管理

ここでは、次の内容について説明します。

ユーザーの管理

すべてのユーザーが管理者ロールを持ちます。次の手順では、ユーザーを作成、変更、および削除する方法について説明します。

ステップ 1 WAE UI から、[ユーザー管理（User Manager）] アイコン（ ）をクリックします。

ステップ 2 ユーザーを追加するには、 をクリックして、該当するすべてのフィールドを入力します。

ステップ 3 ユーザーのパスワードを変更するには、次の手順を実行します。

a) ユーザーの行を選択して をクリックします。

b) パスワードフィールドを更新します。

c) [保存（Save）] をクリックします。

ステップ 4 ユーザーを削除するには、ユーザーの行をクリックしてから  をクリックします。

エージングの構成

デフォルトでは、回路、ポート、ノード、またはリンクがネットワークから消失すると、永久に削除され、再検出する必要があります。消失したこれらの要素を WAE が保持してからネットワークから完全に削除されるまでの期間を設定するには、次の手順を実行します。

（注） これは、すべてのネットワークに構成されるグローバルオプションです。

ステップ 1 エキスパート モードから、/wae:wae/components/dare:aggregators に移動し、[エージング（aging）] タブを選択します。

- [aging-enabled]：[true] を選択してエージングを有効にします。
- [l3-node-aging-duration]：L3 ノードが非アクティブになった後に、ネットワーク内で保持する必要がある期間を入力します。
- [l3-port-aging-duration]：L3 ポートが非アクティブになった後に、ネットワーク内で保持する必要がある期間を入力します。
- [l3-circuit-aging-duration]：L3 回路が非アクティブになった後に、ネットワーク内で保持する必要がある期間を入力します。

  （注） l3-node-aging-duration の値は l3-port-aging-duration より大きくする必要があり、l3-port-aging-duration の値は l3-circuit-aging-duration より大きくする必要があります。

- [l1-node-aging-duration]：L1 ノードが非アクティブになった後に、ネットワーク内で保持する必要がある期間を入力します。
- [l1-port-aging-duration]：L1 ポートが非アクティブになった後に、ネットワーク内で保持する必要がある期間を入力します。
- [l1-link-aging-duration]：L1 リンクが非アクティブになった後に、ネットワーク内で保持する必要がある期間を入力します。

  （注） l1-node-aging-duration の値は l1-port-aging-duration より大きくする必要があり、l1-port-aging-duration の値は l1-link-aging-duration より大きくする必要があります。

ステップ 2 [確定する（Commit）] をクリックします。

wae.conf

wae.conf は、YANG モデル tailf-ncsconfig.yang で正式に定義されている XML 構成ファイルです。この YANG ファイルは、コメント付きの wae.conf.example ファイルと同様に、WAE ディストリビューションに含まれています。

wae.conf ファイルは、WAE ランタイムの基準設定を制御します。wae.conf ファイルで特定の構成パラメータを変更できます。たとえば、WAE が動作するデフォルトポート（ポート 8080）を別のポートに変更できます。

（注） wae.conf ファイルに変更を加えた後は、必ず WAE を再起動してください。

WAE デーモンを起動またはリロードするたびに、./wae.conf または *<waeruntime-directory>*/etc/wae.conf から構成を読み取ります。

<waeruntime-directory>/etc/wae.conf の内容を次の例に示します。

```
<!-- -*- nxml -*- -->
<!-- Example configuration file for wae. -->

<ncs-config xmlns="http://tail-f.com/yang/tailf-ncs-config">

  <!-- WAE can be configured to restrict access for incoming connections -->
  <!-- to the IPC listener sockets. The access check requires that -->
  <!-- connecting clients prove possession of a shared secret. -->
  <ncs-ipc-access-check>
    <enabled>false</enabled>
    <filename>${NCS_DIR}/etc/ncs/ipc_access</filename>
  </ncs-ipc-access-check>

  <!-- Where to look for .fxs and snmp .bin files to load -->

  <load-path>
    <dir>./packages</dir>
    <dir>${NCS_DIR}/etc/ncs</dir>

    <!-- To disable northbound snmp altogether -->
    <!-- comment out the path below -->
    <dir>${NCS_DIR}/etc/ncs/snmp</dir>
  </load-path>

  <!-- Plug and play scripting -->
  <scripts>
    <dir>./scripts</dir>
    <dir>${NCS_DIR}/scripts</dir>
  </scripts>

  <state-dir>./state</state-dir>

  <notifications>
    <event-streams>

      <!-- This is the builtin stream used by WAE to generate northbound -->
      <!-- notifications whenever the alarm table is changed. -->
      <!-- See tailf-ncs-alarms.yang -->
      <!-- If you are not interested in WAE northbound netconf notifications -->
```

```
<!-- remove this item since it does consume some CPU -->
<stream>
  <name>wae-alarms</name>
  <description>WAE alarms according to tailf-ncs-alarms.yang</description>
  <replay-support>false</replay-support>
  <builtin-replay-store>
    <enabled>false</enabled>
    <dir>./state</dir>
    <max-size>S10M</max-size>
    <max-files>50</max-files>
  </builtin-replay-store>
</stream>

<!-- This is the builtin stream used by WAE to generate northbound -->
<!-- notifications for internal events. -->
<!-- See tailf-ncs-devices.yang -->
<!-- Required for cluster mode. -->
<stream>
  <name>wae-events</name>
  <description>WAE event according to tailf-ncs-devices.yang</description>
  <replay-support>true</replay-support>
  <builtin-replay-store>
    <enabled>true</enabled>
    <dir>./state</dir>
    <max-size>S10M</max-size>
    <max-files>50</max-files>
  </builtin-replay-store>
</stream>

<!-- This is the builtin stream used by WAE to generate northbound -->
<!-- notifications for kicker event stream. -->
<!-- See tailf-kicker.yang -->
<!-- Required for cluster mode. -->
<stream>
  <name>kicker-events</name>
  <description>NCS event according to tailf-kicker.yang</description>
  <replay-support>true</replay-support>
  <builtin-replay-store>
    <enabled>true</enabled>
    <dir>./state</dir>
    <max-size>S10M</max-size>
    <max-files>50</max-files>
  </builtin-replay-store>
</stream>

<!-- This is the builtin stream used by WAE to generate northbound -->
<!-- notifications forwarded from devices. -->
<!-- See tailf-event-forwarding.yang -->
<stream>
  <name>device-notifications</name>
  <description>WAE events forwarded from devices</description>
  <replay-support>true</replay-support>
  <builtin-replay-store>
    <enabled>true</enabled>
    <dir>./state</dir>
    <max-size>S10M</max-size>
    <max-files>50</max-files>
  </builtin-replay-store>
</stream>

<!-- This is the builtin stream used by WAE to generate northbound -->
<!-- notifications for plan state transitions. -->
<!-- See tailf-ncs-plan.yang -->
<stream>
```

```
        <name>service-state-changes</name>
        <description>Plan state transitions according to
        tailf-ncs-plan.yang</description>
        <replay-support>false</replay-support>
        <builtin-replay-store>
          <enabled>false</enabled>
          <dir>./state</dir>
          <max-size>S10M</max-size>
          <max-files>50</max-files>
        </builtin-replay-store>
      </stream>
      <stream>
        <name>XtcNotifications</name>
        <description>Xtc object change notifications</description>
        <replay-support>false</replay-support>
      </stream>
    </event-streams>
  </notifications>

  <!-- Where the database (and init XML) files are kept -->
  <cdb>
    <db-dir>./ncs-cdb</db-dir>
    <!-- Always bring in the good system defaults -->
    <init-path>
      <dir>${NCS_DIR}/var/ncs/cdb</dir>
    </init-path>
  </cdb>


  <!--&#xa;      These keys are used to encrypt values of the types&#xa;
tailf:des3-cbc-encrypted-string, tailf:aes-cfb-128-encrypted-string&#xa;      and
tailf:aes-256-cfb-128-encrypted-string.&#xa;      For a deployment install it is highly
 recommended to change&#xa;      these numbers to something random (done by WAE "system
 install")&#xa;  -->
  <encrypted-strings>
    <DES3CBC>
      <key1>0123456789abcdef</key1>
      <key2>0123456789abcdef</key2>
      <key3>0123456789abcdef</key3>
      <initVector>0123456789abcdef</initVector>
    </DES3CBC>

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

    <AES256CFB128>
      <key>0123456789abcdef0123456789abcdef0123456789abcdef0123456789abcdef</key>
    </AES256CFB128>
  </encrypted-strings>


  <logs>
    <syslog-config>
      <facility>daemon</facility>
    </syslog-config>

    <ncs-log>
      <enabled>true</enabled>
      <file>
        <name>./logs/wae.log</name>
        <enabled>true</enabled>
      </file>
```

```
    <syslog>
      <enabled>true</enabled>
    </syslog>
  </ncs-log>

  <developer-log>
    <enabled>true</enabled>
    <file>
      <name>./logs/devel.log</name>
      <enabled>true</enabled>
    </file>
  </developer-log>
  <developer-log-level>error</developer-log-level>

  <audit-log>
    <enabled>true</enabled>
    <file>
      <name>./logs/audit.log</name>
      <enabled>true</enabled>
    </file>
  </audit-log>

  <netconf-log>
    <enabled>true</enabled>
    <file>
      <name>./logs/netconf.log</name>
      <enabled>true</enabled>
    </file>
  </netconf-log>

  <snmp-log>
    <enabled>true</enabled>
    <file>
      <name>./logs/snmp.log</name>
      <enabled>true</enabled>
    </file>
  </snmp-log>

  <webui-access-log>
    <enabled>true</enabled>
    <dir>./logs</dir>
  </webui-access-log>


   <!-- This log is disabled by default if wae is installed using -->
  <!-- the 'system-install' flag. It consumes a lot of CPU power -->
  <!-- to have this log turned on, OTOH it is the best tool to -->
  <!-- debug must expressions in YANG models -->

  <xpath-trace-log>
    <enabled>false</enabled>
    <filename>./logs/xpath.trace</filename>
  </xpath-trace-log>

  <error-log>
    <enabled>true</enabled>
    <filename>./logs/wae-err.log</filename>
  </error-log>

  <progress-trace>
    <enabled>true</enabled>
    <dir>./logs</dir>
  </progress-trace>
</logs>
```

```
<ssh>
  <algorithms>
    <kex>diffie-hellman-group14-sha1</kex>
    <mac>hmac-sha2-512,hmac-sha2-256,hmac-sha1</mac>
    <encryption>aes128-ctr,aes192-ctr,aes256-ctr</encryption>
  </algorithms>
</ssh>

<aaa>
  <ssh-server-key-dir>${NCS_DIR}/etc/ncs/ssh</ssh-server-key-dir>

  <!-- Depending on OS - and also depending on user requirements -->
  <!-- the pam service value value must be tuned. -->

  <pam>
    <enabled>true</enabled>
    <service>common-auth</service>
  </pam>
  <external-authentication>
    <enabled>false</enabled>
    <executable>$WAE_ROOT/lib/exec/wae-ldap-auth</executable>
  </external-authentication>

  <local-authentication>
    <enabled>true</enabled>
  </local-authentication>

</aaa>

<!-- Hash algorithm used when setting leafs of type ianach:crypt-hash, -->
<!-- e.g. /aaa/authentication/users/user/password -->
<crypt-hash>
  <algorithm>sha-512</algorithm>
</crypt-hash>

<!-- Disable this for performance critical applications, enabling -->
<!-- rollbacks means additional disk IO for each transaction -->
<rollback>
  <enabled>true</enabled>
  <directory>./logs</directory>
  <history-size>50</history-size>
</rollback>


<cli>
  <enabled>true</enabled>

  <!-- Use the builtin SSH server -->
  <ssh>
    <enabled>true</enabled>
    <ip>0.0.0.0</ip>
    <port>2024</port>
  </ssh>

  <prompt1>\u@wae> </prompt1>
  <prompt2>\u@wae% </prompt2>

  <c-prompt1>\u@wae# </c-prompt1>
  <c-prompt2>\u@wae(\m)# </c-prompt2>

  <show-log-directory>./logs</show-log-directory>
  <show-commit-progress>true</show-commit-progress>
  <suppress-commit-message-context>maapi</suppress-commit-message-context>
```

```
    <suppress-commit-message-context>system</suppress-commit-message-context>
  </cli>

  <webui>
    <absolute-timeout>P1Y</absolute-timeout>
    <custom-headers>
      <header>
        <name>X-Content-Type-Options</name>
        <value>nosniff</value>
      </header>
      <header>
        <name>Content-Security-Policy</name>
        <value>default-src 'self'; script-src 'self'; img-src 'self' data:;
block-all-mixed-content; base-uri 'self'; frame-ancestors 'none'; style-src 'self'
'unsafe-inline'</value>
      </header>
    </custom-headers>
    <idle-timeout>PT30M</idle-timeout>
    <allow-symlinks>true</allow-symlinks>
    <enabled>true</enabled>
    <transport>
      <tcp>
        <enabled>true</enabled>
        <ip>0.0.0.0</ip>
        <port>8080</port>
        <redirect>https://@HOST@:8443</redirect>
        <!-- Uncomment this to enable support for IPv6&#xa;        <extra-listen>&#xa;
         <ip>::</ip>&#xa;          <port>8080</port>&#xa;        </extra-listen>&#xa;
      -->
      </tcp>
      <ssl>
        <enabled>true</enabled>
        <ip>0.0.0.0</ip>
        <port>8443</port>
        <key-file>${NCS_DIR}/var/ncs/webui/cert/host.key</key-file>
        <cert-file>${NCS_DIR}/var/ncs/webui/cert/host.cert</cert-file>
        <!-- Uncomment this to enable support for IPv6&#xa;        <extra-listen>&#xa;
         <ip>::</ip>&#xa;          <port>8443</port>&#xa;        </extra-listen>&#xa;
      -->
      </ssl>
    </transport>

    <cgi>
      <enabled>true</enabled>
      <php>
        <enabled>false</enabled>
      </php>
    </cgi>
  </webui>

  <rest>
    <enabled>true</enabled>
    <enable-legacy>true</enable-legacy>
  </rest>

  <restconf>
    <enabled>true</enabled>
  </restconf>

  <netconf-north-bound>
    <enabled>true</enabled>

    <transport>
      <ssh>
```

```
        <enabled>true</enabled>
        <ip>0.0.0.0</ip>
        <port>2022</port>
        <!-- Uncomment this to enable support for IPv6&#xa;        <extra-listen>&#xa;
         <ip>::</ip>&#xa;          <port>2022</port>&#xa;        </extra-listen>&#xa;
      -->
      </ssh>
      <tcp>
        <enabled>false</enabled>
        <ip>127.0.0.1</ip>
        <port>2023</port>
      </tcp>
    </transport>
  </netconf-north-bound>

  <netconf-call-home>
    <enabled>false</enabled>

    <transport>
      <tcp>
        <ip>0.0.0.0</ip>
        <port>4334</port>
      </tcp>
    </transport>
  </netconf-call-home>

  <!-- <ha> -->
  <!--   <enabled>true</enabled> -->
  <!-- </ha> -->

  <large-scale>
    <lsa>
      <!-- Enable Layered Service Architecture, LSA. This requires&#xa;           a
separate Cisco Smart License.&#xa;      -->
      <enabled>true</enabled>
    </lsa>
  </large-scale>

</ncs-config>
```

