



# Administration

---

This section contains the following topics:

- [Manage Users, on page 1](#)
- [Configure Aging, on page 2](#)
- [wae.conf, on page 2](#)
- [Configure High Availability, on page 9](#)
- [Configure LDAP, on page 13](#)
- [Status Dashboard, on page 16](#)
- [Understand WAE CLI Logging, on page 18](#)
- [Database Locking, on page 27](#)
- [Security, on page 28](#)
- [Clear WAE Operational Data, on page 30](#)
- [Back Up and Restore the WAE Configuration, on page 31](#)
- [WAE Diagnostics, on page 31](#)

## Manage Users


All users have the administrator role. The following procedure describes how to create, modify, and delete users.


---

**Step 1** From the WAE UI, click the User Manager icon ().

**Step 2** To add a user, click  and fill in all applicable fields.

**Step 3** To change a user's password:

- a) Select the user row and click .
- b) Update the password fields.
- c) Click **Save**.

**Step 4** To delete a user, click the user row and click .

## Configure Aging

By default, when a circuit, port, node, or link disappears from a network, it is permanently removed and must be rediscovered. To configure how long WAE retains these elements that have disappeared before they are permanently removed from the network, complete the following steps.



**Note** This is a global option that will be configured for all networks.

**Step 1** From the Expert Mode, navigate to `/wae:wae/components/aggregators` and click the **aging** tab.

- **aging-enabled**— Select true to enable aging.
- **l3-node-aging-duration**—Enter the time duration for which an L3 node must be kept in the network after it becomes inactive.
- **l3-port-aging-duration**—Enter the time duration for which an L3 port must be kept in the network after it becomes inactive.
- **l3-circuit-aging-duration**—Enter the time duration for which an L3 circuit must be kept in the network after it becomes inactive.

**Note** The value of `l3-node-aging-duration` must be greater than `l3-port-aging-duration` which in turn must be greater than `l3-circuit-aging-duration`.

- **l1-node-aging-duration**—Enter the time duration for which an L1 node must be kept in the network after it becomes inactive.
- **l1-port-aging-duration**—Enter the time duration for which an L1 port must be kept in the network after it becomes inactive.
- **l1-link-aging-duration**—Enter the time duration for which an L1 link must be kept in the network after it becomes inactive.

**Note** The value of `l1-node-aging-duration` must be greater than `l1-port-aging-duration` which in turn must be greater than `l1-link-aging-duration`.

**Step 2** Click **Commit**.

## wae.conf

`wae.conf` is an XML configuration file that is formally defined by a YANG model, `tailf-ncsconfig.yang`. This YANG file is included in the WAE distribution, as is a commented `wae.conf.example` file.

The `wae.conf` file controls the baseline of the WAE run time. You can change certain configuration parameters in the `wae.conf` file; for example, you can change the default port that WAE runs on (port 8080) to another port.



**Note** Make sure you restart WAE after you make any changes to `wae.conf` file.

Whenever you start or reload the WAE daemon, it reads its configuration from `./wae.conf` or `<waeruntime-directory>/etc/wae.conf`.

The following example shows `<waeruntime-directory>/etc/wae.conf` contents:

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

<ncs-config xmlns="http://tail-f.com/yang/taillf-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 taillf-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 taillf-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>
    </event-streams>
  </notifications>
</ncs-config>
```

```

    </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>

<!--
    These keys are used to encrypt values of the types
    tailf:des3-cbc-encrypted-string, tailf:aes-cfb-128-encrypted-string
    and tailf:aes-256-cfb-128-encrypted-string.
    For a deployment install it is highly recommended to change
    these numbers to something random (done by WAE "system install")
-->
<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
    <extra-listen>
        <ip>::</ip>
        <port>8080</port>
    </extra-listen>
        -->
    </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
    <extra-listen>
        <ip>::</ip>
        <port>8443</port>
    </extra-listen>
        -->
    </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
        <extra-listen>
            <ip>::</ip>
            <port>2022</port>
        </extra-listen>
        -->
    </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
            a separate Cisco Smart License.
        -->
        <enabled>>true</enabled>
    </lsa>
</large-scale>

</ncs-config>

```

The default values for many configuration parameters are defined in the YANG file. See [wae.conf Configuration Parameters](#).

## Configure High Availability

Cisco WAE supports High Availability (HA) with automatic failover. Two instances of WAE nodes are configured to run in parallel, where the primary node is configured in primary mode and the secondary node is configured in standby mode. The primary node listens to the connection from secondary nodes on port 4570 (or the port configured in the wae.conf file). Committed CDB data is mirrored to the secondary node at regular intervals. Note that any write operations to the CDB (NIMO operations, agent processes, or scheduler actions) which are performed on a node that is in standby mode will fail.

If the primary node fails, the secondary node will takeover as primary node. Once the primary mode is enabled on the secondary node, write operations are allowed, the CDB is rebuilt, and any scheduled jobs will run. It resumes operations that the primary node previously performed.

**Step 1** On both primary and secondary nodes, edit the `wae.conf` file to enable HA.

```
<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>
```

**Note** Make sure your `/etc/hosts` file is updated with the hostname to IP address mapping.

**Step 2** Restart Cisco WAE on both nodes using Supervisor

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

**Step 3** On both nodes, do one of the following:

- From the Cisco WAE CLI:

```
# 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>
```

On the primary, initiate `be-primary` and verify status:

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

Initiate `be-secondary` on the secondary node and verify status:

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

- From the WAE UI:

- a. Click the HA configuration icon.

**Note** Make sure your `/etc/hosts` file is updated with the hostname to IP address mapping

- b. Enter the N1 Node and N2 Node details.

**Note** Provide Fully Qualified Domain Name for the Node Name of both nodes.

- c. Enter the **Cluster-ID**.

- d. Select **be-primary**, **be-secondary** or **be-none**.

- e. Click **Secondary**.

**Note** Once a node is selected as secondary node, only WAE UI → HA Configuration and WAE UI → Status pages are enabled for that node.

When moving a node from **be-none** to **be-secondary**, make sure that there are no collections running on the node and that there are no agents/scheduler tasks configured.

**Step 4** The status of the node is displayed on the HA Configuration page.

**Note** After HA is configured on both nodes, any further configuration changes is allowed only on the primary node.

**Step 5** Set up passwordless ssh between both the nodes for data sync.

Generate authentication key

