴ • style=login_passwordの場合は、SetAdminUser() APIを使用して管理者のユーザ名とパスワードを指定する必要があります。 • style=rpcの場合、リモート手順呼び出しに基づくネイティブWindows認証が使用されるように、コードをWindowsマシン上で実行する必要があります。 • STYLE=HOSTSEQUIVの場合、サーバコンテキストは /etc/hosts.equiv NetAppストレージシステム上のファイルに照らして認証されます。 この場合、管理者のユーザ名とパスワードを設定する必要はありません。 • style=certificateの場合は、証明書を使用して、ユーザ名とパスワードを必要とせずにサーバに接続しようとするクライアントを認証できます。この形式は、転送タイプを内部的にHTTPSに設定します。 SetClientCertificate() メソッドを使用して、サーバがクライアントを認証するための証明書を指定できます。 サーバのIDを正しく認証するには、サーバの証明書とホスト名の検証が必要です。この形式を使用すると、サーバ証明書（ホスト名を含む）の検証がデフォルトで有効になります。 HostnameVerification プロパティを使用すると、ホスト名の検証を無効にできます。サーバ証明書の検証は、 ServerCertificateVerification プロパティを使用して無効にできます。 注： HOSTSEQUIV形式とRPC形式はONTAP APIにのみ適用され、証明書形式はUnified ManagerサーバのAPI通信にのみ適用されます。	NaServer
NaServer.transport_T型TransportType	転送タイプを取得します。設定することも可能です。転送タイプはTRANSPORT_TYPE 値 SERVER_TYPE.FILER およびのデフォルトの転送タイプはHTTPです SERVER_TYPE.DFM。 で SERVER_TYPE.OCUMサポートされる転送タイプはHTTPSのみで、デフォルトで設定されています。	NaServer

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
ServerCertificateVerification	このプロパティは、クライアントによるサーバ証明書の検証を有効または無効にします。 この値を設定する前に、転送タイプをHTTPSに設定する必要があります。 **注：** ホスト名の検証は、サーバ証明書の検証中にデフォルトで有効になっています。認証形式がcertificateに設定されている場合、サーバ証明書の検証はデフォルトで有効になります。 **注:** サーバー証明書の検証に使用する信頼されたルート証明書は、Windows証明書ストアの[信頼されたルート証明機関]フォルダに追加する必要があります。	NaServer
ホスト名の検証	このプロパティは、サーバ証明書の検証中のホスト名の検証を有効または無効にします。この値を設定する前に、サーバ証明書の検証を有効にする必要があります。 **注：** 認証形式がcertificateに設定されている場合、ホスト名の検証はデフォルトで有効になります。ホスト名の検証により、クライアントが接続するホスト名が、SSL接続の一部としてサーバが送り返す証明書のホスト名（CN名）と一致していることが確認されます。 **注：** ホスト名の検証では、アプリケーションがNaServerコンストラクタでサーバに接続するIPアドレスではなく、ホスト名（CN名）を指定する必要があります。	NaServer
SetClientCertificate（文字列証明書）	SSL接続中にサーバが認証するクライアント証明書の場所を設定します。	NaServer
AddNewEncryptedChild（Stリング名、文字列の内容）	コンテンツに含まれるデータを暗号化します。指定された名前と暗号化されたコンテンツを持つ新しい子要素を追加します。	NaElement
GetChildEncryptContent（Stリング名）	指定された名前の子を検索し、その文字列コンテンツを復号化して、復号化されたコンテンツを返します。	NaElement
GetEncryptContent()	この要素のコンテンツの値を復号化して返します。	NaElement
エンコード（char[]バイト）	Base16を使用して一部のバイトをエンコードします。	ベース16
decode（文字列テキスト）	Base16文字の文字列をバイトにデコードします。	ベース16
初期化	暗号化エンジンを初期化します。 **注:** このメソッドは、一意の暗号化または復号化操作の前に呼び出します。	ARCFOUR
crypt（char[]入力、char[]出力）	暗号化または復号化します。これは、データのストリームを暗号化するために複数回呼び出すことができます。 **注:** `init()` 一意の暗号化または復号化操作の前に必ずを呼び出してください。	ARCFOUR
EncryptAndEncode（文字列入力）	暗号化とBase16の両方がデフォルトキーを使用して文字列をエンコードする便利なメソッド。	ARCFOUR
DecodeAndDecrypt（文字列入力）	`encryptAndEncode()` デフォルトのキーを使用してエンコードを元に戻し、から暗号化する簡易メソッド。	ARCFOUR

© 2023 NetApp, Inc. All rights reserved.