多くの構成パラメータのデフォルト値は、YANG ファイルで定義されています。wae.conf 構成パラメータ を参照してください。

ハイ アベイラビリティの設定

Cisco WAE は、高可用性（HA）と自動フェールオーバーをサポートしています。WAE ノードの 2 つのインスタンスは、並行して実行するように設定されています。プライマリノードはプライマリモードで設定され、セカンダリノードはスタンバイモードで設定されます。プライマリノードは、ポート 4570（または wae.conf ファイルで設定されたポート）でセカンダリノードからの接続をリッスンします。コミットされた CDB データは、定期的にセカンダリノードにミラーリングされます。スタンバイモードのノードで実行される CDB への書き込み操作（NIMO 操作、エージェントプロセス、またはスケジューラアクションなど）は失敗することに注意してください。

プライマリノードに障害が発生すると、セカンダリノードがプライマリノードとして引き継ぎます。セカンダリノードでプライマリモードが有効になると、書き込み操作が許可され、CDB

が再構築され、スケジュールされたジョブが実行されます。セカンダリノードでプライマリノードが以前に実行した操作が再開されます。

ステップ 1 プライマリノードとセカンダリノードの両方で、wae.conf ファイルを編集して HA を有効にします。

```
<ha>
  <enabled>true</enabled>
  <ip>0.0.0.0</ip>
  <!-- The following port configuration is optional.
       Default port is 4570. This option can be used
       to override the default port -->
  <!-- <port>4570</port> -->
</ha>
```

（注） /etc/hosts ファイルがホスト名から IP アドレスへのマッピングで更新されていることを確認してください。

ステップ 2 スーパーバイザを使用して両方のノードで Cisco WAE を再起動する

```
sudo supervisorctl restart wae:*
```

ステップ 3 両方のノードで、次のいずれかを実行します。

- Cisco WAE CLI から：

```
# wae ha-config nodes n1-name <hostname1>
# wae ha-config nodes n1-address <server-ip1>
# wae ha-config nodes n1-wae-uname <user1>
# wae ha-config nodes n2-name <hostname2>
# wae ha-config nodes n2-address <server-ip2>
# wae ha-config nodes n2-wae-uname <user2>
# wae ha-config cluster-id <cluster-id>
# wae ha-config temp-dir-location <temp-location>
```

プライマリノードで be-primary を開始し、ステータスを確認します。

```
# wae ha-config be-primary
# wae ha-config status
```

セカンダリノードで be-secondary を開始し、ステータスを確認します。

```
# wae ha-config be-secondary
# wae ha-config status
```

（注） temp-dir-location は、アーカイブプランファイルがプライマリからセカンダリの場所にコピーされる場所へのパスです。そのため、複製中にすべてのアーカイブファイルを保持するのに十分なスペースがあるディレクトリへのパスを追加することをお勧めします。

- WAE UI から：

1. [HA設定（HA configuration）] アイコンをクリックします。

   （注） /etc/hosts ファイルがホスト名から IP アドレスへのマッピングで更新されていることを確認してください

   。

2. N1 ノードと N2 ノードの詳細を入力します。

（注）　両方のノードのノード名に完全修飾ドメイン名を指定します。

3. [クラスタID（Cluster-ID）] を入力します。
4. [プライマリ指定（be-primary）]、[セカンダリ指定（be-secondary）]、または[指定なし（be-none）] を選択します。
5. [セカンダリ（Secondary）] をクリックします。

