entuity
Taking the Work out of Network Management

Entuity® 16.5

Entuity System Administrator Reference Manual

Entuity empowers service providers, systems integrators, and enterprises with network control and predictability foundational to meeting any of today's complex IT infrastructure challenges. Entuity provides a succinct suite of the most important functionality for network management – inventory, fault, and performance management – but presented in an easy to use, quick to deploy format.

North America Headquarters
4 Mount Royal Avenue
Suite 340
Marlborough, MA 01752
Tel: +1 508 357 6344
Fax: +1 508 357 6358

EMEA Headquarters
9a Devonshire Square
London,
EC2M 4YN
Tel: +44 (0)20 7444 4800
Fax: +44 (0)20 7444 4808

Entuity

Contents

Figures

Tables

1 Entuity Reference Materials

These Entuity reference materials provide access to information useful to system administrators wanting to understand, or adjust, Entuity configuration. Specifically it details:

- Descriptions of Entuity system processes, utilities and third party tools.
- System files, including configuration options.
- Generic trap definitions, detailing the OIDs and trap formats of generic standard and standard enterprise traps. Entuity identifies the OID substring and then the trap number, from which it can generate an appropriate event in Event Viewer.
- Internal Entuity identifiers. Entuity uses a series of codes to identify the types of objects it manages. These internal codes are sometimes useful when troubleshooting or integrating with Entuity.
- Entuity RESTful API implementation.
- Entuity data model a knowledge of which is useful when developing Data Export functionality or User Defined Polling.

For details on managed object attributes refer to the *Entuity User Reference Manual*, useful for understanding the relevance of attribute values. The *Entuity User Reference Manual* manual groups attributes according to their parent type, for example:

- *Device Attributes*
- *Port Attributes*
- *Availability Monitoring Attributes*
- *Virtual Platform Attributes*.

2 Entuity System Processes and Utilities

This section gives brief details of the system processes and tools that monitor and manage the Entuity environment. Care should always be taken when running Entuity processes from the command line or using the supplied utilities tools. When in doubt always consult your Entuity support before undertaking any actions.

Where mentioned, the processes generate messages in log files, whether to provide information or flag errors. These log files will automatically wrap when they have reached a pre-determined size.

List of Entuity Processes and Utilities

The system processes are listed here, together with how they are called and whether they are always running.

Processes	Run By/From	Always Running	Location
`applicationMonitor`	`starteye`	Yes	*entuity_home*`\bin`
`authtool` (.bat)	manual	No	*entuity_home*`\bin`
`autoDiscovery`	Application Server (tomcat)	No	*entuity_home*`\bin`
`backup`	`provost`, manual	No	*entuity_home*`\bin`
`cfgdigest`			*entuity_home*`\lib\tools`
`changeState`			*entuity_home*`\bin`
`checkvcs`			*entuity_home*`\bin`
`checkLicense`			*entuity_home*`\bin`
`configure`			*entuity_home*`\install`
`customPoller`		Configurable	*entuity_home*`\lib\tools`
`dbcheck`	`starteye`, `configure`, manual	No	*entuity_home*`\bin`
`devDefunct`	`provost`	No	*entuity_home*`\lib\tools`
`deviceDelete`			*entuity_home*`\lib\tools`
`devpoller`	Adding of a device	No	*entuity_home*`\bin`
`devsysman`	`provost`	No	*entuity_home*`\bin`
`diskMonitor`	`starteye`	Yes	*entuity_home*`\bin`
`dnsproxy`	`provost`	No	*entuity_home*`\bin`
`DsKernelStatic`	`starteye`	Yes	*entuity_home*`\bin`