表7）C言語の関数の詳細

API名	説明	名前：
shttpc_new (shttpc_type_t sh_type, int timeout)	ソケットを作成し（IPv4ソケットのみ）、それを `sh_socket` メンバーに割り当てます。発信者は、`shttpc_delete` ソケットを閉じてメモリを解放するためにを呼び出す必要があります。	shttpc.c
shttpc_new_ipv6 (shttpc_type_t sh_type, int timeout, struct addrinfo*addrinfo)	addrinfoに基づいてIPv4/IPv6ソケットを作成します。発信者は、`shttpc_delete` ソケットを閉じてメモリを解放するためにを呼び出す必要があります。	shttpc.c
shttpc_delete（shttpc_t sock）shttpc_delete（shttpc_t sock）	ソケットを閉じて `shttpc` オブジェクトを削除します。	shttpc.c
shttpc_setsock_timeout（shttpc_t sock）	ソケットタイムアウトを設定し、ソケットオプションも設定します。	shttpc.c
shttpc_can_retry_on_error（int rc）	致命的でないエラーで再試行できるかどうかを確認します。	shttpc.c
shttpc_connect_status（int rc）	接続エラーステータスを抽出します。	shttpc.c
shttpc_get_connect_error（shttpc_t sock、uint64_t endtime、struct timeval * tv）	接続エラーの詳細を抽出します。	shttpc.c
shttpc_certificate_passwd_cb（char * buf、int size、int rwflag、void * passwd）	暗号化を使用してPEM証明書をロードするときに呼び出されるデフォルトのパスワードコールバック。	shttpc.c
shttpc_verify_server_certificate（SSL * SSL、cert_auth_info * cert_info）	ピア証明書の共通名を特定のホスト名と照合する機能。CNが指定されたホスト名と一致した場合は0を返し、一致しなかった場合は0を返します。[1]	shttpc.c
shttpc_get_verify_cert_error_string（long err_no）	指定された証明書検証エラーについて、人間が判読可能なエラー文字列を返します。	shttpc.c
shttpc_connect_ssl_with_cert（shttpc_t sock、uint64_t endtime、struct timeval * tv、cert_auth_info * cert_info）	SSL接続を確立します。OpenSSLライブラリを使用します。この関数はラッパーオーバーで `shttpc_connect_ipv6()` あり、追加の証明書の詳細を入力として受け取ります。	shttpc.c
shttpc_connect(shttpc_t sock, const struct sockaddr*addr, int addrlen)	プレーンTCP / SSLを使用してホストに接続します。OpenSSLライブラリを使用してSSLを使用して接続します。	shttpc.c
shttpc_connect_ipv6_with_cert（shttpc_t sock、struct addrinfo * addr、size_t addrlen、cert_auth_info * cert_info）	この関数はラッパーオーバーで `shttpc_connect_ipv6()` あり、追加の証明書の詳細を入力として受け取ります。	shttpc.c
shttpc_connect_ipv6（shttpc）_t sock、struct addrinfo * addr、size_t addrlen）	プレーンTCP / SSLを使用してホストに接続します。OpenSSLライブラリを使用してSSLを使用して接続します。	shttpc.c
shttpc_read（shttpc_t sock、void * buf、size_t len）	ソケットをバッファに読み込み `buf`ます。 最大バッファサイズは、SSLを使用して接続する場合はlenです。	shttpc.c
shttpc_write（shttpc_t sock、const void * buf、size_t len）	ソケットにデータを書き込みます。データ全体が書き込まれるまで試行を続けます。	shttpc.c
HTTP_PARSE_URL(const char * url、http_url_t * PURL)	URLをコンポーネントに分割します。	HTTP.c