（注）
- ノードがセカンダリノードとして選択されると、[WAE UI] → [HA設定（HA Configuration）] および [WAE UI] → [ステータス（Status）] ページのみがそのノードに対して有効になります。
- ノードを [指定なし（be-none）] から [セカンダリ指定（be-secondary）] に移動するときは、ノードで収集が実行されていないこと、およびエージェント/スケジューラタスクが設定されていないことを確認してください。

ステップ 4 ノードのステータスは、[HA設定（HA Configuration）] ページに表示されます。

（注）　両方のノードで HA を設定した後は、プライマリノードでのみ設定を変更できます。

ステップ 5 データ同期のために、両方のノード間でパスワードなしの ssh を設定します。

SSH 認証キーの生成

```
# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user/.ssh/id_rsa.
Your public key has been saved in /home/user/.ssh/id_rsa.pub.
The key fingerprint is:
1x:x2:x4:22:5a:7x:2x:a5:a5:4x:6x:88:2c:33:x8:77 root@remote-host
```

公開キーをリモートホストにコピーする

```
# ssh-copy-id -i ~/.ssh/id_rsa.pub user@remote-host
user@remote-hosts's password:
```

または、サーバーに openssh-clients（ssh-copy-id コマンドユーティリティを提供するパッケージ）がインストールされていない場合は、次のコマンドで認証キーをコピーできます。

```
# cat ~/.ssh/id_rsa.pub | ssh user@remote-host "cat >> ~/.ssh/authorized_keys"
```

（注）　フェイルオーバーの場合、データ同期は反対方向に行われるため、両方のノードにパスワードなしの ssh を設定する必要があります。

ステップ 6 グローバルスケジューラを使用してデータ複製をスケジュールする： `<wae-run-directory>/networks`、`<wae-run-directory>/agents` および任意のアーカイブの下で、プライマリノードでの HA データ同期を使用してファイルを複製します。

データ複製には `rsync` ユーティリティが必要です。

システムで `rsync` ユーティリティが使用できない場合は、次のコマンドを使用してインストールします。

```
sudo yum install rsync
```

（注）
- プライマリノードがダウンすると、セカンダリノードがプライマリとして起動します。プライマリノードは、復元されると [なし（None）] として起動します。このノードを、セカンダリとして再度手動で設定する必要があります。
- フェイルオーバー後に完全なネットワークモデルを取得するには、XTC ベースの NIMO で収集の実行をスケジュールする必要があります。収集の実行がスケジュールされていない場合、収集が手動で実行されるまで、XTC ベースのリアクティブな変更は機能しません。
- ノードが [なし（None）] 状態から [セカンダリ（Secondary）] 状態に移行するときは、そのノードで収集またはエージェントが実行されていないことを確認してください。
- Netflow ワークフローと layout-nimo は、HA ではサポートされていません。

ハイ アベイラビリティのトラブルシューティング

HA 構成のトラブルシューティング時に、次の 2 つのログを確認する必要があります。

- <*wae_run_directory*>/logs/wae-java-vm.log
- <*wae_run_directory*>/logs/devel.log

次の表に、HA エラーとその意味を示します。

エラー コード	ENUM	説明
25	CONFD_ERR_HA_CONNECT	リモート HA ノードへの接続に失敗しました。
26	CONFD_ERR_HA_CLOSED	リモート HA ノードが WAE への接続を閉じたか、confd_ha_beseecondary() コール中にプライマリからの同期応答の待機にタイムアウトがありました。
27	CONFD_ERR_HA_BADFXS	リモート HA ノードには、WAE とは異なる FXS ポートのセットまたは異なるバージョンの FXS ポートがあります。
28	CONFD_ERR_HA_BADTOKEN	リモート HA ノードに WAE とは異なるトークンがあります。
29	CONFD_ERR_HA_BADNAME	リモート HA ノードの名前が、WAE でキャプチャされた名前とは異なります。

エラー コード	ENUM	説明
30	CONFD_ERR_HA_BIND	着信 HA 接続用の HA ソケットのバインドに失敗しました。
31	CONFD_ERR_HA_NOTICK	リモート HA ノードがインターバル ライブティックの生成に失敗しました。

LDAP の設定

Cisco WAE は、Lightweight Directory Access Protocol（LDAP）を使用して外部ユーザーの認証をサポートしています。

WAE で LDAP を設定する前に、次のことを確認してください。

- LDAP ディレクトリツリーとその内容に精通している必要があります。
- LDAP サーバーと収集の詳細をインストールして設定します。
- LDAPS プロトコルを使用するには、SSL 証明書を取得してキーストアに追加します。

  SSL 証明書を取得してインポートするコマンド

  次のコマンドを使用して、自己署名証明書を cert.pem ファイルに保存します。

  ```
  # openssl s_client -connect <ldap-host>:<ldap-ssl-port> </dev/null 2>/dev/null | sed
   -n '/^-----BEGIN/,/^-----END/ { p }' > cert.pem
  ```

  次のコマンドを使用して、デフォルトのキーストアパスを取得します。通常、デフォルトのキーストアパスは /etc/pki/java/cacerts for CentOS 7 with open-jdk です。

  ```
  # $WAE_ROOT/lib/exec/test-java-ssl-conn <ldap-host> <ldap-ssl-port> 2>1 | grep
  "trustStore is:"
  ```

  次のコマンドを使用して、証明書をデフォルトのキーストアにインポートします

  ```
  # sudo keytool -import -keystore <default-key-store-path> -storepass changeit -noprompt
   -file cert.pem
  ```

LDAP 設定のトラブルシューティングを行うには、次のログを表示します。

- LDAP 設定ログ：<*wae_run_directory*>/logs/wae-javavm.log
- LDAP 認証ランタイム ログ：<wae_run_directory> /logs/wae-ldap-auth.log

CLI を使用した LDAP の設定

始める前に

LDAP の設定 （13 ページ） の説明に従って、前提条件を満たしていることを確認します。

ステップ 1 wae.conf ファイルを編集して、外部認証を有効にします。

```
<<external-authentication>
  <enabled>true</enabled>
  <executable>$WAE_ROOT/lib/exec/wae-ldap-auth.sh</executable>
</external-authentication>
```

ステップ 2 WAE を再起動します。

```
# wae --start
```

ステップ 3 WAE CLI を使用して LDAP サーバーの詳細を設定します。

例：LDAP 設定

```
# wae_cli -u admin
# conf

(config)# wae ldap-config enabled
(config)# wae ldap-config protocol ldap
(config)# wae ldap-config server 10.220.121.47
(config)# wae ldap-config port 389
(config)# wae ldap-config search-base ou=people,dc=planetexpress,dc=com
(config)# wae ldap-config principal-expression "(uid={0})"
(config)# commit
```

例：SSL および admin ユーザーを使用した LDAP 設定

```
# wae_cli -u admin
# conf

(config)# devices authgroups group ldap-search default-map
(config)# devices authgroups group ldap-search default-map remote-name cn=admin,dc=company,dc=com
(config)# devices authgroups group ldap-search default-map remote-password HelloDolly
(config)# commit

(config)# wae ldap-config enabled
(config)# wae ldap-config protocol ldaps
(config)# wae ldap-config server 10.222.121.48
(config)# wae ldap-config port 636
(config)# wae ldap-config search-base ou=people,dc=company,dc=com
(config)# wae ldap-config principal-expression "(uid={0})"
(config)# wae ldap-config ldap-auth-group ldap-search
(config)# wae ldap-config keystore-path /home/centos/apps/wae712/wae/etc/wae-ldap-keystore
(config)# wae ldap-config keystore-pass wae-ldap-ks#
(config)# commit
(config)# exit
```

例：MS Active Directory サーバーの LDAP 設定

```
# wae_cli -u admin
# conf

(config)# devices authgroups group  ad-user ldap-search default-map
(config)# devices authgroups group  ad-user ldap-search default-map remote-name
CN=waeuser1,CN=Users,DC=woadtest,DC=local
(config)# devices authgroups group  ad-user ldap-search default-map remote-password HelloWAE
(config)# commit

(config)# wae ldap-config enabled
(config)# wae ldap-config protocol ldap
(config)# wae ldap-config server waelab.cisco.com
```

```
(config)# wae ldap-config port 389
(config)# wae ldap-config search-base cn=users,dc=woadtest,dc=local
(config)# wae ldap-config principal-expression "(sAMAccountName={0})"
(config)# wae ldap-config ldap-auth-group ad-user
(
(config)# commit
(config)# exit
```

WAE UI を使用した LDAP の設定

始める前に

LDAP の設定 （13 ページ） の説明に従って、前提条件を満たしていることを確認します。

ステップ 1 wae.conf ファイルを編集して、外部認証を有効にします。

```
<<external-authentication>
  <enabled>true</enabled>
  <executable>$WAE_ROOT/lib/exec/wae-ldap-auth.sh</executable>
</external-authentication>
```

ステップ 2 WAE を再起動します。

ステップ 3 WAE UI から、LDAP 設定アイコン（  ）をクリックします。

ステップ 4 デフォルトでは、有効なトグルスイッチはオンになっています。オンでない場合は、ユーザー認証にLDAPサーバーの使用を有効にして、スイッチをオンに切り替えます。