Table 1 Entuity System Processes and Utilities

Processes	Run By/From	Always Running	Location
dumpipnettoport			*entuity_home*\lib\tools
dumpiptodev			*entuity_home*\lib\tools
dumpvip			*entuity_home*\lib\tools
duplexman	provost	No	*entuity_home*\bin
encode_keychange			*entuity_home*\lib\tools
eyepoller			*entuity_home*\bin
FixNewBinVendor			*entuity_home*\lib\tools
getDownstream			*entuity_home*\lib\tools
hostIdent			*entuity_home*\lib\tools
httpd	Always Running	Yes	*entuity_home*\lib\apache\bin
install			ISO image
instService			*entuity_home*\bin
ipman	provost	No	*entuity_home*\bin
jasperStudio			*entuity_home*\bin
kill			*entuity_home*\lib\tools
licenseSrvr	starteye	Yes	*entuity_home*\bin
macman	provost	No	*entuity_home*\bin
macScheduler	starteye	Yes	*entuity_home*\bin
myisamchk			*entuity_home*\database\bin
myisampack			*entuity_home*\database\bin
mysql			*entuity_home*\database\bin
mysqladmin			*entuity_home*\database\bin
mysqlcheck			*entuity_home*\database\bin
mysqld-nt			*entuity_home*\database\bin
mysqldump			*entuity_home*\database\bin
mysqlimport			*entuity_home*\database\bin
mysqlshow			*entuity_home*\database\bin
newcommunity			*entuity_home*\lib\tools
nicman	provost	No	*entuity_home*\bin
perror			*entuity_home*\database\bin
probity			*entuity_home*\lib\tools
prodigy	provost	No	*entuity_home*\bin
profluent	provost	No	*entuity_home*\bin
prole	provost	No	*entuity_home*\bin

Table 1 Entuity System Processes and Utilities

Processes	Run By/From	Always Running	Location
proliferate	autoDiscovery, manual	N0	*entuity_home*\bin
prolifsys			*entuity_home*\bin
prologV2	Always Running	Yes	*entuity_home*\bin
protean	provost	No	*entuity_home*\bin
provost	starteye	Yes	*entuity_home*\bin
replace			*entuity_home*\database\bin
restore			*entuity_home*\bin
rollLog			*entuity_home*\lib\tools
runbg			*entuity_home*\bin
search			*entuity_home*\bin
setupProle			*entuity_home*\bin
showdevs			*entuity_home*\lib\tools
slalogger			*entuity_home*\bin
snmpbulkget			*entuity_home*\lib\tools
snmpdelta			*entuity_home*\lib\tools
snmpdump			*entuity_home*\lib\tools
snmpget			*entuity_home*\lib\tools
snmpgetnext			*entuity_home*\lib\tools
snmpset			*entuity_home*\lib\tools
snmpstatus			*entuity_home*\lib\tools
snmptable			*entuity_home*\lib\tools
snmptranslate			*entuity_home*\lib\tools
snmpusm			*entuity_home*\lib\tools
snmpvacm			*entuity_home*\lib\tools
snmpwalk			*entuity_home*\lib\tools
start			*entuity_home*\bin
starteotssvr	Always Running	Yes	*entuity_home*\bin
starteye			*entuity_home*\bin
stop			*entuity_home*\bin
stopeye			*entuity_home*\bin
stpman	provost	No	*entuity_home*\bin
swdoc			*entuity_home*\bin
swmaint			*entuity_home*\bin

Table 1 Entuity System Processes and Utilities

Processes	Run By/From	Always Running	Location
sysLogger	starteye	Yes	*entuity_home*\bin
TestMsgFilter			*entuity_home*\lib\tools
ticker	Always Running	Yes	*entuity_home*\bin
trapsplit	System Administrator	Not Run.	*entuity_home*\bin
trashtable			*entuity_home*\lib\tools
trendmigrate			*entuity_home*\lib\tools
vendinfo			*entuity_home*\lib\tools
viewserver	starteye		*entuity_home*\bin
vipman	provost	No	*entuity_home*\bin
vtpDomainTool			*entuity_home*\lib\tools
vtpman	provost	No	*entuity_home*\bin

Table 1 Entuity System Processes and Utilities

applicationMonitor

Location *entuity_home*\bin
Type process, runs every 120 seconds
Invoked By starteots
User Invocation Command line
Invoked Processes n/a
Configured Through web UI, entuity.cfg, startup_O/S.cfg
Log File *entuity_home*\log\applicationMonitor.log[1..4]

Description

applicationMonitor performs all forms of availability monitoring, i.e. device, server and application availability. Full functionality is available for devices with IPv4 management addresses, with currently more limited support for devices with IPv6 management addresses.

When Entuity monitors a device using IPv6, then applicationMonitor monitors the device management address using ICMPv6. applicationMonitor can raise events when the management address fails to respond, but does not perform traceroute or route cause analysis. Also, applicationMonitor does not monitor other IPv6 addresses on the device.

When there are IPv4 addresses on a device with an IPv6 management address, Entuity only considers the device as down when all of the addresses are unreachable.