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
HTTP_FREE_URL（http_url_t * PURL）	で割り当てられているスペースの割り当てを解除し `http_parse_url()` ます。	HTTP.c
bool_t http_write（shttpc_t sock、const void * line、size_t len）	ソケットに行を書き込みます。成功した場合はTrueを返します。	HTTP.c
bool_t http_print(shttpc_t sock, const char *fmt,...)	ソケットに行をフォーマットして書き込みます。成功した場合はTrueを返します。	HTTP.c
bool_t http_getline（shttpc_t sock、char * line、int len）	ソケットからバッファに行を読み込みます。`\r\n` 各行の最後にあるを取り除きます。 成功した場合はTrueを返します。	HTTP.c
http_strip_headers（shttpc_t sock、stab_t * headers）	ヘッダーを読んで、それらを刺して保存します。 エラー（ENOMEMなど）が発生した場合、戻り値は負です。 **注：** 成功すると、HTTPステータスを返します。statusがHTTP_OKの場合、ヘッダーの値が入力されます。この値を解放する必要があります（stab_deleteなど）。ステータスがエラーを示している場合は、ヘッダを解放することを心配する必要はありません。	HTTP.c
HTTP_BIND_SOCKET(const shttpc_t socket, uint16_t reservedPort, const struct addrinfo*addrinfo)	ソケットを予約ポートおよびファミリ（IPv4またはIPv6）にバインドします。この関数は、指定されたポートからポート1への使用可能な予約済みポートのいずれかにソケットをバインドしようとします。成功した場合は0、失敗した場合はEPERMエラーを返します。	HTTP.c
http_open_socket_reserved_ex_wt_wcert（const char * host、uint16_t port、shttpc_t * socketP、bool_t reserved、shttpc_type_t conn_type、 int timeout、cert_auth_info * cert_info、bool_t use_sslv3）	この関数はラッパーオーバーで `http_open_socket_reserved()` あり、追加の証明書の詳細を入力として受け取ります。	HTTP.c
bool_t http_get_auth（const AuthInfo * auth、char ** encoded_str）	認証メッセージを取得します。	HTTP.c
HTTP_POST_REQUEST (shttpc_t sock, const char *url, const AuthInfo*auth_info, const void *post_data, size_t post_data_len, stab_t *headersp)	`http_post_request` は、のHTTP POSTラッパーです。『`http_method_request()`』 **注：** パラメータの詳細については、を参照してください `http_method_request()`。 エラーが発生した場合は負の値になり、すべて問題がなければ0になります。	HTTP.c
HTTP_GET_REQUEST（shttpc_t sock、const char * url、const AuthInfo * auth_info、stab_t * headersp）	HTTP_GET_REQUESTは、http_METHOD_REQUEST ()のHTTP GETラッパーです。 詳細については、`http_method_request()` を参照してください。エラーが発生した場合は負の値になり、すべて問題がなければ0になります。	HTTP.c
bool_t http_read_chunk（shttpc_t sock、char ** pbuf、size_t * pread）	サーバーから1つのデータチャンクを読み取ります。`pbuf` 読み取られたデータが格納されます。preadにはソケットから読み取られたバイト数が格納されます。 成功した場合はTrue、失敗した場合はFalseを返します。	HTTP.c

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
na_startup(char * errbuff、int errbuffsize)	APIの準備のための1回限りの起動。	na.c
na_startup_without_ntapadmin（char * errbuff、int errbuffsize）	APIやネットワークの準備をするための1回限りのスタートアップ。この関数は、 `na_startup ntapadmin` ライブラリのロードを無効にするAPIを内部的に呼び出します。 これは、WindowsシステムでのRPC通信に使用されます。UNIXシステムでは、単に `na_startup` APIを呼び出します。	na.c
na_server_set_timeout（na_server_t * s、int timeout）	接続を行うとき、またはソケットからデータを読み書きするときに、サーバーのタイムアウトを設定します。	na.c
na_server_set_server_type（na_server_t * s、na_server_type_t type）	filer、dfm、またはocumのいずれかのキーワードを渡して、サーバがストレージシステム（Filer）サーバかActive IQ Unified Manager（旧OnCommand Unified Manager）サーバかを示します。 も使用する場合 `set_port()`は、 `set_port()` このルーチンを呼び出した後にを呼び出します。 デフォルト値はFILERです。	na.c
na_server_set_style（na_server_t * s、na_style_t style）	• 原因サーバにログインを渡し、ユーザ名とパスワードを使用したHTTPシンプル認証を使用します。 • `hosts.equiv` Filer上のファイルを使用してアクセス権を決定するホストを渡します（`root` その場合は、ユーザ名を指定する必要があります）。 • 証明書を渡して、Unified Managerサーバで証明書ベースの認証を使用します。 **注:** style=certificateの場合、証明書を使用して、ユーザ名とパスワードを必要とせずにサーバに接続しようとするクライアントを認証できます。この形式は、転送タイプを内部的にHTTPSに設定します。サーバのIDを正しく認証するには、サーバの証明書の検証が必要です。この形式を使用すると、サーバ証明書の検証がデフォルトで有効になり、サーバ証明書の検証では常にホスト名の検証が有効になります。を使用して、（ホスト名を含む）サーバ証明書の検証を無効にすることができます `set_server_cert_verification()`。	na.c
na_server_set_transport（na_server_t * s、na_server_transport_t transport、const union ZfD_setopt * transportarg、int port）	デフォルトの転送タイプをオーバーライドします。現在の有効な転送タイプはHTTPとHTTPSです。	na.c
na_server_transport_t na_server_get_transport_tタイプ（na_server_t * s）	この接続に使用される転送タイプを取得します。	na.c
na_server_set_port（na_server_t * s、int port）	このサーバのデフォルトポートを上書きします。また電話をかける場合は `set_server_type()`、電話をかける前に電話をかける必要があり `set_port()` ます。	na.c
na_server_get_port（na_server_t * s）	リモートサーバに使用されているポートを取得します。	na.c
na_server_close（na_server_t * s）	サーバからの接続を閉じます。	na.c