ステップ 5 LDAP オプションを入力します。詳細については、LDAP 設定オプション （15 ページ）を参照してください。

ステップ 6 [保存（Save）] をクリックします。

LDAP 設定オプション

表 *1 : LDAP* フィールドの説明

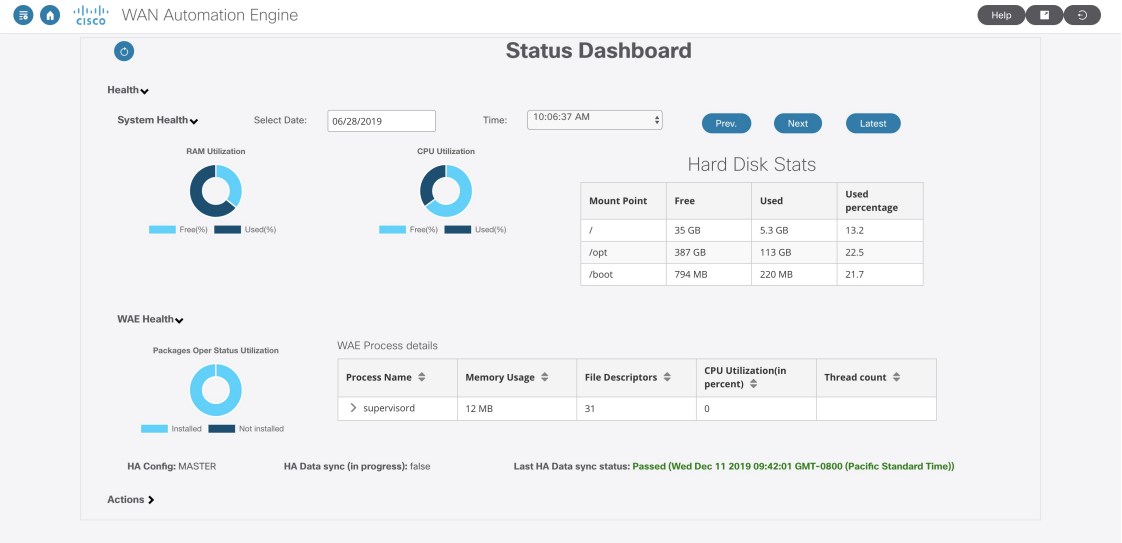
フィールド	説明
プロトコル	LDAP サーバーに到達するために使用するプロトコルです。 • LDAP：通信をクリアテキストで送信します。 • LDAPS：暗号化された安全な通信を送信します。 デフォルト値は LDAPS です。

フィールド	説明
サーバ <ldap-server>	LDAP サーバーの IP アドレスまたは FQDN。これはサーバーのホスト名の末尾に DNS ドメイン名を付加したものです。 FQDN 形式：<LDAP_hostname> <domain>.com
ポート	LDAP サーバーに到達するために使用するポートです。暗号化されていない認証の場合、デフォルトは TCP 389 です。暗号化されている認証の場合、デフォルトは TCP 636 です。 デフォルト値は 389 です。
検索ベース（Search Base） <ldap-base-ou>	WAE サーバーへのログイン許可を持っている必要があるすべてのユーザーアカウントの、基本検索 OU の識別名です。
主表現	デフォルト： `LDAP.Principal.Expr: (userPrincipalName={0})`, `{0}` トークンは、ログインページでユーザーが入力したユーザー名に置き換えられます。 `userPrincipalName=` は、LDAP 検索ベースでユーザーを識別するユーザーオブジェクトの LDAP 属性と一致する必要があります。 LDAP スキーマから、ユーザー固有の属性 `uid` を使用します。 WAE サーバーは、LDAP 検索ベースツリーの下のすべてのオブジェクトを検索します。 `uid=cisco-mate-user1` 一般的な選択肢には、`userPrincipalName` または `userName` などがあります。
LDAP 認証グループ（LDAP Auth Group）	LDAP 検索の実行に使用するユーザー名およびパスワード。 enable-password フィールドは LDAP には使用されません。UI での設定時に任意のダミー値を入力します。
キーストアパス（Keystore Path）	SSL が有効な場合のキーストアパス。
キーストアパスワード（Keystore Pass）	キーストアのパスワード。

ステータスダッシュボード

Cisco WAE が突然動作を停止したり、クラッシュしたりする場合があります。トラフィックポーラーが機能しなくなったり、アーカイブ中に問題が発生したりすることがあります。これらの問題は、WAE システムリソースが使い果たされていることが原因である場合があります。WAE のステータスダッシュボードは、システムリークの原因となるプロセスや、リソースを使い切っているプロセスを特定することにより、このような状況の対処に役立ちます。

ステータスダッシュボードにアクセスするには、Cisco WAE UI に移動し、[ステータスダッシュボード（Status Dashboard）] をクリックします。



ステータスダッシュボードは、主に 2 つのセクションに分かれています。

- ヘルス（Health）
- アクション（Actions）

[ヘルス（Health）] は、さらに [システムヘルス（System Health）] と [WAEヘルス（WAE Health）] に分けられます。

[システムヘルス（System Health）] は、RAM 使用率、CPU 使用率、およびシステムレベルのハードディスク統計をキャプチャします。ツールは 10 分ごとに実行され、統計が生成されます。[日付の選択（Select Date）] および [時刻（Time）] フィールドを使用して、必要なレポートにアクセスします。

デフォルトでは、最新のレポートが表示されます。レポート間を移動するには、[前へ（prev）]、[次へ（next）] ボタンを使用します。

RAM 使用率と CPU 使用率のグラフには、使用済みスペースと空きスペースが表示されます。使用済みまたは空き領域にカーソルを合わせると、実際の使用率が表示されます。

[WAEヘルス（WAE Health）] は、メモリ使用量、ファイル記述子、CPU 使用率など、プロセスレベルの使用状況の詳細をキャプチャします。プロセスで問題が発生した場合は、日付と時刻のフィールドを使用して、関連するすべてのプロセスレベルの詳細にアクセスします。

Cisco WAE の展開後、一部のパッケージがアンインストールされたままになる場合があります。[パッケージ運用使用率ステータス（Packages Oper Status Utilization）] チャートは、インストール済みおよびアンインストール済みパッケージの割合を示します。インストール済みまたは未インストールの領域をクリックすると、インストール済みまたは未インストールのパッケージのリストが表示されます。

HA 設定が有効になっている場合、システムがプライマリかセカンダリかを示します。

[アクション（Actions）] セクションには、NIMO アクションとエージェントアクションのステータスが表示されます。

[NIMOアクション（NIMO Actions）] セクションで、NIMO カードをクリックして NIMO ステータスの詳細を取得します。

（注） [NIMOアクション（NIMO Actions）] は、過去のアクションデータではなく、現在のアクションのみを表示します。

[エージェントアクション（Agent Actions）] セクションには、特定のエージェントのアクションステータスが表示されます。

WAE CLI ロギングについて

WAE は豊富なロギング機能を備えています。WAE は `wae.conf` ファイルで指定されたディレクトリにログを記録します。最も有用なログファイルは次のとおりです。

- `wae.log`：WAE デーモンログ。syslog に構成できます。
- `wae_err.log.1`、`wae_err.log.idx`、`wae_err.log.siz`：WAE デーモンに問題がある場合、このログにはサポートのためのデバッグ情報が含まれています。コマンド **wae --printlog wae_err.log** で内容を表示します。
- `audit.log`：すべてのノースバウンドインターフェイスをカバーする中央監査ログ。syslog に構成できます。
- `localhost:8080.access`：デーモンへのすべての http リクエスト。組み込み Web サーバーのアクセスログです。このファイルは、Apache などで定義されている Common Log Format に準拠しています。このログはデフォルトで無効であり、ローテーションされません。そのため、**logrotate(8)** を使用してください。
- `devel.log`：ユーザー作成コードをトラブルシューティングするためのデバッグログ。このログはデフォルトで有効であり、ローテーションされません。そのため、**logrotate(8)** を使用してください。このログは、`java-vm` または `python-vm` ログとともに使用してください。ユーザーコードは `vm` ログに記録され、対応するライブラリは `devel.log` に

記録されます。実稼働システムではこのログを無効にしてください。syslog に構成できます。

- `wae-java-vm.log`、`wae-python-vm.log`：サービスアプリケーションなど、Java または Python VM で実行されるコードのログ。Java および Python コードを作成する開発者は、このログを（`devel.log` と組み合わせて）デバッグに使用します。
- `netconf.log`、`snmp.log`：ノースバウンドエージェントのログ。syslog に構成できます。
- `rollback`*NNNNN*：すべての WAE コミットは、対応するロールバックファイルを生成します。`wae.conf` で、ロールバックファイルの最大数とファイル番号を構成できます。
- `xpath.trace`：XPATH は、XML テンプレートなど多くの場所で使用されます。このログファイルには、すべての XPATH 式の評価が示されます。テンプレートの XPATH をデバッグするには、代わりに CLI で pipe-target debug を使用します。
- `ned-cisco-ios-xr-pe1.trace`：デバイストレースがオンになっている場合、デバイスごとにトレースファイルが作成されます。ファイルの場所は `wae.conf` では構成されませんが、CLI などでデバイストレースがオンになっているときに構成されます。

Syslog

WAE では、BSD または IETF syslog フォーマット（RFC5424）を使用して、ローカルまたはリモートの syslog サーバーに syslog を送信できます。`wae.conf` ファイルを使用して、syslog に保存するログ（`ncs.log`、`devel.log`、`netconf.log`、または `snmp.log`）を選択できます。