Entuity currently supports application monitoring using IPv4 and does not support application monitoring using IPv6.

authtool

Location	*entuity_home*\bin
Type	Utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

authtool is intended to assist testing of external user authentication configurations and management of the Entuity emergency access user accounts. In Windows environments it is a batch file, `authtool.bat`.

Usage, Syntax and Options

The general syntax for this tool is:

```
authtool [-d] actionName <arguments>
```

where:

- *-d* is optional and specifies verbose output.
- *actionName* is the name of action to perform.
- *arguments* specify input to that action and are specific for that action. In many cases if arguments to the action are not supplied, authtool prompts for their entry.

Syntax Options

- list

  Lists emergency access user accents used to logon in an emergency situation.

  ```
  ./authtool list
  Emergency access is enabled
  Users:
    eUser
    root
  Total users:2
  ```

- check

  Checks whether user is able to logon in emergency situation. You must enter the emergency access user name and password.

  ```
  ./authtool check
  name of the user must be present and non-empty
  ```

```
Please enter name of the user:
root
user's password must be present and non-empty
Please enter user's password:
root
Emergency access is enabled
User 'root' is allowed to connect
```

- passwd

  Creates new, or updates an existing, emergency access user. To access this function you must enter a valid Entuity administrator username and password, and then specify the emergency access username and password.

```
./authtool passwd
name of an administrator user must be present and non-empty
Please enter name of an administrator user:
admin
administrator user's password must be present and non-empty
Please enter administrator user's password:
admin
name of the user must be present and non-empty
Please enter name of the user:
root
user's password must be present and non-empty
Please enter user's password:
root
Please re-enter user's password:
root
Emergency access is enabled
Password set for the user 'root'
```

- delete

  Deletes the named emergency access user profile.

```
./authtool delete
name of the user to delete must be present and non-empty
Please enter name of the user to delete:
root
Emergency access is enabled
Are you sure you want to delete user named 'root'? [yes/no]:
```

```
yes
```

- serverAccess

  authtool serverAccess allows you to check user access module for a particular user, and optionally specify the user group.

  ```
  authtool serverAccess user=jsmith groups=operation
  Testing server access for user 'jsmith' as member of:
    operation
  Access to server allowed
  ```

- mapping

  Performs mapping of supplied attributes to groups.

  You invoke mapping action as follows:

  ```
  authtool mapping attributeName=attributeValue
  attributeName=attributeValue
  ```

  For example to invoke authtool:

  ```
  authtool mapping userName=cwilliams groups="Network Admin"
  ```

  You can also run authtool mapping just against the group:

  ```
  authtool -d mapping groups=developers
  ```

- logon

  Once you have configured external authentication, or are in the process of doing so, you can test the user logon configuration, with the authtool logon function:

  ```
  authtool logon [user=username] [password=password]
  ```

- ldaptree

  Displays the whole LDAP tree, with the option of listing the details of one entry.

  ```
  authtool ldaptree [url=] [user=username] [password=password] [basedn=]
  [entry=]
  ```

  This example shows an LDAP tree for an example LDAP implementation:

  ```
  # ./authtool ldaptree url=ldap://10.44.3.73
  o=nokia
    ou=groups,o=nokia
      cn=i_ext_s_axs_tool_admin,ou=groups,o=nokia
      cn=i_ext_s_axs_tool_user,ou=groups,o=nokia
    ou=people,o=nokia
      cn=tu1,ou=people,o=nokia
  ```

```
cn=tu2,ou=people,o=nokia
cn=tu3,ou=people,o=nokia
cn=tu4,ou=people,o=nokia
cn=tu5,ou=people,o=nokia
cn=tu6,ou=people,o=nokia
```

This example shows the detail of an LDAP entry:

```
# ./authtool ldaptree url=ldap://10.44.3.73 entry=tu1
cn=tu1,ou=people,o=nokia
    userPassword: [B@5e179a
    objectClass: person
    nokiaMemberOf: i_ext_s_axs_tool_user
    sn: u1
    cn: tu1
```

- encrypt

  Encrypts the LDAP administrator's password.