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
na_encrypt_basic（const char * key、const char * input、char * output、size_t nバイト）	入力値（一連のバイト）を取得し、暗号化されたバージョンを生成します。	na.c
na_elem_t * na_child_add_string_encryPTED（na_elem_t * n、const char * name、const char * contents、const char * key）	と同じ `child_add_string`ですが、 `value key` 現在のオブジェクトに要素を追加する前にで暗号化します。 これは現在、特定の鍵交換操作にのみ使用されています。クライアントとサーバーの両方がの値を知ってい `key` て、このルーチンとそのコンパニオンを使用することに同意する必要があります `child_get_string_encrypted()`。 指定されたキーがNoneの場合、デフォルトのキーが使用されます。	na.c
char * na_child_get_string_encryPTED（na_elem_t * n、const char * name、const char * key）	指定された子の値を取得し `name key` 、で復号化してから返します。 指定したキーがNoneの場合は、デフォルトのキーが使用されます。	na.c
na_elem_t * na_zapi_get_result（na_elem_stack_t * elemStack）	呼び出されたzapiの結果を取得します。	na.c
na_server_init_cert_info（na_server_t * srv）	指定されたサーバコンテキストの証明書認証構造のメモリを初期化します。	na.c
na_server_set_client_cert_and_key（na_server_t * srv、const char * cert_file、const char * key_file、const char * key_passwd）	証明書を使用するサーバによるクライアント認証に必要なクライアント証明書とキーファイルを設定します。キーファイルが定義されていない場合は、証明書ファイルがキーファイルとして使用されます。	na.c
na_server_set_server_cert_verification（na_server_t * srv、int enable）	クライアントによるサーバ証明書の検証を有効または無効にします。styleがcertificateに設定されている場合、サーバ証明書の検証はデフォルトで有効になります。ホスト名（CN）の検証は、サーバ証明書の検証中に有効になります。 `set_hostname_verification()` APIを使用して、ホスト名の検証を無効にできます。	na.c
na_server_is_server_cert_verification_enabled （ const na_server_t * srv）	サーバ証明書の検証を有効にするかどうかを指定します。有効になっている場合はTrueを返し、有効になっていない場合はFalseを返します。	na.c
na_server_set_ca_certs（na_server_t * srv、const char * CAfile）	このアプリケーションによって信頼され、サーバ証明書の検証に使用されるCAの証明書を指定します。	na.c
na_server_set_hostname_verification（na_server_t * srv、int enable）	サーバ証明書の検証中のクライアントによるホスト名検証を有効または無効にします。	na.c
na_server_is_hostname_verification_enabled（const na_server_t * srv）	ホスト名の検証を有効にするかどうかを指定します。有効になっている場合はTrueを返し、有効になっていない場合はFalseを返します。	na.c
na_server_set_sslv3（na_s server_t * srv、int enable）	HTTPS転送で使用するSSLv3プロトコルを有効または無効にします。デフォルトでは、SSLv3プロトコルは無効になっています。	na.c
na_server_is_sslv3_enabled（na_server_t * srv）	HTTPS転送でSSLv3プロトコルを使用できるかどうかを指定します。SSLv3が有効な場合は1を返し、有効でない場合は0を返します。	na.c

© 2023 NetApp, Inc. All rights reserved.

