外部設定ファイルの認証

外部設定ファイルの認証

Cisco ESC リリース 4.0 以前では、ESC は、デイゼロ設定、モニタリング、展開、および LCS アクションの一部として、いくつかの外部設定ファイルとスクリプトをサポートしています。ESC は、展開の一部として、認証の有無にかかわらず、リモートサーバからのこれらのファイルの取得をサポートしています。

ESC リリース 4.0 以降、ファイルロケータ属性は展開レベル、つまり展開コンテナの直下で定義されます。これにより、複数の VM グループとそのデイゼロ設定および LCS アクションが、展開内の必要な場所で同じファイルロケータを参照できるようになります。

展開データモデルの例は次のとおりです。

```
<esc_datamodel xmlns="http://www.cisco.com/esc/esc">
  <tenants>
    <tenant>
      <name>sample-tenant</name>
      <deployments>
        <deployment>
          <name>sample-deployment</name>
          <file_locators>
            <file_locator>
              <name>post_deploy_alive_script</name>
              <remote_file>
                <file_server_id>http-my-server</file_server_id>
              <remote_path>/share/qatest/vnfupgrade/lcspostdeployalive.sh</remote_path>

                <local_target>vnfupgrade/lcspostdepalive.sh</local_target>
                <persistence>FETCH_ALWAYS</persistence>
                <properties/>
              </remote_file>
            </file_locator>
            <file_locator>
              <name>asa-day0-config</name>
              <remote_file>
                <file_server_id>http-my-server</file_server_id>
                <remote_path>/share/qatest/day0/asa_config.sh</remote_path>
                <local_target>day0.1/asa_config.sh</local_target>
```

外部設定ファイルの認証

```
        <persistence>FETCH_ALWAYS</persistence>
      </remote_file>
    </file_locator>
    <file_locator>
      <name>scriptlocator</name>
      <remote_file>
        <file_server_id>dev_test_server</file_server_id>
        <remote_path>/share/users/gomoore/actionScript.sh</remote_path>
        <local_target>action/actionScript.sh</local_target>
        <persistence>FETCH_MISSING</persistence>
        <properties/>
      </remote_file>
    </file_locator>
  </file_locators>
  <policies>
    <policy>
      <name>VNFUPGRADE_POST_DEPLOY_ALIVE</name>
      <conditions>
        <condition>
          <name>LCS::POST_DEPLOY_ALIVE</name>
        </condition>
      </conditions>
      <actions>
        <action>
          <name>post_deploy_alive_action</name>
          <type>SCRIPT</type>
          <properties>
            <property>
              <name>file_locator_name</name>
              <value>post_deploy_alive_script</value>
            </property>
          </properties>
        </action>
      </actions>
    </policy>
  </policies>
<vm_group>
    <name>ASA-group</name>
    <image>ASAImage</image>
    <flavor>m1.large</flavor>
    <recovery_policy>
      <max_retries>1</max_retries>
    </recovery_policy>
    <scaling>
      <min_active>1</min_active>
      <max_active>1</max_active>
      <elastic>true</elastic>
    </scaling>
    <placement>
      <type>affinity</type>
      <enforcement>strict</enforcement>
    </placement>
    <bootup_time>120</bootup_time>
    <recovery_wait_time>60</recovery_wait_time>
    <interfaces>
      <interface>
        <nicid>0</nicid>
        <network>my-net</network>
      </interface>
    </interfaces>
    <kpi_data>
      <kpi>
        <event_name>VM_ALIVE</event_name>
        <metric_value>1</metric_value>
```

外部設定ファイルの認証

```
                    <metric_cond>GT</metric_cond>
                    <metric_type>UINT32</metric_type>
                    <metric_occurrences_true>1</metric_occurrences_true>
                    <metric_occurrences_false>5</metric_occurrences_false>
                    <metric_collector>
                      <nicid>0</nicid>
                      <type>ICMPPing</type>
                      <poll_frequency>5</poll_frequency>
                      <polling_unit>seconds</polling_unit>
                      <continuous_alarm>false</continuous_alarm>
                    </metric_collector>
                  </kpi>
                </kpi_data>
                <rules>
              <admin_rules>
                    <rule>
                      <event_name>VM_ALIVE</event_name>
                      <action>ALWAYS log</action>
                      <action>TRUE servicebooted.sh</action>
                      <action>FALSE recover autohealing</action>
                    </rule>
                  </admin_rules>
                </rules>
                <config_data>
                  <configuration>
                    <dst>ASA.static.txt</dst>
                    <file_locator_name>asa-day0-config</file_locator_name>
                  </configuration>
                </config_data>
                <policies>
                  <policy>
                    <name>SVU1</name>
                    <conditions>

<condition><name>LCS::DEPLOY_UPDATE::PRE_VM_VOLUME_DETACH</name></condition>
                    </conditions>
                    <actions>
                      <action>
                        <name>LOG</name><type>pre_defined</type>
                      </action>
                      <action>
                        <name>pre_vol_detach</name>
                        <type>SCRIPT</type>
                        <properties>
                          <property>
                            <name>file_locator_name</name>
                            <value>scriptlocator</value>
                          </property>
                          <property>
                            <name>exit_val</name>
                            <value>0</value>
                          </property>
                        </properties>
                      </action>
                    </actions>
                  </policy>
                </policies>
              </vm_group>
            </deployment>
          </deployments>
        </tenant>
      </tenants>
    </esc_datamodel>
```