```
# 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
```

Copy the public key to the remote host

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

Alternatively if the server is not installed with openssh-clients (a package which provides ssh-copy-id command utility) you can copy the authentication key with the command:

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

**Note** In case of failover, the data sync happens in opposite direction and hence passwordless ssh has to be setup for both the nodes.

**Step 6** Schedule data replication using Global Scheduler - HA data sync on primary node to replicate files under `<wae-run-directory>/networks`, `<wae-run-directory>/agents` and any archives.

`rsync` utility is required for data replication.

Install `rsync` utility if its is not available in your system using the following command:

```
sudo yum install rsync
```

- Note**
- Once the primary node goes down, the secondary node comes up as primary. The primary node comes up as None once it is restored. You have to manually configure this node as Secondary again.
  - It is necessary to schedule the run collection for XTC based NIMOs to get the complete network model after failover. If the run collection is not scheduled, XTC based reactive changes will not work until collection is run manually.
  - When a Node is moved from None state to Secondary state, make sure that collection or agents are not running on that node.
  - Netflow workflow and layout-nimo are not supported under HA.

## Troubleshoot High Availability

There are two logs that should be looked at when troubleshooting HA configuration:

- `<wae_run_directory>/logs/wae-java-vm.log`

- `<wae_run_directory>/logs/devel.log`

The following table lists HA errors and their meanings.

Error Code	ENUM	Description
25	CONFD_ERR_HA_CONNECT	Failed to connect to a remote HA node.
26	CONFD_ERR_HA_CLOSED	A remote HA node closed its connection to WAE or there was a timeout waiting for a sync response from the primary during a <code>confd_ha_besecondary()</code> call.
27	CONFD_ERR_HA_BADFXS	A remote HA node has either a different set of FXS ports or different versions of FXS ports compared to WAE.
28	CONFD_ERR_HA_BADTOKEN	A remote HA node has a different token than WAE.
29	CONFD_ERR_HA_BADNAME	A remote HA node has a different name than the name captured in WAE.
30	CONFD_ERR_HA_BIND	There was a failure to bind the HA socket for incoming HA connections.
31	CONFD_ERR_HA_NOTICK	A remote HA node failed to produce the interval live ticks.

Example of `wae-java-vm.log` contents:

```
<ERR> 21-Sep-2018::10:55:15.127 compute9-wae ncs[2751]: - Failed to connect to primary:
host is unreachable
<ERR> 21-Sep-2018::10:55:27.135 compute9-wae ncs[2751]: - Failed to connect to primary:
host is unreachable
<INFO> 22-Sep-2018::10:54:53.681 HaActionCb Did-25-Worker-19: - Invoking beSlave with myNode
= compute8-wae.local primary = [compute9-wae.local : 192.0.0.81]
<INFO> 22-Sep-2018::10:54:53.782 HaApplicationComponent$1 Thread-110: - HA Notification:
HaNotification[HA_INFO_SECONDARY_INITIALIZED, nomaster=0, ha_node=null, cdb_init_by_copy=true,
be_seecondary_result=0]
```

Example of `devel.log` contents:

```
<DEBUG> 22-Sep-2018::10:54:53.747 compute8-wae ncs[1489]: ncs HA: Requested sync with
primary: ok
<INFO> 22-Sep-2018::10:54:53.778 compute8-wae ncs[1489]: devel-cdb Sync of configuration
db from primary complete, new transaction id is 1537-584870-864905@compute9-wae.local
<INFO> 22-Sep-2018::10:54:53.781 compute8-wae ncs[1489]: devel-cdb Loaded oper data from
./ncs-cdb/0.cdb (12.19 KiB data in 0.001s)
<INFO> 22-Sep-2018::10:54:53.782 compute8-wae ncs[1489]: ncs HA_INFO_SECONDARY_INITIALIZED
<DEBUG> 22-Sep-2018::10:54:53.783 compute8-wae ncs[1489]: devel-c action action() succeeded
for callpoint 'ha-point' path /wae:wae/ha-config
```

# Configure LDAP

Cisco WAE supports authentication of foreign users using Lightweight Directory Access Protocol (LDAP).

Before you configure LDAP on WAE:

- You should be familiar with the LDAP directory tree and its contents.
- Install and configure LDAP server and collection details.
- To use LDAPS protocol, get the SSL certificate and add it to a keystore.

## Commands to get and import SSL certificate

Save the self signed certificate to cert.pem file using the following command:

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

Get the default key-store path using the following command. Typically the default key-store path is /etc/pki/java/cacerts for CentOS 7 with open-jdk

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

Import cert into default key-store using following command

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

To troubleshoot LDAP configuration, view the following logs:

- LDAP configuration log—<wae\_run\_directory>/logs/wae-javavm.log
- LDAP authentication runtime log—<wae\_run\_directory>/logs/wae-ldap-auth.log

## Configure LDAP Using the CLI

### Before you begin

Confirm prerequisites are met as described in [Configure LDAP, on page 13](#).

**Step 1** Edit the wae.conf file to enable external authentication.

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

**Step 2** Restart WAE.