次の例は、一般的な syslog 構成を示しています。

```
<syslog-config>
  <facility>daemon</facility>

  <udp>
     <enabled>false</enabled>
     <host>127.0.0.1</host>
     <port>895</port>
  </udp>

  <syslog-servers>
    <server>
      <host>127.0.0.2</host>
      <version>1</version>
    </server>
    <server>
      <host>127.0.0.3</host>
      <port>7900</port>
      <facility>local4</facility>
      </server>
   </syslog-servers>
  </syslog-config>

<ncs-log>
  <enabled>true</enabled>
```

```
  <file>
   <name>./logs/ncs.log</name>
   <enabled>true</enabled>
  </file>
  <syslog>
    <enabled>true</enabled>
  </syslog>
</ncs-log>
```

Syslog のメッセージと形式

次の表に、WAE syslog メッセージとそのフォーマットを示します。

記号	フォーマット文字列	備考
DAEMON_DIED	"Daemon ~s died"	外部データベースデーモンがその制御ソケットを閉じました。
DAEMON_TIMEOUT	"Daemon ~s timed out"	外部データベースデーモンがクエリに応答しませんでした。
NO_CALLPOINT	"no registration found for callpoint ~s of type=~s"	ConfD は XML ツリーにデータを入力しようとしましたが、関連するコールポイントにコードが登録されていませんでした。
CDB_DB_LOST	"CDB: lost DB, deleting old config"	CDB はデータスキーマファイルを検出しましたが、データファイルは検出しませんでした。空のデータベースから開始して CDB をリカバリしました。
CDB_CONFIG_LOST	"CDB: lost config, deleting DB"	CDB はデータファイルを検出しましたが、スキーマファイルは検出しませんでした。空のデータベースから開始して CDB をリカバリしました。
CDB_UPGRADE_FAILED	"CDB: Upgrade failed: ~s"	自動 CDB アップグレードに失敗しました。つまり、サポートされていない方法でデータモデルが変更されました。
CDB_INIT_LOAD	"CDB load: processing file: ~s"	CDB は初期化ファイルを処理しています。
CDB_OP_INIT	"CDB: Operational DB re-initialized"	アップグレードまたは破損したファイルが原因で、運用データベースが削除され、再初期化されました。
CDB_CLIENT_TIMEOUT	"CDB client (~s) timed out, waiting for ~s"	CDB クライアントがタイムアウト時間内に応答できず、切断されました。

記号	フォーマット文字列	備考
INTERNAL_ERROR	"Internal error: ~s"	ConfD 内部エラーが発生しました。シスコテクニカルサポートに報告する必要があります。
AAA_LOAD_FAIL	"Failed to load AAA: ~s"	外部データベースの動作に問題があるか、AAA のマウントまたは入力が不適切なため、AAA データをロードできませんでした。
EXTAUTH_BAD_RET	"External auth program (user=~s) ret bad output: ~s"	認証は外部であり、外部プログラムが不適切な形式のデータを返しました。
BRIDGE_DIED	"confd_aaa_bridge died - ~s"	ConfD が confd_aaa_bridge を開始するように構成され、C プログラムが停止しました。
PHASE0_STARTED	"ConfD phase0 started"	ConfD は開始フェーズ 0 を開始しました。
PHASE1_STARTED	"ConfD phase1 started"	ConfD は開始フェーズ 1 を開始しました。
STARTED	"ConfD started vsn: ~s"	ConfD が開始しました。
UPGRADE_INIT_STARTED	"Upgrade init started"	インサービスアップグレードの初期化が開始しました。
UPGRADE_INIT_SUCCEEDED	"Upgrade init succeeded"	インサービスアップグレードの初期化に成功しました。
UPGRADE_PERFORMED	"Upgrade performed"	インサービスアップグレードが実行されましたが、まだコミットされていません。
UPGRADE_COMMITTED	"Upgrade committed"	インサービスアップグレードがコミットされました。
UPGRADE_ABORTED	"Upgrade aborted"	インサービスアップグレードが中止されました。
CONSULT_FILE	"Consulting daemon configuration file ~s"	ConfD は構成ファイルを読み取っています。
STOPPING	"ConfD stopping (~s)"	ConfD が停止しています（たとえば、**confd --stop** のため）。

記号	フォーマット文字列	備考
RELOAD	"Reloading daemon configuration"	デーモン構成のリロードを開始しました。
BADCONFIG	"Bad configuration: ~s:~s: ~s"	confd.conf に不正なデータが含まれています。
WRITE_STATE_FILE_FAILED	"Writing state file failed: ~s: ~s (~s)"	状態ファイルの書き込みに失敗しました。
READ_STATE_FILE_FAILED	"Reading state file failed: ~s: ~s (~s)"	状態ファイルの読み取りに失敗しました。
SSH_SUBSYS_ERR	"ssh protocol subsys - ~s"	クライアントは **\"subsystem\"** コマンドを正しく送信しませんでした。
SESSION_LIMIT	"Session limit of type '~s' reached, rejected new session request"	セッション制限に達しました。新しいセッションリクエストは拒否されました。
CONFIG_TRANSACTION_LIMIT	"Configuration transaction limit of type '~s' reached, rejected new transaction request"	構成トランザクション制限に達しました。新しいトランザクションリクエストは拒否されました。
ABORT_CAND_COMMIT	"Aborting candidate commit, request from user, reverting configuration"	ユーザーのリクエストにより、候補コミットを中止しています。構成を元に戻しています。
ABORT_CAND_COMMIT_TIMER	"Candidate commit timer expired, reverting configuration"	候補コミットタイマーが期限切れになりました。構成を元に戻しています。
ABORT_CAND_COMMIT_TERM	"Candidate commit session terminated, reverting configuration"	候補コミットセッションが終了しました。構成を元に戻しています。
ROLLBACK_REMOVE	"Found half created rollback0 file - removing and creating new"	半分しか作成されていないことがわかった rollback0 ファイルを削除して再作成しています。
ROLLBACK_REPAIR	"Found half created rollback0 file - repairing"	半分しか作成されていないことがわかった rollback0 ファイルを修復しています。
ROLLBACK_FAIL_REPAIR	"Failed to repair rollback files"	ロールバックファイルの修復に失敗しました。
ROLLBACK_FAIL_CREATE	"Error while creating rollback file: ~s: ~s"	ロールバックファイルの作成中にエラーが発生しました。

記号	フォーマット文字列	備考
ROLLBACK_FAIL_RENAME	"Failed to rename rollback file ~s to ~s: ~s"	ロールバックファイルの名前を変更できませんでした。
NS_LOAD_ERR	"Failed to process namespace ~s: ~s"	システムは、ロードされた名前空間を処理できませんでした。
NS_LOAD_ERR2	"Failed to process namespaces: ~s"	システムは、ロードされた名前空間を処理できませんでした。
FILE_LOAD_ERR	"Failed to load file ~s: ~s"	システムは、そのロードパスにファイルをロードできませんでした。
FILE_LOADING	"Loading file ~s"	システムは、ファイルのロードを開始しています。
SKIP_FILE_LOADING	"Skipping file ~s: ~s"	システムは、ファイルをスキップしました。
FILE_LOAD	"Loaded file ~s"	システムは、ファイルをロードしました。
LISTENER_INFO	"~s to listen for ~s on ~s:~s"	ConfD は、着信接続をリッスンするために開始または停止します。
NETCONF_HDR_ERR	"Got bad NETCONF TCP header"	ユーザーとグループが正しくフォーマットされていないことを示すクリアテキストのヘッダー。
LIB_BAD_VSN	"Got library connect from wrong version (~s, expected ~s)"	ConfD に接続しているアプリケーションで、ConfD バージョンと一致しないライブラリバージョン（たとえば、古いバージョンのクライアントライブラリ）が使用されました。
LIB_BAD_SIZES	"Got connect from library with insufficient keypath depth/keys support (~s/ ~s, needs ~s/~s)"	ConfD に接続しているアプリケーションで、データモデルで使用されるキーの深さと数を処理できないライブラリバージョンが使用されました。
LIB_NO_ACCESS	"Got library connect with failed access check: ~s"	アプリケーションが ConfD に接続したときにアクセスチェックエラーが発生しました。
SNMP_NOT_A_TRAP	"SNMP gateway: Non-trap received from ~s"	トラップ受信ポートで UDP パッケージを受信しましたが、SNMP トラップではありません。