  `authtool encrypt [user=username] [password=password]`

autoDiscovery

Location	*entuity_home*\bin
Type	Process
Invoked By	Application Server (tomcat)
User Invocation	Command line, web UI
Invoked Processes	proliferate
Configured Through	autodisc.cfg, entuity.cfg, command line
Log File	*entuity_home*\log\autoDiscovery.log [1..4]

Description

Entuity strongly recommend you configure how Entuity discovers network objects using the administration Inventory page, available through the web UI. This section details the options available when you decide to configure autoDiscovery through configuration files, or run it from the command line. Consult with your Entuity contact before configuring autoDiscovery through configuration files, or running it from the command line.

autoDiscovery can only be configured and run once. If, for example, you configure and run autoDiscovery from the web UI and then attempt to run it from the command line Entuity reports autoDiscovery is already running and does not start a second instance.

Through `provost.conf` you can configure when `autoDiscovery` runs, for example so each Sunday at 01:00 `provost` runs `autoDiscovery`. When `autoDiscovery` starts, and every subsequent minute whilst it is running, it checks the value of *automatic* in the autoDiscovery section of `entuity.cfg`. When it is set to:

- 0, `autoDiscovery` is not automatically started. When it is already running having been:
  - manually started, then this setting is ignored.
  - automatically started, then `autoDiscovery` is stopped.
- 1, `autoDiscovery` runs. It finds devices on a network, by 'pinging' every IP address on each specified network.

The `autoDiscovery` utility finds devices on a network, by 'pinging' every IP address on each specified network, and finding further subnets using SNMPv1/v2c and SNMPv3.

By default autoDiscovery:

- Does not search new subnets unless you use **-follow**.
- does not search the local subnet when you include addresses. To search the local subnet use **-local**.
- Command line values take precedence over any configuration file values, apart from when including addresses, excluding addresses and specifying port and community strings where the values are combined.
- Generates output to the `dev.txt` file. To specify a different name, use a parameter of **–o** <filename>. (See *Chapter 3 - Entuity System Files*.)

  The device file is written in a format that can be used directly by `proliferate`, consisting of lines of IP addresses followed by community strings (the file also contains comments, beginning with the '*#*' character).

`autoDiscovery` calls `proliferate` which by default adds all SNMP pollable devices to the candidate devices list in Entuity. Devices of a type Entuity:

- Recognizes are added to the candidate devices as devices of that type.
- Can generate an uncertified vendor file, are added to the candidate devices list as Unclassified.
- Does not recognize are added to the candidate devices list as devices without type.

Usage, Syntax and Options

Usage 1

This usage is only available with SNMPv1/v2c devices:

```
autoDiscovery [ -follow ] [ options... ]
```

Search only the network(s) to which the current host is attached.

In this usage autoDiscovery is run by itself. autoDiscovery only includes the local subnet to which the host is attached in the search. Any new subnets that are discovered are not followed.

The -follow option enables following of new subnets.

Usage 2

This usage is only available with SNMPv1/v2c devices:

```
autoDiscovery -in addresses [ -ex addresses ] [ -local ] [-follow ] [
options... ]
```

Search only the hosts or networks specified.

- -in <addresses>

  Comma separated list of hosts or networks to include in the search.

- -ex <addresses>

  Comma separated list of hosts or networks to exclude from the search.

- -local

  Search the networks to which the current host is attached.

- -follow

  Automatically search new network(s) that are discovered.

In this usage `autoDiscovery` is run with a list of addresses to include in, or exclude from, the search. When you want to:

- Search the network to which the current host is attached either include it in the list or use the -local option.
- Follow new subnets the -follow option must be given.

Usage 3

This usage is available with SNMPv1/v2c and SNMPv3 devices:

```
autoDiscovery -config [ file ] [ options... ]
```

Read options from configuration file.

- -config

  `autoDiscovery` can read options and data, such as included addresses, from a configuration file. When a file is not specified then autoDiscovery looks for the default configuration file, *entuity_home*`\etc\autodisc.cfg`.

  It is preferable to specify all of the required options in the configuration file, although you can also use the command line. When options have already been specified in the configuration file, the command line options usually take precedence. The exceptions are include addresses, exclude addresses, ports and community strings where configuration file and command line values are combined.

  The scope of `autoDiscovery's` search is therefore derived from a combination of command line, configuration file and default values, for example:

  - If you do not include addresses, `autoDiscovery` takes the host's subnet as the scope. If you have included addresses but also want to search the host's subnet then use -local. Alternatively, you can give the local network as an included address.
  - If you include addresses through the configuration file and command line,

`autoDiscovery` takes the combined address list as its scope. Similarly, if you exclude addresses through the configuration file and command line `autoDiscovery` takes the combined list and excludes it from the scope.