```
# wae --start
```

**Step 3** Configure LDAP server details using the WAE CLI.

### Example: LDAP configuration

```
# 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
```

### Example: LDAP configuration with SSL and the admin user

```
# 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
```

### Example: LDAP configuration for MS Active Directory Server

```
# 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
```

## Configure LDAP Using the WAE UI


### Before you begin

Confirm prerequisites are met as described in [Configure LDAP, on page 13](#).

**Step 1** Edit the wae.conf file to enable external authentication.

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

**Step 2** Restart WAE.

**Step 3** From the WAE UI, click the LDAP configuration icon ()

**Step 4** By default, the Enabled toggle switch is on. If not, enable use of the LDAP server for user authentication and toggle the switch on.

**Step 5** Enter the LDAP options. See [LDAP Configuration Options, on page 15](#) for more information.

**Step 6** Click **Save**.

## LDAP Configuration Options

*Table 1: LDAP Field Descriptions*

Field	Description
Protocol	<p>Protocol used to reach the LDAP server.</p> <ul style="list-style-type: none"> <li>• LDAP—Transmits communication in clear text.</li> <li>• LDAPS—Transmits communication that is encrypted and secure.</li> </ul> <p>Default value is LDAP.</p>
Server <ldap-server>	<p>LDAP server IP address or FQDN, which is the server's hostname with the DNS domain name appended to the end.</p> <p>FQDN format: &lt;LDAP_hostname&gt; . &lt;domain&gt; .com</p>
Port	<p>Port used to reach the LDAP server. For unencrypted authentication the default is TCP 389. For encrypted authentication the default is TCP 636.</p> <p>Default value is 389.</p>
Search Base <ldap-base-ou>	<p>This is the Distinguished Name of the base search OU for all user accounts that should have permission to login to the WAE Server.</p>

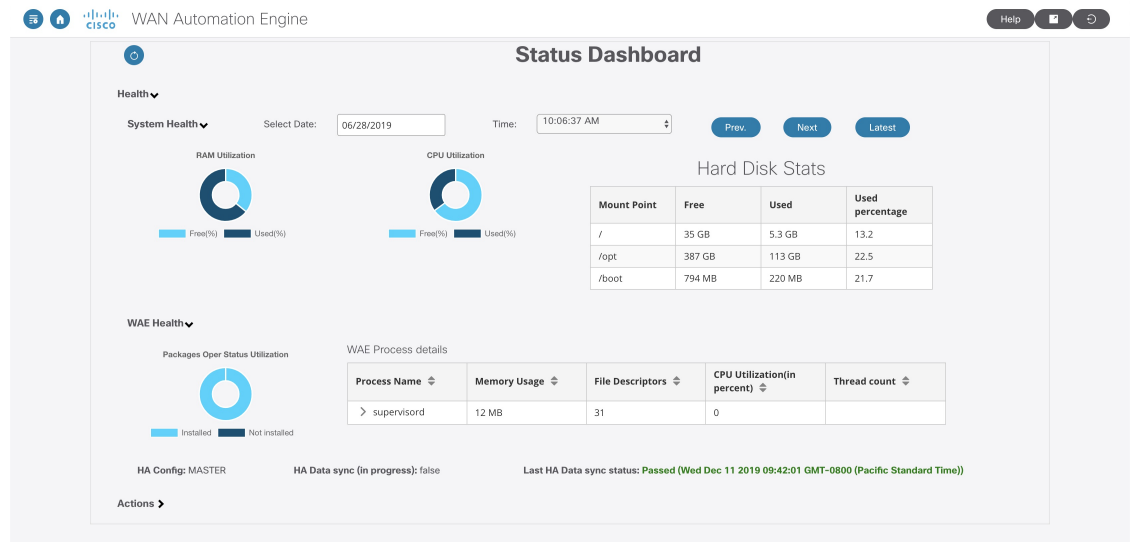
Field	Description
Principal Expression	<p>Default : <code>LDAP.Principal.Expr:(userPrincipalName={0})</code> ,</p> <p>The {0} token will be replaced by the user's input for username at the login page.</p> <p>The <code>userPrincipalName=</code> must match a User Objects' LDAP attribute that identifies the user under the LDAP search base.</p> <p>From the LDAP schema, use the User Unique Attribute <code>uid</code>.</p> <p>The WAE server will search all objects under the LDAP search base tree for:</p> <pre>uid=cisco-mate-user1</pre> <p>Common alternatives include <code>userPrincipalName</code> or <code>userName</code> etc.</p>
LDAP Auth Group	<p>Username/password used to perform LDAP search.</p> <p>Enable-password field is not used for LDAP. Enter any dummy value while configuring from UI.</p>
Keystore Path	Keystore path when SSL is enabled.
Keystore Pass	Keystore password.

## Status Dashboard

There are situations when Cisco WAE suddenly stops working or it crashes. There are times when traffic poller stops working, or there are problems that occur during archive. These issues sometimes are due to WAE system resources being used up. The status dashboard in WAE helps to address such situations by identifying the processes that cause system leaks or processes that completely use the resources.

To access the status dashboard, navigate to Cisco WAE UI, and click **Status Dashboard**.





The Status Dashboard is mainly divided into two sections:

- Health
- Actions

**Health** is further divided into **System Health** and **WAE Health**

**System Health** captures RAM utilization, CPU utilization and system level hard disk statistics. The tool runs once every 10 minutes and generates the statistics. Use the **Select Date** and **Time** fields to access the required report.

By default, the latest reports are displayed. Use **prev**, **next** buttons to navigate between reports.

The RAM utilization and CPU utilization charts display the used and free spaces. Hover over the used or free areas to read the actual % utilization.

**WAE Health** captures process level usage details like memory usage, file descriptors and CPU utilization. In case of any issue with a process, access all the relevant process level details using the date and time fields.

There are circumstances wherein after Cisco WAE is deployed, some packages remain uninstalled. **Packages Oper Status Utilization** chart gives the % of installed and uninstalled packages. Click the installed/not installed area to see the list of packages installed/not installed.

HA configuration, if enabled, shows if system is a primary or a secondary.

**Actions** section displays the status of NIMO actions and agent actions.

In **NIMO Actions** section, click the NIMO card to get further details of the status of the NIMOs.



**Note** NIMO action displays only current action, not historical action data.

The **Agent Actions** section displays the action status of a particular agent.

## Understand WAE CLI Logging

WAE has extensive logging functionality. WAE logs to the directory specified in the `wae.conf` file. The following are the most useful log files:

- `wae.log`—WAE daemon log; can be configured to syslog.
- `wae_err.log.1`, `wae_err.log.idx`, `wae_err.log.siz`—If the WAE daemon has a problem, this log contains debug information for support. Display the content with the command **wae --printlog wae\_err.log**.
- `audit.log`—Central audit log that covers all northbound interfaces; can be configured to syslog.
- `localhost:8080.access`—All http requests to the daemon. This is an access log for the embedded web server. This file adheres to the Common Log Format, as defined by Apache and others. This log is disabled by default and is not rotated; that is, use **logrotate(8)**.
- `devel.log`—Debug log for troubleshooting user-written code. This log is enabled by default and is not rotated; that is, use **logrotate(8)**. Use this log with the `java-vm` or `python-vm` logs. The user code logs in the `vm` logs and the corresponding library logs in `devel.log`. Disable this log in production systems. Can be configured to syslog.
- `wae-java-vm.log`, `wae-python-vm.log`—Log for code running in Java or Python VMs, such as service applications. Developers writing Java and Python code use this log (in combination with `devel.log`) for debugging.
- `netconf.log`, `snmp.log`—Log for northbound agents; can be configured to syslog.
- `rollbackNNNNN`—All WAE commits generate a corresponding rollback file. You can configure the maximum number of rollback files and file numbering in `wae.conf`.
- `xpath.trace`—XPath is used in many places, such as XML templates. This log file shows the evaluation of all XPath expressions. To debug XPath for a template, use the `pipe-target debug` in the CLI instead.
- `ned-cisco-ios-xr-pe1.trace`—If device trace is turned on, a trace file is created for each device. The file location is not configured in `wae.conf` but is configured when device trace is turned on, such as in the CLI.

## Syslog

Using BSD or IETF syslog format (RFC5424), WAE can syslog to a local or remote syslog server. You can use the `wae.conf` file to choose which logs to save to syslog: `ncs.log`, `devel.log`, `netconf.log`, or `snmp.log`.

The following example shows a common syslog configuration:

```
<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 Messages and Formats

The following table lists WAE syslog messages and formats.

Symbol	Format String	Comment
DAEMON_DIED	"Daemon ~s died"	An external database daemon closed its control socket.
DAEMON_TIMEOUT	"Daemon ~s timed out"	An external database daemon did not respond to a query.
NO_CALLPOINT	"no registration found for callpoint ~s of type=~s"	ConfD tried to populate an XML tree, but no code had registered under the relevant callpoint.
CDB_DB_LOST	"CDB: lost DB, deleting old config"	CDB found its data schema file but not its data files. CDB recovered by starting from an empty database.
CDB_CONFIG_LOST	"CDB: lost config, deleting DB"	CDB found its data files but not its schema file. CDB recovered by starting from an empty database.
CDB_UPGRADE_FAILED	"CDB: Upgrade failed: ~s"	Automatic CDB upgrade failed, meaning the data model was changed in a way that is not supported.
CDB_INIT_LOAD	"CDB load: processing file: ~s"	CDB is processing an initialization file.

Symbol	Format String	Comment
CDB_OP_INIT	"CDB: Operational DB re-initialized"	The operational database was deleted and reinitialized because of an upgrade or a corrupt file.
CDB_CLIENT_TIMEOUT	"CDB client (~s) timed out, waiting for ~s"	A CDB client failed to answer within the timeout period and was disconnected.
INTERNAL_ERROR	"Internal error: ~s"	A ConfD internal error occurred and should be reported to Cisco technical support.
AAA_LOAD_FAIL	"Failed to load AAA: ~s"	Failed to load the AAA data because the external database is misbehaving or AAA is mounted or populated badly.
EXTAUTH_BAD_RET	"External auth program (user=~s) ret bad output: ~s"	Authentication is external and the external program returned badly formatted data.
BRIDGE_DIED	"confd_aaa_bridge died - ~s"	ConfD is configured to start the confd_aaa_bridge and the C program died.
PHASE0_STARTED	"ConfD phase0 started"	ConfD has started its start phase 0.
PHASE1_STARTED	"ConfD phase1 started"	ConfD has started its start phase 1.
STARTED	"ConfD started vsn: ~s"	ConfD has started.
UPGRADE_INIT_STARTED	"Upgrade init started"	In-service upgrade initialization started.
UPGRADE_INIT_SUCCEEDED	"Upgrade init succeeded"	In-service upgrade initialization succeeded.
UPGRADE_PERFORMED	"Upgrade performed"	In-service upgrade was performed but not yet committed.
UPGRADE_COMMITTED	"Upgrade committed"	In-service upgrade was committed.
UPGRADE_ABORTED	"Upgrade aborted"	In-service upgrade was terminated.
CONSULT_FILE	"Consulting daemon configuration file ~s"	ConfD is reading its configuration file.
STOPPING	"ConfD stopping (~s)"	ConfD is stopping (for example, due to <b>confd --stop</b> ).
RELOAD	"Reloading daemon configuration"	Initiated daemon configuration reload.
BADCONFIG	"Bad configuration: ~s:~s: ~s"	confd.conf contains bad data.
WRITE_STATE_FILE_FAILED	"Writing state file failed: ~s: ~s (~s)"	Failed to write a state file.
READ_STATE_FILE_FAILED	"Reading state file failed: ~s: ~s (~s)"	Failed to read a state file.
SSH_SUBSYS_ERR	"ssh protocol subsys - ~s"	Client did not send the <b>\"subsystem\"</b> command correctly.