記号	フォーマット文字列	備考
SNMP_TRAP_V1	"SNMP gateway: V1 trap received from ~s"	トラップ受信ポートで SNMPv1 トラップを受信しましたが、v1 トラップの転送はサポートされていません。
SNMP_TRAP_NOT_FORWARDED	"SNMP gateway: Can't forward trap from ~s; ~s"	SNMP トラップが転送されませんでした。
SNMP_TRAP_UNKNOWN_ SENDER	"SNMP gateway: Not forwarding trap from ~s; the sender is not recognized"	SNMP トラップが転送されるはずでしたが、送信者が confd.conf にリストされていませんでした。
SNMP_TRAP_OPEN_PORT	"SNMP gateway: Can't open trap listening port ~s: ~s"	SNMP トラップをリッスンするためのポートを開けませんでした。
SNMP_TRAP_NOT_RECOGNIZED	"SNMP gateway: Can't forward trap with OID ~s from ~s; There is no notification with this OID in the loaded models"	トラップ受信ポートで SNMP トラップを受信しましたが、その定義が不明です。
XPATH_EVAL_ERROR1	"XPath evaluation error: ~s for ~s"	xpath 式の評価中にエラーが発生しました。
XPATH_EVAL_ERROR2	"XPath evaluation error: '~s' resulted in ~s for ~s"	xpath 式の評価中にエラーが発生しました。
CANDIDATE_BAD_FILE_FORMAT	"Bad format found in candidate db file ~s; resetting candidate"	候補データベースファイルのフォーマットが正しくありません。候補データベースが空のデータベースにリセットされます。
CANDIDATE_CORRUPT_FILE	"Corrupt candidate db file ~s; resetting candidate"	候補データベースファイルが壊れているため、読み取ることができません。候補データベースが空のデータベースにリセットされます。
MISSING_DES3CBC_SETTINGS	"DES3CBC keys were not found in confd.conf"	confd.conf に DES3CBC キーが見つかりませんでした。
MISSING_AESCFB128_ SETTINGS	"AESCFB128 keys were not found in confd.conf"	confd.conf に AESCFB128 キーが見つかりませんでした。
SNMP_MIB_LOADING	"Loading MIB: ~s"	SNMP エージェントが MIB ファイルをロードしています。
SNMP_CANT_LOAD_MIB	"Can't load MIB file: ~s"	SNMP エージェントが MIB ファイルのロードに失敗しました。

記号	フォーマット文字列	備考
SNMP_WRITE_STATE_ FILE_FAILED	"Write state file failed: ~s: ~s"	SNMP エージェント状態ファイルの書き込みに失敗しました。
SNMP_READ_STATE_ FILE_FAILED	"Read state file failed: ~s: ~s"	SNMP エージェント状態ファイルの読み取りに失敗しました。
SNMP_REQUIRES_CDB	"Can't start SNMP. CDB is not enabled"	SNMP エージェントを開始する前に、CDB を有効にする必要があります。
FXS_MISMATCH	"Fxs mismatch, slave is not allowed"	セカンダリは、異なる fxs ファイルを持つプライマリに接続されています。
TOKEN_MISMATCH	"Token mismatch, slave is not allowed"	セカンダリは、不正な認証トークンでプライマリに接続されています。
HA_SLAVE_KILLED	"Slave ~s killed due to no ticks"	セカンダリノードは、ティックを生成しませんでした。
HA_DUPLICATE_NODEID	"Nodeid ~s already exists"	すでに存在するノード ID を持つセカンダリが到着しました。
HA_FAILED_CONNECT	"Failed to connect to master: ~s"	セカンダリがプライマリに接続できなかったため、ライブラリをセカンダリにする呼び出しの試行に失敗しました。
HA_BAD_VSN	"Incompatible HA version (~s, expected ~s), slave is not allowed"	セカンダリは、互換性のない HA プロトコルバージョンでプライマリに接続されています。
NETCONF	"~s"	NETCONF トラフィックログメッセージ。
DEVEL_WEBUI	"~s"	デベロッパー Web UI ログメッセージ。
DEVEL_AAA	"~s"	デベロッパー AAA ログメッセージ。
DEVEL_CAPI	"~s"	デベロッパー C API ログメッセージ。
DEVEL_CDB	"~s"	デベロッパー CDB ログメッセージ。
DEVEL_CONFD	"~s"	デベロッパー ConfD ログメッセージ。
DEVEL_SNMPGW	"~s"	デベロッパー SNMP ゲートウェイログメッセージ。
DEVEL_SNMPA	"~s"	デベロッパー SNMP エージェントログメッセージ。

記号	フォーマット文字列	備考
NOTIFICATION_REPLAY_ STORE_FAILURE	"~_s"	組み込みの通知再生ストアで障害が発生しました。
EVENT_SOCKET_TIMEOUT	"Event notification subscriber with bitmask ~s timed out, waiting for ~s"	イベント通知サブスクライバは、構成されたタイムアウト期間内に応答しませんでした。
EVENT_SOCKET_WRITE_BLOCK	"~s"	イベントソケットへの書き込みが長時間ブロックされました。
COMMIT_UN_SYNCED_DEV	"Committed data towards device ~s which is out of sync"	同期状態が不良または不明なデバイスに対してデータがコミットされました。
NCS_SNMP_INIT_ERR	"Failed to locate snmp_init.xml in loadpath ~s"	ロードパスで snmp_init.xml が見つかりませんでした。
NCS_JAVA_VM_START	"Starting the NCS Java VM"	NCS Java VM を起動しています。
NCS_JAVA_VM_FAIL	"The NCS Java VM ~s"	NCS Java VM の障害またはタイムアウトが発生しました。
NCS_PACKAGE_SYNTAX_ERROR	"Failed to load NCS package: ~s; syntax error in package file"	パッケージファイルの構文エラー。
NCS_PACKAGE_DUPLICATE	"Failed to load duplicate NCS package ~s: (~s)"	重複するパッケージが見つかりました。
NCS_PACKAGE_COPYING	"Copying NCS package from ~s to ~s"	パッケージがロードパスからプライベートディレクトリにコピーされました。
NCS_PACKAGE_UPGRADE_ ABORTED	"NCS package upgrade failed with reason '~s'"	CDB のアップグレードが中止されました。これは、CDB が変更されていないことを意味します。ただし、パッケージの状態は変更されました。
NCS_PACKAGE_BAD_ NCS_VERSION	"Failed to load NCS package: ~s; requires NCS version ~s"	パッケージの NCS バージョンが正しくありません。
NCS_PACKAGE_BAD_ DEPENDENCY	"Failed to load NCS package: ~s; required package ~s of version ~s is not present (found ~s)"	NCS パッケージの依存関係が正しくありません。
NCS_PACKAGE_CIRCULAR_ DEPENDENCY	"Failed to load NCS package: ~s; circular dependency found"	NCS パッケージに循環依存関係があります。

記号	フォーマット文字列	備考
CLI_CMD	"CLI '~s'"	ユーザーが CLI コマンドを実行しました。
CLI_DENIED	"CLI denied '~s'"	権限が原因で、ユーザーは CLI コマンドの実行を拒否されました。
BAD_LOCAL_PASS	"Provided bad password"	ローカルに構成されたユーザーが間違ったパスワードを入力しました。
NO_SUCH_LOCAL_USER	"no such local user"	存在しないローカルユーザーがログインしようとしました。
PAM_LOGIN_FAILED	"pam phase ~s failed to login through PAM: ~s"	ユーザーが PAM 経由でログインできませんでした。
PAM_NO_LOGIN	"failed to login through PAM: ~s"	ユーザーが PAM 経由でログインできませんでした。
EXT_LOGIN	"Logged in over ~s using externalauth, member of groups: ~s~s"	外部認証されたユーザーがログインしました。
EXT_NO_LOGIN	"failed to login using externalauth: ~s"	ユーザーの外部認証に失敗しました。
GROUP_ASSIGN	"assigned to groups: ~s"	ユーザーは一連のグループに割り当てられました。
GROUP_NO_ASSIGN	"Not assigned to any groups - all access is denied"	ユーザーはログインしましたが、どのグループにも割り当てられていません。
MAAPI_LOGOUT	"Logged out from maapi ctx=~s (~s)"	管理エージェント API（MAAPI）ユーザーがログアウトされました。
SSH_LOGIN	"logged in over ssh from ~s with authmeth:~s"	ユーザーが ConfD の組み込み SSH サーバーにログインしました。
SSH_LOGOUT	"Logged out ssh <~s> user"	ユーザーが ConfD の組み込み SSH サーバーからログアウトされました。
SSH_NO_LOGIN	"Failed to login over ssh: ~s"	ユーザーが ConfD の組み込み SSH サーバーにログインできませんでした。
NOAAA_CLI_LOGIN	"logged in from the CLI with aaa disabled"	ユーザーが --noaaa フラグを confd_cli に使用しました。
WEB_LOGIN	"logged in through Web UI from ~s"	ユーザーが Web UI を介してログインしました。

記号	フォーマット文字列	備考
WEB_LOGOUT	"logged out from Web UI"	Web UI ユーザーがログアウトしました。
WEB_CMD	"WebUI cmd '~s'"	ユーザーが Web UI コマンドを実行しました。
WEB_ACTION	"WebUI action '~s'"	ユーザーが Web UI アクションを実行しました。
WEB_COMMIT	"WebUI commit ~s"	ユーザーが Web UI コミットを実行しました。
SNMP_AUTHENTICATION_FAIL	"ESDNMP authentication failed: ~s"	SNMP 認証に失敗しました。
LOGIN_REJECTED	"~s"	ユーザーの認証がアプリケーションのコールバックによって拒否されました。
COMMIT_INFO	"commit ~s"	構成変更に関する情報が実行中のデータストアにコミットされました。
CLI_CMD_DONE	"CLI done"	CLI コマンドが正常に完了しました。
CLI_CMD_ABORTED	"CLI aborted"	CLI コマンドが中止されました。
NCS_DEVICE_OUT_OF_SYNC	"NCS device-out-of-sync Device '~s' Info '~s'"	check-sync アクションで、デバイスの非同期が報告されました。
NCS_SERVICE_OUT_OF_SYNC	"NCS service-out-ofsync Service '~s' Info '~s'"	check-sync アクションで、サービスの非同期が報告されました。
NCS_PYTHON_VM_START	"Starting the NCS Python VM"	NCS Python VM を起動しています。
NCS_PYTHON_VM_FAIL	"The NCS Python VM ~s"	NCS Python VM が失敗したか、タイムアウトしました。
NCS_SET_PLATFORM_ DATA_ERRORS	"NCS Device '~s' failed to set platform data Info '~s'"	デバイスは、接続時にプラットフォーム運用データを設定できませんでした。
NCS_SMART_LICENSING_ START	"Starting the NCS Smart Licensing Java VM"	NCS スマートライセンス Java VM を起動しています。
NCS_SMART_LICENSING_FAIL	"The NCS Smart Licensing Java VM ~s"	NCS スマートライセンス Java VM が失敗したか、タイムアウトしました。
NCS_SMART_LICENSING_ GLOBAL_NOTIFICATION	"Smart Licensing Global Notification: ~s"	スマートライセンスのグローバル通知。