- If you include and exclude the same port, then `autoDiscovery` excludes the port from the search.
- If you want `autoDiscovery` to follow subnets it discovers then it must be configured with -follow.

Syntax Options

- -addpingonly

  Instructs `autoDiscovery` to set management level for the device to Ping Only when a device only responds to ping. When autoDiscovery -nodb is set, this option is ignored.

- -auto

  Instructs `autoDiscovery` to check the value of the *automatic* variable in the autodiscovery section of `entuity.cfg`. When this value is:

  - 0, autoDiscovery is stopped and does not run automatically.
  - 1, autoDiscovery runs each Sunday at 01:00 hours.

  During the configuration of Entuity if you created your device file using autoDiscovery, then autoDiscovery is running using -auto. By default *automatic* is set to 1, so autoDiscovery will automatically run every Sunday (for details on *automatic* see `entuity.cfg`).

- -c <string>

  Where <string> is comma separated list of community strings, no white spaces between, to be tried when SNMP data is requested. The default - “public” - should be included in the list if it is required. If the **-c** parameter is not specified, “public” is used.

- -dontallowipchange

  Instructs autoDiscovery to use the first discovered IP address on a device as its management address.

- -excludesysoids=<sysoid>

  Excludes the detailed sysoid from AutoDiscovery. This example excludes Cisco Unified Communications Manager from AutoDiscovery:

  ```
  -excludesysoids=1.3.6.1.4.1.311.1.1.3.1.2
  ```

- -f <n>

  This sets the SNMP final wait period, the period autoDiscovery waits to capture responses from final requests. When SNMP responses are slow or you are using more threads you may need to increase this final wait period.

  The SNMP final wait is linked to the Ping response time (-pt). The default of 30 seconds is 10 times the Ping response time. If you amend the Ping response time you may want to maintain this 10:1 relationship.

To change the final wait period, enter the new value in seconds.

- -h and -?

  Both open the help file, supplying an up-to-date list of commands and associated descriptions.

- -hn

  Do not resolve discovered IP addresses to host names. The default is on.

- -i

  Instructs autoDiscovery to mark all interfaces on discovered devices as unmanaged.

- -ith <n>

  Determines the number of addresses autoDiscovery can ping simultaneously, by setting the number of threads on the IP address queue. The default is 512 (see *Setting the Number of Threads*).

- -m

  Instructs autoDiscovery to mark only management interfaces on discovered devices as managed.

- -ma <n>

  Sets the largest allowed subnet size that is included in the autoDiscovery search, e.g. -ma 16 excludes from the search subnets that have more than 16 addresses. The default is unlimited, therefore all classes of subnets are fully pingable.

- -nodb

  Do not automatically populate the database.

- -o <filename>

  Name of device output file (default is `dev.txt`).

- -p <ports>

  Where <ports> is a comma separated list of ports, no white spaces between, to be tried when SNMP data is requested.

- -progress

  Includes progress details to standard out.

- -prune

  autoDiscovery discards networks if it receives a Network Unreachable response for the address or a subnet within it.

  As prune causes autoDiscovery to discard networks you must be careful that you specify the search address(es) at an appropriate level. If you specify a network address that has a number of subnets, it only requires one of those subnets to be unreachable for autoDiscovery to regard that whole network address being unreachable. autoDiscovery then stops searching the specified network address (possibly missing reachable subnets) and moves to the next specified address.

For example, this network list is suitable for -prune:

```
212.15.70.0
212.15.71.0
212.15.72.0
204.4.143.0
```

These are Class C subnets which do not contain subnets. If one of these networks is unreachable, it is not searched, speeding up the autoDiscovery process. The unreachable subnet does not stop autoDiscovery searching the other two subnets.

In this network list the first address is not suitable for -prune:

```
212.15.0.0
204.4.143.0
```

It is a Class B subnet which, in this example, contains subnets `212.15.70.0`, `212.15.71.0`, and `212.15.72.0`. If a Class C subnet within the specified Class B subnet is unreachable (e.g. does not yet exist), autoDiscovery stops the discovery process on the entire Class B subnet, and if applicable searches the next specified address.

Continuing the example, if `212.15.70.0` is reached, but `212.15.71.0` is unreachable then autoDiscovery does not search for `212.15.72.0`. autoDiscovery searches the next specified address, `204.4.143.0`. The only data returned from `212.15.0.0` is from the first subnet, `212.15.70.0`.