Symbol	Format String	Comment
SESSION_LIMIT	"Session limit of type '~s' reached, rejected new session request"	Session limit reached; new session request was rejected.
CONFIG_TRANSACTION_LIMIT	"Configuration transaction limit of type '~s' reached, rejected new transaction request"	Configuration transaction limit reached; new transaction request was rejected.
ABORT_CAND_COMMIT	"Aborting candidate commit, request from user, reverting configuration"	Terminating candidate commit due to user request. Reverting the configuration.
ABORT_CAND_COMMIT_TIMER	"Candidate commit timer expired, reverting configuration"	Candidate commit timer expired; reverting configuration.
ABORT_CAND_COMMIT_TERM	"Candidate commit session terminated, reverting configuration"	Candidate commit session terminated; reverting configuration.
ROLLBACK_REMOVE	"Found half created rollback0 file - removing and creating new"	Removing and recreating a rollback0 file that was found only half created.
ROLLBACK_REPAIR	"Found half created rollback0 file - repairing"	Repairing a rollback0 file that was found only half created.
ROLLBACK_FAIL_REPAIR	"Failed to repair rollback files"	Failed to repair a rollback file.
ROLLBACK_FAIL_CREATE	"Error while creating rollback file: ~s: ~s"	An error occurred while creating a rollback file.
ROLLBACK_FAIL_RENAME	"Failed to rename rollback file ~s to ~s: ~s"	Failed to rename a rollback file.
NS_LOAD_ERR	"Failed to process namespace ~s: ~s"	System failed to process a loaded namespace.
NS_LOAD_ERR2	"Failed to process namespaces: ~s"	System failed to process a loaded namespace.
FILE_LOAD_ERR	"Failed to load file ~s: ~s"	System failed to load a file in its load path.
FILE_LOADING	"Loading file ~s"	System is starting to load a file.
SKIP_FILE_LOADING	"Skipping file ~s: ~s"	System skipped a file.
FILE_LOAD	"Loaded file ~s"	System loaded a file.
LISTENER_INFO	"~s to listen for ~s on ~s:~s"	ConfD starts or stops to listen for incoming connections.
NETCONF_HDR_ERR	"Got bad NETCONF TCP header"	The clear text header that indicates users and groups was formatted badly.
LIB_BAD_VSN	"Got library connect from wrong version (~s, expected ~s)"	An application connecting to ConfD used a library version that does not match the ConfD version (for example, an old version of the client library).

Symbol	Format String	Comment
LIB_BAD_SIZES	"Got connect from library with insufficient keypath depth/keys support (~s/ ~s, needs ~s/~s)"	An application connecting to ConfD used a library version that cannot handle the depth and the number of keys used by the data model.
LIB_NO_ACCESS	"Got library connect with failed access check: ~s"	An access check failure occurred when an application connected to ConfD.
SNMP_NOT_A_TRAP	"SNMP gateway: Non-trap received from ~s"	A UDP package was received on the trap receiving port, but it's not an SNMP trap.
SNMP_TRAP_V1	"SNMP gateway: V1 trap received from ~s"	An SNMPv1 trap was received on the trap receiving port, but forwarding v1 traps is not supported.
SNMP_TRAP_NOT_FORWARDED	"SNMP gateway: Can't forward trap from ~s; ~s"	An SNMP trap was not forwarded.
SNMP_TRAP_UNKNOWN_SENDER	"SNMP gateway: Not forwarding trap from ~s; the sender is not recognized"	An SNMP trap was supposed to be forwarded, but the sender was not listed in confd.conf.
SNMP_TRAP_OPEN_PORT	"SNMP gateway: Can't open trap listening port ~s: ~s"	Could not open the port for listening to SNMP traps.
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"	An SNMP trap was received on the trap receiving port, but its definition is unknown.
XPATH_EVAL_ERROR1	"XPath evaluation error: ~s for ~s"	An error occurred while evaluating an xpath expression.
XPATH_EVAL_ERROR2	"XPath evaluation error: '~s' resulted in ~s for ~s"	An error occurred while evaluating an xpath expression.
CANDIDATE_BAD_FILE_FORMAT	"Bad format found in candidate db file ~s; resetting candidate"	The candidate database file has a bad format. The candidate database is reset to an empty database.
CANDIDATE_CORRUPT_FILE	"Corrupt candidate db file ~s; resetting candidate"	The candidate database file is corrupt and cannot be read. The candidate database is reset to an empty database.
MISSING_DES3CBC_SETTINGS	"DES3CBC keys were not found in confd.conf"	DES3CBC keys were not found in confd.conf.
MISSING_AESCFB128_SETTINGS	"AESCFB128 keys were not found in confd.conf"	AESCFB128 keys were not found in confd.conf.
SNMP_MIB_LOADING	"Loading MIB: ~s"	The SNMP agent is loading a MIB file.
SNMP_CANT_LOAD_MIB	"Can't load MIB file: ~s"	The SNMP agent failed to load a MIB file.

Symbol	Format String	Comment
SNMP_WRITE_STATE_FILE_FAILED	"Write state file failed: ~s: ~s"	Failed to write the SNMP agent state file.
SNMP_READ_STATE_FILE_FAILED	"Read state file failed: ~s: ~s"	Failed to read the SNMP agent state file.
SNMP_REQUIRES_CDB	"Can't start SNMP. CDB is not enabled"	CDB must be enabled before the SNMP agent can start.
FXS_MISMATCH	"Fxs mismatch, slave is not allowed"	A secondary connected to a primary with different fxs files.
TOKEN_MISMATCH	"Token mismatch, slave is not allowed"	A secondary connected to a primary with a bad authorization token.
HA_SLAVE_KILLED	"Slave ~s killed due to no ticks"	A secondary node did not produce its ticks.
HA_DUPLICATE_NODEID	"Nodeid ~s already exists"	A secondary arrived with a node ID that already exists.
HA_FAILED_CONNECT	"Failed to connect to master: ~s"	An attempted library to become a secondary call failed because the secondary could not connect to the primary.
HA_BAD_VSN	"Incompatible HA version (~s, expected ~s), slave is not allowed"	A secondary connected to a primary with an incompatible HA protocol version.
NETCONF	"~s"	NETCONF traffic log message.
DEVEL_WEBUI	"~s"	Developer web UI log message.
DEVEL_AAA	"~s"	Developer AAA log message.
DEVEL_CAPI	"~s"	Developer C API log message.
DEVEL_CDB	"~s"	Developer CDB log message.
DEVEL_CONFD	"~s"	Developer ConfD log message.
DEVEL_SNMPGW	"~s"	Developer SNMP gateway log message.
DEVEL_SNMPA	"~s"	Developer SNMP agent log message.
NOTIFICATION_REPLAY_STORE_FAILURE	"~_s"	A failure occurred in the built-in notification replay store.
EVENT_SOCKET_TIMEOUT	"Event notification subscriber with bitmask ~s timed out, waiting for ~s"	An event notification subscriber did not reply within the configured timeout period.
EVENT_SOCKET_WRITE_BLOCK	"~s"	Write on an event socket was blocked for too long.
COMMIT_UN_SYNCED_DEV	"Committed data towards device ~s which is out of sync"	Data was committed toward a device with a bad or unknown sync state.