記号	フォーマット文字列	備考
NCS_SMART_LICENSING_ ENTITLEMENT_NOTIFICATION	"Smart Licensing Entitlement Notification: ~s"	スマートライセンス資格の通知。
NCS_SMART_LICENSING_EVALUATION_COUNTDOWN	"Smart Licensing evaluation time remaining: ~s"	スマートライセンス評価の残り時間。
DEVEL_SLS	"~s"	デベロッパー スマート ライセンス API ログメッセージ。
JSONRPC_REQUEST	"JSON-RPC: '~s' with JSON params ~s"	JSON-RPC メソッドがリクエストされました。
DEVEL_ECONFD	"~s"	デベロッパー econfd API ログメッセージ。
CDB_FATAL_ERROR	"fatal error in CDB: ~s"	CDB で回復不能なエラーが発生しました。
LOGGING_STARTED	"Daemon logging started"	ロギングサブシステムが開始されました。
LOGGING_SHUTDOWN	"Daemon logging terminating, reason: ~s"	ロギングサブシステムが終了しました。
REOPEN_LOGS	"Logging subsystem, reopening log files"	ロギングサブシステムがログファイルを再度開きました。
OPEN_LOGFILE	"Logging subsystem, opening log file '~s' for ~s"	特定タイプのロギングのターゲットファイルを示します。
LOGGING_STARTED_TO	"Writing ~s log to ~s"	サブシステムのログを特定のファイルに書き込みます。
LOGGING_DEST_CHANGED	"Changing destination of ~s log to ~s"	ターゲットログファイルが別のファイルに変更されます。
LOGGING_STATUS_CHANGED	"~s ~s log"	サブシステムのロギングステータス（有効/無効）の変更を通知します。
ERRLOG_SIZE_CHANGED	"Changing size of error log (~s) to ~s (was ~s)"	エラーログのログサイズの変更を通知します。
CGI_REQUEST	"CGI: '~s' script with method ~s"	CGI スクリプトがリクエストされました。
MMAP_SCHEMA_FAIL	"Failed to setup the shared memory schema"	共有メモリスキーマの設定に失敗しました。

記号	フォーマット文字列	備考
KICKER_MISSING_SCHEMA	"Failed to load kicker schema"	キッカースキーマのロードに失敗しました。
JSONRPC_REQUEST_IDLE_TIMEOUT	"Stopping session due to idle timeout: ~s"	JSON-RPC アイドルタイムアウト。
JSONRPC_REQUEST_ ABSOLUTE_TIMEOUT	"Stopping session due to absolute timeout: ~s"	JSON-RPC 絶対タイムアウト。

データベースのロック

このセクションでは、WAE に存在するさまざまなロックと、それらがどのように相互作用するかについて説明します。

グローバルロック

WAE 管理バックプレーンは、データストア（実行中）をロックし続けます。このロックはグローバルロックと呼ばれ、データストアへの排他的アクセスを許可するメカニズムを提供します。グローバルロックは、NETCONF <lock> 操作や Maapi.lock() 呼び出しなどノースバウンドエージェントを介して明示的に取得できる唯一のロックです。

グローバルロックは、データストア全体に対して行うことも、部分的なロックにする（データモデルのサブセットに対して行う）こともできます。部分ロックは、NETCONF および MAAPI を介して公開されます。

エージェントは、グローバルロックを要求して、排他的な書き込みアクセスを確保できます。エージェントがグローバルロックを保持している場合、他の誰もそのデータストアに書き込むことはできません。この動作は、トランザクションエンジンによって強制されます。他のロック所有者（部分ロックを含む）がなく、すべてのデータプロバイダーがロック要求を承認した場合に、実行中のグローバルロックがエージェントに付与されます。各データプロバイダー（CDB または外部データプロバイダー）には、ロックを拒否または受け入れるために呼び出される lock() コールバックがあります。**ncs --status** の出力には、ロックステータスが含まれます。

トランザクションロック

ノースバウンドエージェントは、WAE 管理バックプレーンに対するユーザーセッションを開始します。各ユーザーセッションは、複数のトランザクションを開始できます。トランザクションは、読み取り/書き込みまたは読み取り専用です。

トランザクションエンジンには、実行中のデータストアに対する内部ロックがあります。これらのトランザクションロックは、データストアに対する構成の更新をシリアル化するために存在し、グローバルロックとは別のものです。

ノースバウンドエージェントが実行中のデータストアを新しい構成で更新する場合、トランザクションロックを暗黙的に取得して解放します。トランザクションエンジンは、トランザクション ステート マシンを通過するときにロックを管理します。ノースバウンドエージェントにトランザクションロックを公開する API はありません。

トランザクションエンジンがトランザクションのロックを取得する場合（たとえば、検証状態に入るとき）、最初に他のトランザクションがロックを保持していないことを確認します。次に、そのデータストアにグローバルロックが設定されているユーザーセッションがないことを確認します。最後に、transLock() コールバックを使用して各データプロバイダーを呼び出します。

ノースバウンドエージェントとグローバルロック

暗黙的なトランザクションロックとは対照的に、一部のノースバウンドエージェントは、グローバルロックへの明示的なアクセスを公開します。管理 API は、Maapi.lock() および Maapi.unlock() メソッド（および部分ロック用の対応する Maapi.lockPartial() Maapi.unlockPartial()）を提供することにより、グローバルロックを公開します。ユーザーセッションが確立（または接続）されると、これらの関数を呼び出すことができます。

CLI では、次のように、さまざまな構成モードに入るときにグローバルロックが取得されます。

- **config exclusive**：実行中のデータストアのグローバルロックを取得します。
- **config terminal**：ロックを取得しません。

CLI は、構成モードが終了するまでグローバルロックを保持します。

エキスパート モードは、CLI と同じように動作し、前述の CLI モードに対応する [プライベート編集（Edit private）] および [排他編集（Edit exclusive）] と呼ばれる編集タブがあります。

NETCONF エージェントは、<lock> 操作を、リクエストされたデータストアのグローバルロックのリクエストに変換します。部分ロックも partial-lock rpc を通じて公開されます。

外部データ プロバイダーと CDB

外部データプロバイダーは、lock() および unlock() コールバックを実装する必要はありません。WAE は、グローバルロックが取得されている間、データプロバイダーへの transLock() 状態遷移の開始を試みません。データプロバイダーがロック用のコールバックを実装する理由は、他の誰かがデータプロバイダーのデータベースに書き込むことができる場合です。

CDB は、lock() コールバックと unlock() コールバックを無視します（データ プロバイダー インターフェイスが唯一の書き込みインターフェイスであるため）。

CDB には、データベースに独自の内部ロックがあります。実行中のデータストアには、1 つの書き込みロックと複数の読み取りロックがあります。データストアにアクティブな読み取りロックがある場合、データストアの書き込みロックを取得することはできません。CDB のロックは、リーダーが常にデータの一貫したビューを取得できるようにするために存在します

（YANG リストエントリの getNext() の呼び出しの間に別のユーザーが構成ノードを削除すると、混乱が生じます）。

トランザクション中、transLock() はトランザクションのデータストアに対して CDB 読み取りロックを取得しますが、writeStart() は読み取りロックを解放し、代わりに書き込みロックを取得しようとします。CDB 外部リーダークライアントは、Cdb.startSession() と Cdb.endSession() の間で暗黙的に CDB 読み取りロックを取得します。つまり、CDB クライアントが読み取りを行っている間、トランザクションは writeStart() を通過できません。逆に、トランザクションが writeStart() と commit() または abort() の間にある間は、CDB リーダーを開始できません。

CDB のオペレーショナルストアにはロックがありません。WAE のトランザクションエンジンは、そこからのみ読み取ることができます。CDB クライアントの書き込みは、書き込み操作単位でアトミックです。

ユーザーセッションへのロックの影響

セッションがロックされているデータストアを変更しようとすると、失敗します。たとえば、CLI は次のように出力します。

```
admin@wae(config)# commit
Aborted: the configuration database is locked
```

一部のロックは持続時間が短いため（CDB 読み取りロックなど）、WAE はデフォルトで、失敗した操作を構成可能な時間だけ再試行するように構成されています。この時間が経過してもデータストアがロックされたままの場合、操作は失敗します。