外部設定ファイルの認証

展開を実行する前に、APIを使用してリモートサーバ（ファイルサーバ）を個別に設定する必要があります。REST API と NETCONF API の両方がサポートされます。

- URL、ユーザ名を含む認証の詳細、およびパスワードを含むリモートサーバ。設定には REST または NETCONF を使用できます。

  （注） ユーザ名とパスワードはオプションです。パスワードはESC内で暗号化されます。

  展開前にリモートファイルサーバを設定する必要があります。クレデンシャルは、展開中にいつでも更新できます。

- ファイルロケータが展開データモデルに追加されます。ファイルサーバへの参照と、ダウンロードするファイルへの相対パスが含まれます。

認証を使用してリモートでファイルを取得するには、以下を行う必要があります。

1. リモートサーバを追加します。
2. ファイルロケータでリモートサーバを参照します。ファイルロケータは、デイゼロおよびLCS アクションブロックの設定データの一部です。
3. 展開の一部として、ファイルロケータに基づいてデイゼロおよびライフサイクルステージ（LCS）スクリプトが取得されます。

ファイルサーバのパラメータは次のとおりです。

- id：ファイルサーバのキーと識別子として使用されます。
- base_url：サーバのアドレス。（例：http://www.cisco.com または https://192.168.10.23）
- file_server_user：サーバへの認証時に使用するユーザ名。
- file_server_password：サーバへの認証用のパスワードを含む文字列。最初に、ユーザは内部で暗号化されたクリアテキスト文字列を指定します。
- properties：将来の拡張性のための名前と値のペア。

ファイルロケータのパラメータは次のとおりです。

- name：ファイルロケータのキーおよび識別子として使用されます。
- local_file または remote_file：ファイルの場所を選択します。ローカルファイルは、ESC VM ファイルシステムにすでに存在するファイルを指定するために使用されます。remote_file は、リモートサーバから取得するファイルを指定するために使用されます。
  - file_server_id：ファイルを取得するファイルサーバオブジェクトの ID。
  - remote_path：ファイルサーバオブジェクトで定義された base_url からのファイルのパス。

外部設定ファイルの認証

- local_target：ファイルを保存するためのオプションのローカル相対ディレクトリ。
- properties：必要な情報の名前と値のペア。
- persistence：ファイルストレージのオプション。値には、CACHE、FETCH_ALWAYS、および FETCH_MISSING（デフォルト）が含まれます。

- checksum：転送されるファイルの有効性を検証するために使用する、オプションの BSD スタイルのチェックサム値。

サーバ接続、ファイルの存在、チェックサムなどのファイルサーバ値の有効性が検証されます。

file_server_password フィールドとプロパティの encrypted_data フィールドの encrypted_data 値は、伝送用 AES / 128 ビットを使用して CFB モードで暗号化されます。データは、サーバへのアクセスに必要になるまで暗号化されたままになります。暗号化された値の詳細については、「設定データの暗号化」を参照してください。

ファイルサーバの例

```
<esc_datamodel xmlns="http://www.cisco.com/esc/esc">
  <file_servers>
    <file_server>
      <id>server-1</id> <!-- unique name for server -->
      <base_url>https://www.some.server.com</base_url>
      <file_server_user>user1</file_server_user>
      <file_server_password>sample_password</file_server_password>
      <!-- encrypted value -->
      <!-- properties list containing additional items in the future -->
      <properties>
        <property>
          <name>server_timeout</name>
          <value>60</value>
      <!-- timeout value in seconds, can be over-ridden in a file_locator -->
        </property>
      </properties>
    </file_server>
    <file_server>
      <id>server-2</id>
      <base_url>https://www.some.other.server.com</base_url>
          <properties>
        <property>
          <name>option1</name>
          <encrypted_value>$8$EADFAQE</encrypted_value>
        </property>
    </file_server>
  </file_servers>
</esc_datamodel>
```