Symbol	Format String	Comment
NCS_SNMP_INIT_ERR	"Failed to locate snmp_init.xml in loadpath ~s"	Failed to locate snmp_init.xml in the load path.
NCS_JAVA_VM_START	"Starting the NCS Java VM"	Starting the NCS Java VM.
NCS_JAVA_VM_FAIL	"The NCS Java VM ~s"	An NCS Java VM failure or timeout occurred.
NCS_PACKAGE_SYNTAX_ERROR	"Failed to load NCS package: ~s; syntax error in package file"	Syntax error in package file.
NCS_PACKAGE_DUPLICATE	"Failed to load duplicate NCS package ~s: (~s)"	Duplicate package found.
NCS_PACKAGE_COPYING	"Copying NCS package from ~s to ~s"	A package was copied from the load path to a private directory.
NCS_PACKAGE_UPGRADE_ABORTED	"NCS package upgrade failed with reason '~s'"	The CDB upgrade was terminated, implying that the CDB is untouched. However, the package state changed.
NCS_PACKAGE_BAD_NCS_VERSION	"Failed to load NCS package: ~s; requires NCS version ~s"	Bad NCS version for the package.
NCS_PACKAGE_BAD_DEPENDENCY	"Failed to load NCS package: ~s; required package ~s of version ~s is not present (found ~s)"	Bad NCS package dependency.
NCS_PACKAGE_CIRCULAR_DEPENDENCY	"Failed to load NCS package: ~s; circular dependency found"	Circular NCS package dependency.
CLI_CMD	"CLI '~s'"	User executed a CLI command.
CLI_DENIED	"CLI denied '~s'"	Due to permissions, a user was denied from executing a CLI command.
BAD_LOCAL_PASS	"Provided bad password"	A locally configured user provided a bad password.
NO_SUCH_LOCAL_USER	"no such local user"	A non existing local user tried to log in.
PAM_LOGIN_FAILED	"pam phase ~s failed to login through PAM: ~s"	A user failed to log in through PAM.
PAM_NO_LOGIN	"failed to login through PAM: ~s"	A user failed to log in through PAM.
EXT_LOGIN	"Logged in over ~s using externalauth, member of groups: ~s~s"	An externally authenticated user logged in.
EXT_NO_LOGIN	"failed to login using externalauth: ~s"	External authentication failed for a user.
GROUP_ASSIGN	"assigned to groups: ~s"	A user was assigned to a set of groups.



Symbol	Format String	Comment
GROUP_NO_ASSIGN	"Not assigned to any groups - all access is denied"	A user was logged in but was not assigned to any groups.
MAAPI_LOGOUT	"Logged out from maapi ctx=~s (~s)"	A management agent API (MAAPI) user was logged out.
SSH_LOGIN	"logged in over ssh from ~s with authmeth::~s"	A user logged into ConfD's built-in SSH server.
SSH_LOGOUT	"Logged out ssh <~s> user"	A user was logged out from ConfD's built-in SSH server.
SSH_NO_LOGIN	"Failed to login over ssh: ~s"	A user failed to log in to ConfD's built-in SSH server.
NOAAA_CLI_LOGIN	"logged in from the CLI with aaa disabled"	A user used the --noaaa flag to confd_cli.
WEB_LOGIN	"logged in through Web UI from ~s"	A user logged in through the web UI.
WEB_LOGOUT	"logged out from Web UI"	A web UI user logged out.
WEB_CMD	"WebUI cmd '~s'"	A user executed a web UI command.
WEB_ACTION	"WebUI action '~s'"	A user executed a web UI action.
WEB_COMMIT	"WebUI commit ~s"	A user performed a web UI commit.
SNMP_AUTHENTICATION_FAIL	"ESDNMP authentication failed: ~s"	An SNMP authentication failed.
LOGIN_REJECTED	"~s"	Authentication for a user was rejected by application callback.
COMMIT_INFO	"commit ~s"	Information about configuration changes committed to the running data store.
CLI_CMD_DONE	"CLI done"	CLI command finished successfully.
CLI_CMD_ABORTED	"CLI aborted"	CLI command terminated.
NCS_DEVICE_OUT_OF_SYNC	"NCS device-out-of-sync Device '~s' Info '~s'"	A check-sync action reported out-of-sync for a device.
NCS_SERVICE_OUT_OF_SYNC	"NCS service-out-ofsync Service '~s' Info '~s'"	A check-sync action reported out-of-sync for a service.
NCS_PYTHON_VM_START	"Starting the NCS Python VM"	Starting the NCS Python VM.
NCS_PYTHON_VM_FAIL	"The NCS Python VM ~s"	The NCS Python VM failed or timed out.
NCS_SET_PLATFORM_DATA_ERRORS	"NCS Device '~s' failed to set platform data Info '~s'"	The device failed to set the platform operational data at connect.
NCS_SMART_LICENSING_START	"Starting the NCS Smart Licensing Java VM"	Starting the NCS Smart Licensing Java VM.