API名	説明	名前：
do_encrypt_test（char * input、int nバイト）	2回暗号化されたパスワードが、開始時と同じ値で出力されることを確認します。	na_test.c
main_encryptalot（int argc、char * argv[]）	ランダムなパスワード値を使用した暗号化テスト-パスワードの数と長さを制御できます。	na_test.c
nc_api_error_t nc_api_set_transport（nc_api_transport_t transport、const ZfD_setopt_t * opt）	転送タイプを設定します。有効な値は次のとおりです。 • `NC_API_TRANSPORT_HTTP` • `NC_API_TRANSPORT_HTTPS`	nc_api.c
nc_api_error_t nc_api_read_file（const char * host、int port、const authInfo * auth、const char * file、char ** out）	指定されたリモートファイルをバッファにコピーします。呼び出し元は、終了したときに指定されたファイルとバッファのメモリを解放する必要があります。	nc_api.c

詳細情報の入手方法

このドキュメントに記載されている情報の詳細については、以下のドキュメントやWebサイトを確認してください。

- NetAppの製品ドキュメント
  https://www.netapp.com/support-and-training/documentation/

バージョン履歴

バージョン	日付	ドキュメントの改訂履歴
バージョン1.0	2022年11月1日	初版リリース
バージョン2.0	2023年5月	NMSDKログの詳細の更新

© 2023 NetApp, Inc. All rights reserved.

本ドキュメントに記載されている製品や機能のバージョンがお客様の環境でサポートされるかどうかについては、NetApp サポート サイトで Interoperability Matrix Tool (IMT)を参照してください。NetApp IMT には、NetApp がサポートする構成を構築するために使用できる製品コンポーネントやバージョンが定義されています。サポートの可否は、お客様の実際のインストール環境が公表されている仕様に従っているかどうかによって異なります。

機械翻訳に関する免責事項

原文は英語で作成されました。英語と日本語訳の間に不一致がある場合には、英語の内容が優先されます。公式な情報については、本資料の英語版を参照してください。翻訳によって生じた矛盾や不一致は、法令の順守や施行に対していかなる拘束力も法的な効力も持ちません。

著作権に関する情報

Copyright © 2023 NetApp, Inc. All Rights Reserved. Printed in the U.S.このドキュメントは著作権によって保護されています。著作権所有者の書面による事前承諾がある場合を除き、画像媒体、電子媒体、および写真複写、記録媒体、テープ媒体、電子検索システムへの組み込みを含む機械媒体など、いかなる形式および方法による複製も禁止します。

NetApp の著作物から派生したソフトウェアは、次に示す使用許諾条項および免責条項の対象となります。

このソフトウェアは、NetApp によって「現状のまま」提供されています。NetApp は明示的な保証、または商品性および特定目的に対する適合性の暗示的保証を含み、かつこれに限定されないいかなる暗示的な保証も行いません。NetApp は、代替品または代替サービスの調達、使用不能、データ損失、利益損失、業務中断を含み、かつこれに限定されない、このソフトウェアの使用により生じたすべての直接的損害、間接的損害、偶発的損害、特別損害、懲罰的損害、必然的損害の発生に対して、損失の発生の可能性が通知されていたとしても、その発生理由、根拠とする責任論、契約の有無、厳格責任、不法行為（過失またはそうでない場合を含む）にかかわらず、一切の責任を負いません。

NetApp は、ここに記載されているすべての製品に対する変更を随時、予告なく行う権利を保有します。NetApp による明示的な書面による合意がある場合を除き、ここに記載されている製品の使用により生じる責任および義務に対して、NetApp は責任を負いません。この製品の使用または購入は、NetApp の特許権、商標権、または他の知的所有権に基づくライセンスの供与とはみなされません。

このマニュアルに記載されている製品は、1 つ以上の米国特許、その他の国の特許、および出願中の特許により保護されている場合があります。

本書に含まれるデータは市販の製品および / またはサービス（FAR 2.101 の定義に基づく）に関係し、データの所有権は NetApp, Inc.にあります。米国政府は本データに対し、非独占的かつ移転およびサブライセンス不可で、全世界を対象とする取り消し不能の制限付き使用権を有し、本データの提供の根拠となった米国政府契約に関連し、当該契約の裏付けとする場合にのみ本データを使用できます。前述の場合を除き、NetApp, Inc.の書面による許可を事前に得ることなく、本データを使用、開示、転載、改変するほか、上演または展示することはできません。国防総省にかかる米国政府のデータ使用権については、DFARS 252.227-7015(b)項で定められた権利のみが認められます。

商標に関する情報

NetApp、NetAppのロゴ、https://www.netapp.com/company/legal/trademarks/に記載されているマークは、NetApp, Inc.の商標です。その他の会社名と製品名は、それを所有する各社の商標である場合があります。

TR-4945-0523-JP

NetApp