再試行タイムアウトを構成するには、`wae.conf` で `/ncs-config/commit-retry-timeout` 値を設定します。

セキュリティ

WAE には、特定のタスクを実行する権限が必要です。ターゲットシステムによっては、次のタスクにルート権限が必要になる場合があります。

- 特権ポートへのバインド。`wae.conf` 構成ファイルは、WAE が bind(2) する必要があるポート番号を指定します。ポート番号が 1024 より小さい場合、ターゲット オペレーティング システムで WAE が非ルートユーザーとしてこれらのポートにバインドすることを許可しない限り、通常、WAE はルート権限を必要とします。
- PAM を認証に使用する場合、`$NCS_DIR/lib/ncs/priv/pam/epam` としてインストールされたプログラムが PAM クライアントとして機能します。ローカルの PAM 構成によっては、このプログラムにルート権限が必要になる場合があります。PAM がローカルの `passwd` ファイルを読み取るように構成されている場合、プログラムはルートとして実行するか、setuid root である必要があります。ローカル PAM 構成で、WAE に pam_radius_auth などを実行するように指示している場合、ローカル PAM のインストールによっては、ルート権限が必要ない場合があります。

- CLI を使用して実行可能ファイルを実行する CLI コマンドを作成する場合は、`$NCS_DIR/lib/ncs/priv/ncs/cmdptywrapper` プログラムのアクセス許可を変更します。

ルートまたは特定のユーザーとして実行可能ファイルを実行するには、cmdptywrapper を setuid root にします。

```
# chown root cmdptywrapper
# chmod u+s cmdptywrapper
```

これに失敗すると、すべてのプログラムは WAE デーモンを実行しているユーザーとして実行されます。そのユーザーがルートの場合、上記の chmod 操作を実行する必要はありません。

これに失敗すると、すべてのプログラムは confd デーモンを実行しているユーザーとして実行されます。そのユーザーがルートの場合、上記の chmod 操作を実行する必要はありません。

アクションを介して実行される実行可能ファイルの場合、`$NCS_DIR/lib/ncs/priv/ncs/cmdwrapper` プログラムのアクセス許可を変更します。

```
# chown root cmdwrapper
# chmod u+s cmdwrapper
```

WAE は、クリアテキスト TCP を介して NETCONF を終了するように指示できます。これは、デバッグに役立ちます（NETCONF トラフィックをキャプチャして分析できます）。また、SSH 以外のローカル独自のトランスポートメカニズムを提供する場合にも役立ちます。クリアテキスト TCP による終了は認証されません。クリアテキストクライアントは、セッションを実行するユーザーを WAE に通知するだけです。認証は、SSH サーバーなどの外部エンティティによってすでに行われていることが前提です。クリアテキスト TCP が有効になっている場合、WAE はこれらの接続のために localhost（127.0.0.1）にバインドする必要があります。

クライアントライブラリは WAE に接続します。たとえば、CDB API は TCP ベースであり、CDB クライアントは WAE に接続します。WAE は、`wae.conf` パラメータ `/ncs-config/ncs-ipc-address/ip`（デフォルトのアドレスは 127.0.0.1）および `/ncs-config/ncs-ipcaddress/port`（デフォルトのポートは 4565）を介して、これらの接続に使用するアドレスを学習します。

WAE は、同じソケット上でさまざまな種類の接続を多重化します（IP とポートの組み合わせ）。次のプログラムはソケットに接続します。

- **ncs --reload** などのリモートコマンド。
- CDB クライアント。
- 外部データベース API クライアント。
- 管理エージェント API（MAAPI）クライアント。
- ncs_cli プログラム。

デフォルトでは、上記のプログラムは信頼できると見なされます。MAAPI クライアントと ncs_cli は、WAE に接続する前にユーザーを認証します。CDB クライアントと外部データベース API クライアントは信頼できると見なされるため、認証は必要ありません。

ncs-ipc-address ソケットはシステムへの完全な非認証アクセスを許可するため、信頼できないネットワークからソケットにアクセスできないようにすることが重要です。アクセスチェックを使用して、ncs-ipc-address ソケットへのアクセスを制限することもできます。IPC ポートへのアクセスの制限 （34 ページ） を参照してください。

IPC ポートへのアクセスの制限

デフォルトでは、IPC ポートに接続するクライアントは信頼できると見なされます。認証は必要ありません。リモートアクセスを防止するために、WAE は `/ncs-config/ncs-ipc-address/ip` に 127.0.0.1 を使用します。ただし、アクセスチェックを構成することで、IPC ポートへのアクセスを制限できます。

アクセスチェックを有効にするには、`wae.conf` 要素 `/ncs-config/ncs-ipc-accesscheck/enabled` を **true** に設定し、`/ncs-config/ncs-ipc-accesscheck/filename` にファイル名を指定します。ファイルには、共有秘密（ランダムな文字による文字列）が含まれている必要があります。IPC ポートに接続するクライアントは、WAE 機能へのアクセス権が付与される前に、チャレンジハンドシェイクを提供する必要があります。

（注） このファイルのアクセス許可は、IPC ポートへの接続が許可されている WAE デーモンおよびクライアントプロセスによってのみファイルが読み取られるように、OS ファイル権限によって制限する必要があります。たとえば、デーモンとクライアントの両方が root として実行されている場合、ファイルは root によって所有され、「所有者による読み取り」権限（モード 0400）のみを持つことができます。別の方法は、デーモンとクライアントのみが属するグループを作成し、ファイルのグループ ID をそのグループに設定し、「グループによる読み取り」（モード 040）権限のみを持つことです。

クライアントライブラリに秘密を提供し、アクセス チェック ハンドシェイクを使用するように指示するには、環境変数 NCS_IPC_ACCESS_FILE を、シークレットを含むファイルのフルパス名に設定します。上記のすべてのクライアントにはこれで十分です。このチェックを有効にするためにアプリケーションコードを変更する必要はありません。

（注） アクセスチェックは、デーモンとクライアントの両方に対して有効または無効にする必要があります。たとえば、`wae.conf` 要素 `/ncsconfig/ncs-ipc-access-check/enabled` が **true** に設定されていない場合に、秘密が含まれるファイルを環境変数 NCS_IPC_ACCESS_FILE で指してクライアントが起動される場合、クライアント接続は失敗します。

WAE 運用データのクリア

データベースから WAE 運用データを消去するには、それぞれの NIMO ネットワークモデルからモデル l1-model を削除する必要があります。その後、デバイスツリーを削除します。NIMO ネットワークモデルにレイアウトがある場合は、それらのレイアウトを NIMO ネットワークモデルから削除します。

次のコマンド例は、as64002 ネットワークモデルとデバイスツリーから運用データを消去する方法を示しています。

```
delete networks network as64002 model
delete networks network as64002 layouts
delete networks network as64002 l1-model
delete devices device *
commit
```

WAE 構成のバックアップと復元

YANG ランタイムフレームワークを使用すると、WAE 構成を簡単にバックアップおよび復元できます。収集を開始する前（つまり、運用データが読み込まれる前）に、WAE 構成をバックアップすることをお勧めします。

- WAE 構成をバックアップするには、次の手順を実行します。

  ```
  admin@wae% save /home/wae/wae-backup.cfg
  ```

  上記のコマンドは、構成データと運用データの両方をバックアップします。構成データのみをバックアップするには、WAE 運用データのクリア （35 ページ）の説明に従って、データベースから運用データを消去する必要があります。すべての運用データが削除されるため、実稼働環境で運用データを消去する前に注意してください。

- WAE 構成を復元するには、次の手順を実行します。

  ```
  [wae@wae ~]$ ncs_load -l -m -F j wae-backup.cfg
  ```

WAE 診断

Cisco WAE には、次の操作が可能な診断ユーティリティが含まれています。

- システムの正常性に関する診断チェックを実行し、推奨事項を作成する。
- エンジニアが問題をトラブルシューティングするために必要となる、必須のデータや正常性チェックレポートをすべて収集する。

このツールには、正常性チェックスクリプトを追加できる拡張可能なフレームワークが備わっています。追加の正常性チェックスクリプトは、Python またはシェルのいずれかに含めることができ、<wae-install-directory> /bin/diagnostics/ ディレクトリに配置する必要があります。

WAE 診断ツールの使用

WAE 診断は、wae-diagnostics コマンドを使用して実行できます。

（注） コマンドを実行する前に waerc をソースに設定します。

```
wae-diagnostics [-h] [-c] [-D] [-j] [-e] [-d] [--run-diagnostics] [-o OUT_DIR] -r RUN_DIR
 -i INSTALL_DIR
```

where

必須の引数	
-r RUN_DIR	run-dir：RUN_DIR 実行ディレクトリパス
INSTALLDIR	install-dir：INSTALL_DIR インストールディレクトリ パス
オプションの引数	
-h	help-- このヘルプメッセージを表示して終了
-c	collect-logs-- すべてのログとトラブルシューティング データを収集して照合。 --collect-db-files を使用して指定されていない限り、DB ファイルを除外します。
-D	collect-db-files-- db ファイルを収集
-j	java-stats-- Java スレッド統計情報を収集
-e	enable-debug-- ログレベルをデバッグに設定
-d	disable-debug-- デバッグログレベルを無効化
--run-diagnostics	WAE で診断を実行
-o OUT_DIR	out-dir--OUT_DIR 出力ディレクトリ パス。データアーカイブはこのディレクトリに作成されます。指定しない場合、データアーカイブは現在のディレクトリに作成されます。

例：

- 次のコマンドは、WAEインストールで診断チェックを実行し、ログを含む診断情報を収集します。

```
wae-diagnostics -r <run-directory-path> -i <install-dir-path>
```

- 次のコマンドは、WAEインストールで診断チェックを実行し、ログ、ネットワークデータ、およびJVMデータを含む診断情報を収集します。

```
wae-diagnostics -cDj -r <run-directory-path> -i <install-dir-path>
```

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。