Symbol	Format String	Comment
NCS_SMART_LICENSING_FAIL	"The NCS Smart Licensing Java VM ~s"	The NCS Smart Licensing Java VM failed or timed out.
NCS_SMART_LICENSING_GLOBAL_NOTIFICATION	"Smart Licensing Global Notification: ~s"	Smart Licensing global notification.
NCS_SMART_LICENSING_ENTITLEMENT_NOTIFICATION	"Smart Licensing Entitlement Notification: ~s"	Smart Licensing entitlement notification.
NCS_SMART_LICENSING_EVALUATION_COUNTDOWN	"Smart Licensing evaluation time remaining: ~s"	Smart Licensing evaluation time remaining.
DEVEL_SLS	"~s"	Developer Smart Licensing API log message.
JSONRPC_REQUEST	"JSON-RPC: '~s' with JSON params ~s"	JSON-RPC method requested.
DEVEL_ECONFD	"~s"	Developer econfd API log message.
CDB_FATAL_ERROR	"fatal error in CDB: ~s"	CDB encountered an unrecoverable error.
LOGGING_STARTED	"Daemon logging started"	Logging subsystem started.
LOGGING_SHUTDOWN	"Daemon logging terminating, reason: ~s"	Logging subsystem terminated.
REOPEN_LOGS	"Logging subsystem, reopening log files"	Logging subsystem reopened log files.
OPEN_LOGFILE	"Logging subsystem, opening log file '~s' for ~s"	Indicate target file for certain type of logging.
LOGGING_STARTED_TO	"Writing ~s log to ~s"	Write logs for a subsystem to a specific file.
LOGGING_DEST_CHANGED	"Changing destination of ~s log to ~s"	The target log file will change to another file.
LOGGING_STATUS_CHANGED	"~s ~s log"	Notify a change of logging status (enabled/disabled) for a subsystem.
ERRLOG_SIZE_CHANGED	"Changing size of error log (~s) to ~s (was ~s)"	Notify a change of log size for an error log.
CGI_REQUEST	"CGI: '~s' script with method ~s"	CGI script requested.
MMAP_SCHEMA_FAIL	"Failed to setup the shared memory schema"	Failed to set up the shared memory schema.
KICKER_MISSING_SCHEMA	"Failed to load kicker schema"	Failed to load the kicker schema.
JSONRPC_REQUEST_IDLE_TIMEOUT	"Stopping session due to idle timeout: ~s"	JSON-RPC idle timeout.
JSONRPC_REQUEST_ABSOLUTE_TIMEOUT	"Stopping session due to absolute timeout: ~s"	JSON-RPC absolute timeout.

# Database Locking

This section explains the different locks that exist in WAE and how they interact.

## Global Locks

The WAE management backplane keeps a lock on the data store: *running*. This lock is known as the global lock and provides a mechanism to grant exclusive access to the data store. The global lock is the only lock that can explicitly be taken through a northbound agent—for example, by the NETCONF <lock> operation—or by calling `Maapi.lock()`.

A global lock can be taken for the entire data store, or it can be a partial lock (for a subset of the data model). Partial locks are exposed through NETCONF and MAAPI.

An agent can request a global lock to ensure that it has exclusive write access. When an agent holds a global lock, no one else can write to that data store. This behavior is enforced by the transaction engine. A global lock on *running* is granted to an agent if there are no other lock holders (including partial locks), and if all data providers approve the lock request. Each data provider (CDB or external data provider) has its `lock()` callback invoked to refuse or accept the lock. The output of `ncs --status` includes the lock status.

## Transaction Locks

A northbound agent starts a user session towards the WAE management backplane. Each user session can then start multiple transactions. A transaction is either read/write or read-only.

The transaction engine has its internal locks toward the running data store. These transaction locks exist to serialize configuration updates toward the data store and are separate from global locks.

When a northbound agent wants to update the running data store with a new configuration, it implicitly grabs and releases the transactional lock. The transaction engine manages the lock as it moves through the transaction state machine. No API exposes the transactional lock to the northbound agent.

When the transaction engine wants to take a lock for a transaction (for example, when entering the validate state), it first checks that no other transaction has the lock. It then checks that no user session has a global lock on that data store. Finally, it invokes each data provider with a `transLock()` callback.

## Northbound Agents and Global Locks

In contrast to implicit transactional locks, some northbound agents expose explicit access to global locks. The management API exposes global locks by providing `Maapi.lock()` and `Maapi.unlock()` methods (and the corresponding `Maapi.lockPartial()` `Maapi.unlockPartial()` for partial locking). Once a user session is established (or attached to), these functions can be called.

In the CLI, global locks are taken when entering different configure modes, as follows:

- **config exclusive**—Takes the running data store global lock.
- **config terminal**—Does not grab any locks.

The CLI keeps the global lock until the configure mode is exited.

The Expert Mode behaves in the same way as the CLI: it has edit tabs called **Edit private** and **Edit exclusive**, which correspond to the CLI modes described above.

The NETCONF agent translates the <lock> operation into a request for a global lock for the requested data store. Partial locks are also exposed through the partial-lock rpc.

## External Data Providers and CDB

An external data provider is not required to implement the lock() and unlock() callbacks. WAE never tries to initiate the transLock() state transition toward a data provider while a global lock is taken. The reason for a data provider to implement the locking callbacks is if someone else can write to the data provider's database.

CDB ignores the lock() and unlock() callbacks (because the data provider interface is the only write interface towards it).

CDB has its own internal locks on the database. The running data store has a single write lock and multiple read locks. It is not possible to grab the write lock on a data store while there are active read locks on it. The locks in CDB exist to ensure that a reader always gets a consistent view of the data. (Confusion occurs if another user deletes configuration nodes in between calls to getNext() on YANG list entries.)

During a transaction transLock() takes a CDB read lock toward the transaction's data store and writeStart() tries to release the read lock and grab the write lock instead. A CDB external reader client implicitly takes a CDB read lock between Cdb.startSession() and Cdb.endSession(). This means that while a CDB client is reading, a transaction cannot pass through writeStart(). Conversely, a CDB reader cannot start while a transaction is in between writeStart() and commit() or abort().

The operational store in CDB does not have any locks; WAE's transaction engine can only read from it. CDB client writes are atomic per write operation.

## Lock Impact on User Sessions

When a session tries to modify a data store that is locked, it fails. For example, the CLI might print:

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

Because some locks are short-lived (such as a CDB read lock), WAE is configured by default to retry the failed operation for a configurable length of time. If the data store remains locked after this time, the operation fails.

To configure the retry timeout, set the /ncs-config/commit-retry-timeout value in wae.conf.

## Security

WAE requires privileges to perform certain tasks. Depending on the target system, the following tasks might require root privileges:

- Binding to privileged ports. The wae.conf configuration file specifies which port numbers WAE should bind(2) to. If a port number is lower than 1024, WAE usually requires root privileges unless the target operating system allows WAE to bind to these ports as a non-root user.

- If PAM is used for authentication, the program installed as `$NCS_DIR/lib/ncs/priv/pam/epam` acts as a PAM client. Depending on the local PAM configuration, this program might require root privileges. If PAM is configured to read the local `passwd` file, the program must either run as root, or be `setuid root`. If the local PAM configuration instructs WAE to run for example `pam_radius_auth`, root privileges might not be required, depending on the local PAM installation.
- If the CLI is used to create CLI commands that run executables, modify the permissions of the `$NCS_DIR/lib/ncs/priv/ncs/cmdptywrapper` program.