- -pt <n>

  Set ping timeout to *n* seconds, the default is 3 seconds. You can:

  - Decrease the timeout period to speed up autoDiscovery. On a slow network you are increasing the probability of not including every single device.
  - Increase the timeout period to improve the reliability of autoDiscovery results. On a slow network this increases the length of time it takes autoDiscovery to run.

- -rememberendhosts

  Maintain a list of all IP addresses, even those that are only able to respond to ping. This is a resource intensive setting.

- -sth <n>

  Determines the number of simultaneous autoDiscovery SNMP requests by setting the number of threads on the SNMP queue. The default is 64 (see *Setting the Number of Threads*).

- -usestdout

  Sets autoDiscovery output to standard out (i.e. the console) rather than the output file.

- -v

  Verbose mode, where detailed diagnostic information is produced and written to the log file, `autodiscovery.log`.

See Also

`proliferate`, `showdevs` and `prolifsys`.

Setting the Timeout Parameter

The ping timeout defaults to 3 seconds, but can be modified using the parameter **–pt <n>**.

The SNMP timeout varies with the 'ping' response time, and so you do not need to specify the SNMP timeout on the command line.

You can speed up `autoDiscovery` by reducing the ping timeout, but risk the possibility on a slow network of not discovering every single device. You can increase confidence in the reliability of the results by increasing the ping timeout.

To change the final wait period, use **–f <n>**. This defaults to 30 to allow for worst case scenario SNMP timeout.

Setting the Number of Threads

You can speed up `autoDiscovery` by increasing the number of threads it uses, as most time is spent waiting for 'ping' responses. However, more threads cost more system resources – and there is no upper limit currently set in `autoDiscovery`. This means that setting the number of threads is an 'advanced' option.

To set the number of threads on the IP address queue, use **–ith <n>**. The default is 512 threads.

To set the number of threads on the SNMP queue, use **–sth <n>**. The default is currently 64 threads. Increasing the *n* argument has a less far-reaching effect than would be the case with **-ith**, as far fewer devices get to the SNMP stage.

Thread specifications that are set on the command line override any that are set in the configuration file.

Writing a Configuration File

Allowed section headings in a configuration file are:

```
[ports]
[community strings]
[included addresses]
[excluded addresses]
[options]
```

An example configuration file:

```
[ports]
161
162
```

```
[community strings]
public
[included addresses]
137.73.8.10/255.255.255.0
slinky.cs.nyu.edu
[options]
-ith=64
-sth=32
-follow
-local
-nodb
```

When a configuration file:

- Does not contain a section of included addresses then the subnet to which the host is currently attached is searched.
- Does not contain a section of ports then the default port 161 is used.
- Does not contain a section of community strings then the default string "public" is used.
- Does not contain a particular option, then default values are used. For example, by default autoDiscovery does not search discovered subnets. Set the option -follow to allow autoDiscovery to search discovered subnets.

Specifying IP Addresses

autoDiscovery takes the IP address and subnet mask of the local machine. You can specify other machines or networks if required.

The format for specifying hosts and subnets is:

{ a[.b[.c[.d]]][/e.f.g.h] | hostname }

where each letter a..h is a number between 0 and 255 decimal inclusive.

IP addresses may be partial, and can optionally be followed by a slash and a subnet mask on the same line. In these cases a subnet is specified. A host can also be a machine name.

Examples are:

- 204.4.143.147 (a machine)
- hurricane (machine)
- 204.4.143 (a subnet)
- 204.4.143.0 (a subnet)
- 204.4.143.147/255.255.255.0 (a subnet).

autoDiscovery is currently sensitive (negatively) to white space in these files.

If you specify a big subnet, or if one turns up during the search, the number of potential addresses is checked against the maximum allowed. The default is not specified, so all sizes of subnets are allowed. You can change this using -ma to reduce the size of subnets that autoDiscovery is allowed to search.

Files

For SNMPv1/v2c and SNMPv3 devices `autoDiscovery` configuration is defined through *entuity_home*`/etc/autodisc.cfg`. In addition you can also configure discovery of SNMPv1/v2c devices from the command line. Where a device supports both SNMPv1/v2c and SNMPv3 credentials Entuity uses SNMPv3.