デイゼロ設定の例

```
<esc_datamodel xmlns="http://www.cisco.com/esc/esc">
  <tenants><tenant>
      <name>sample-tenant</name>
      <deployments><deployment>
          <name>sample-deployment</name>
          <vm_group>
            <name>sample-vm-group</name>
            <config_data>
              <!-- exisiting configuration example - remains valid -->
```

外部設定ファイルの認証

```
                <configuration>
                  <file>file:///cisco/config.sh</file>
                  <dst>config.sh</dst>
                </configuration>
                <!-- new configuration including use of file locators -->
                <configuration>
                  <dst>something</dst>
                  <file_locators>
                    <file_locator>
                      <name>configlocator-1</name> <!-- unique name -->
                      <remote_file>
                        <file_server_id>server-1</file_server_id>
                        <remote_path>/share/users/configureScript.sh</remote_path>
                        <!-- optional user specified local silo directory -->
                        <local_target>day0/configureScript.sh</local_target>
                        <!-- persistence is an optional parameter -->
                        <persistence>FETCH_ALWAYS</persistence>
                        <!-- properties in the file_locator are only used for
                             fetching the file not for running scripts -->
                        <properties>
                          <property>
                            <!-- the property name "configuration_file" with value "true"
 indictates this is the
                                 script to be used just as using the <file> member case
of the configuration -->
                            <name>configuration_file</name>
                            <value>true</value>
                          </property>
                          <property>
                            <name>server_timeout</name>
                            <value>120</value> <!-- timeout value in seconds, overrides
the file_server property -->
                          </property>
                          </properties>
                        </remote_file>
                        <!-- checksum is an optional parameter.
                           The following algorithms are supported: SHA-1, SHA-224, SHA-256,
 SHA-384, SHA-512 -->
                        <checksum>SHA256 (configureScript.sh) =
dd526bb2c0711238ec2649c4b91598fb9a6cf1d2cb8559c337c5f3dd5ea1769e</checksum>
                      </file_locator>
                      <file_locator>
                        <name>configlocator-2</name>
                        <remote_file>
                          <file_server_id>server-2</file_server_id>
                          <remote_path>/secure/requiredData.txt</remote_path>
                          <local_target>day0/requiredData.txt</local_target>
                          <persistence>FETCH_ALWAYS</persistence>
                          <properties/>
                        </remote_file>
                      </file_locator>
                    </file_locators>
                  </configuration>
                </config_data>
              </vm_group>
            </deployment></deployments>
        </tenant></tenants>
</esc_datamodel>
```

デイゼロ設定およびLCSアクションの詳細については、「デイゼロ設定」および「再展開ポリシー」の項を参照してください。

設定データの暗号化

秘密キーと秘密情報を使用して設定データを暗号化できます。ESC では、デイゼロ設定、デイゼロ設定変数、VIM コネクタと VIM ユーザ、および LCS アクションに秘密キーが含まれています。

ConfD は、暗号化された文字列タイプを提供します。組み込みの文字列タイプを使用すると、暗号化された値が ConfD に保存されます。値の暗号化に使用されるキーは、confd.conf に保存されます。

データの暗号化はオプションです。必要に応じて、encrypt_data 値を使用してデータを保存できます。

次の例では、デイゼロ設定データに暗号化された値が含まれています。encrypted_data は組み込みの文字列タイプ *tailf:aes-cfb-128-encrypted-string* を使用します。

```
choice input_method {
  case file {
    leaf file {
      type ietf-inet-types:uri;
    }
  }
  case data {
    leaf data {
      type types:escbigdata;
    }
  }
  case encrypted_data {
    leaf encrypted_data {
      type tailf:aes-cfb-128-encrypted-string;
    }
  }
}
```

Advanced Encryption Standard（AES）キーの生成

AES キーの長さは 16 バイトで、32 文字の 16 進数文字列が含まれています。

暗号化を機能させるには、confd.conf で AES キーを設定する必要があります。

```
/opt/cisco/esc/esc-confd/esc_production_confd.conf
```

```
<encryptedStrings>
    <AESCFB128>
        <key>0123456789abcdef0123456789abcdef</key>
        <initVector>0123456789abcdef0123456789abcdef</initVector>
    </AESCFB128>
</encryptedStrings>
```

デフォルトの AES キーは confD で使用できます。

```
0123456789abcdef0123456789abcdef
```

confD キーはハードコードされています。escadm.py はランダムな AES キーを生成し、confD が開始する前にデフォルトの confD AES キーを置き換えます。