To run an executable as root or as a specific user, make `cmdptywrapper` `setuid root`:

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

Failing that, all programs are executed as the user running the WAE daemon. If that user is root, you need not perform the `chmod` operations above.

Failing that, all programs are executed as the user running the `confd` daemon. If that user is root, you need not perform the preceding `chmod` operations.

For executables that run via actions, modify the permissions of the `$NCS_DIR/lib/ncs/priv/ncs/cmdwrapper` program:

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

WAE can be instructed to terminate NETCONF over clear text TCP, which is useful for debugging (NETCONF traffic can be captured and analyzed) and when providing a local proprietary transport mechanism other than SSH. Clear text TCP termination is not authenticated; the clear text client simply tells WAE which user the session should run as. The assumption is that authentication is already done by an external entity, such as an SSH server. If clear text TCP is enabled, WAE must bind to `localhost (127.0.0.1)` for these connections.

Client libraries connect to WAE. For example, the CDB API is TCP-based and a CDB client connects to WAE. WAE learns which address to use for these connections through the `wae.conf` parameters `/ncs-config/ncs-ipc-address/ip` (the default address is `127.0.0.1`) and `/ncs-config/ncs-ipcaddress/port` (the default port is `4565`).

WAE multiplexes different kinds of connections on the same socket (IP and port combination). The following programs connect on the socket:

- Remote commands, such as `ncs --reload`.
- CDB clients.
- External database API clients.
- Management agent API (MAAPI) clients.
- The `ncs_cli` program.

By default, the preceding programs are considered trusted. MAAPI clients and the `ncs_cli` authenticate users before connecting to WAE. CDB clients and external database API clients are considered trusted and do not have to authenticate.

Because the `ncs-ipc-address` socket allows full, unauthenticated access to the system, it is important to ensure that the socket is not accessible from untrusted networks. You can also restrict access to the `ncs-ipc-address` socket by means of an access check. See [Restrict Access to the IPC Port, on page 30](#).

## Restrict Access to the IPC Port

By default, clients connecting to the IPC port are considered trusted; no authentication is required. To prevent remote access, WAE relies on the use of 127.0.0.1 for `/ncs-config/ncs-ipc-address/ip`. However, you can restrict access to the IPC port by configuring an access check.

To enable the access check, set the `wae.conf` element `/ncs-config/ncs-ipc-accesscheck/enabled` to **true**, and specify a filename for `/ncs-config/ncs-ipc-accesscheck/filename`. The file should contain a shared secret (a random-character string). Clients connecting to the IPC port must provide a challenge handshake before they are granted access to WAE functions.




---

**Note** The access permissions on this file must be restricted via OS file permissions, such that the file can only be read by the WAE daemon and client processes that are allowed to connect to the IPC port. For example, if both the daemon and the clients run as root, the file can be owned by root and have only "read by owner" permission (mode 0400). Another possibility is to create a group that only the daemon and the clients belong to, set the group ID of the file to that group, and have only "read by group" permission (mode 040).

---

To provide the secret to the client libraries and instruct them to use the access check handshake, set the environment variable `NCS_IPC_ACCESS_FILE` to the full path name of the file that contains the secret. This is sufficient for all clients mentioned above; there is no need to change the application code to enable this check.




---

**Note** The access check must be either enabled or disabled for both the daemon and the clients. For example, if the `wae.conf` element `/ncsconfig/ncs-ipc-access-check/enabled` is not set to **true**, but clients are started with the environment variable `NCS_IPC_ACCESS_FILE` pointing to a file with a secret, the client connections fail.

---

## Clear WAE Operational Data

To clear WAE operational data from the database, you must delete the model, 11-model, from the respective NIMO network models. Then, delete the device tree. If your NIMO network model has layouts, delete those layouts from the NIMO network models.

The following example commands show how to clear operational data from the `as64002` network model and device tree:

```
delete networks network as64002 model
delete networks network as64002 layouts
delete networks network as64002 11-model
delete devices device *
commit
```

## Back Up and Restore the WAE Configuration

With the YANG run-time framework, you can easily back up and restore the WAE configuration. We recommend that you back up the WAE configuration before starting any collection (that is, before any operational data is populated).

- To back up a WAE configuration:

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

The preceding command backs up both the configuration data and the operational data. To back up only the configuration data, you must clear the operational data from the database as described in [Clear WAE Operational Data, on page 30](#). Be careful before clearing operational data in a production environment, because all operational data is deleted.

- To restore a WAE configuration:

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

## WAE Diagnostics

Cisco WAE includes a diagnostics utility that can:

- run diagnostic checks on the health of the system and make recommendations.
- collect all required data/health check reports that might be required for engineers to troubleshoot the issue.

The tool has an extensible framework where additional health-check scripts can be added. Additional health-check scripts can either be in python/shell and must be placed under `<wae-install-directory>/bin/diagnostics/ directory`.

## WAE Diagnostic Tool usage

WAE diagnostics can be executed using the `wae-diagnostics` command:



**Note** Source `waerc` before running the command.

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

where

Required arguments	
<code>-r RUN_DIR</code>	<code>run-dir--RUN_DIR run directory path</code>

-i INSTALL_DIR	install-dir--INSTALL_DIR install directory path
<b>Optional arguments</b>	
-h	help--Show this help message and exit
-c	collect-logs--Collects and collates all logs and troubleshooting data. Excludes DB files unless specified using --collect-db-files
-D	collect-db-files--Collects db files
-j	java-stats--Collects java thread stats
-e	enable-debug--Sets logs levels to debug
-d	disable-debug--Disables debug log level
--run-diagnostics	runs diagnostics on WAE
-o OUT_DIR	out-dir--OUT_DIR output directory path. The data archive will be created in this directory, if not specified, the data archive will be created in current directory.

**Sample:**

- The following command runs diagnostic checks on the WAE installation, collect diagnostics information including logs.

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

- The following command runs diagnostic checks on the WAE installation, collect diagnostics information including logs, network data and JVM data.

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