Discovered devices are added to Entuity and to the device file, by default `dev.txt`.

backup

Location	*entuity_home*`\bin`
Type	Process, by default runs each evening at 23:00
Invoked By	`provost`
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	*entuity_home*`\log\backup.log.[1..4]`

Usage

By default, backup is run automatically by `provost` every evening at 23:00.

You can also run backup from the command line:

```
backup
```

When run manually, then you need to ensure that the Entuity database server is running.

Description

The `backup` utility dumps the:

- **DSALPHA** database to *entuity_home*`\database\data\backupsw`
- **EOSdb** database to *entuity_home*`\database\data\backupdb`
- **GreenIT** database to *entuity_home*`\database\data\GreenIT`
- **secdb** database to *entuity_home*`\database\data\backupsecdb`
- **MySQL** users table to *entuity_home*`\database\data\backupmysql`.

The databases are not backed up individually.

The contents of DSALPHA, EOSdb, secdb and MySQL are dumped automatically. The only exception from EOSdstream is the dsutilization table that contains fast port data. The table structure is backed up but its contents are not currently included in the backup. With regard to EOStrend, all tables that have not been backed up before, or are empty, are backed up, together with all the data contained in existing tables that is more recent than the data in previous backups.

The number of tables (if any) to be backed up is output to the screen, together with the confirmation as to whether or not the backup has been successful.

If you need to restore the databases from a backup, use the restore command.restore both restores the databases and also repairs any errors.

By default this `backup` is run every evening at 23:00 by `provost`. You can also run it from the command line.

See Also

`restore`

cfgdigest

Location	*entuity_home*\lib\tools
Type	Utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

`cfgdigest` is an investigative tool used with Entuity configuration files. It has two usages to:

- Present a configuration file in a standard structure.

  ```
  cfgdigest c:\entuity\etc\sw_cpu_times.cfg
  ```

- Compare two named configuration files.

  ```
  cfgdigest c:\entuity\etc\sw_cpu_times.cfg c:\entuity\etc\
  sw_cpu_times.cfg
  ```

changeState

Location	*entuity_home*\bin
Type	Internal process
Invoked By	`provost`

User Invocation	No
Invoked Processes	n/a
Configured Through	*entuity_home*\etc\provost.conf
Log File	n/a

Description

changeState updates a prole sequence number in the database after prole has run.

checkvcs

Location	*entuity_home*\bin
Type	Internal process
Invoked By	n/a
User Invocation	No
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

Internal use only. Used by the VCS integration to check status of the Entuity system.

checkLicense

Location	*entuity_home*\bin
Type	Utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Syntax

From the command line of the Entuity server machine for which the license was generated you can use checkLicense to check the state of the license. It must always be run with one or more parameters, otherwise it may return an error.

You should always specify the license file, for example when running checkLicense from *entuity_home*|bin of the Entuity server machine use this structure:

```
checkLicense -f c:\Entuity\etc\license.dat
```

When you want to run `checkLicense` on a different machine to the one on which the license is to be installed, then you must define additional parameters, e.g. the operating system to which the Entuity server is installed, its IP address, host identifier, MAC address.

For example this allows you to check from Windows a license generated for an Entuity server installed to a Unix server:

```
checkLicense -s -i 10.0.0.1 -f c:\license\license.dat
```

These options are available with checkLicense:

- -f, indicates the name and, optionally, the location of the license file.
- -h <*host-indent*>, to be used when checking a license intended for an Entuity server installed to Windows, Linux or VMware environments. In those environments the host identifier is an integral part of Entuity licensing.

Options -l, -s, -v and -w should not be used with -h.

- -i <*ipaddress*>, to be used when checking a license intended for an Entuity server installed to Unix environments. In these environments the host machine's IP address is a key part of Entuity licensing.
- -m <*macaddress*>, indicates the host's MAC address is a key part of Entuity licensing. This is reserved for possible future usage.
- -l, indicates that the license you are testing is for an Entuity server installed in a Linux environment (and need only be used when running checkLicense in a non-Linux environment).
- -s, indicates that the license you are testing is for an Entuity server installed in a Unix environment (and need only be used when running checkLicense in a non-Unix environment).
- -w, indicates that the license you are testing is for an Entuity server installed in a Windows environment (and need only be used when running checkLicense in a non-Windows environment).
- -v, indicates that the license you are testing is for an Entuity server installed in a VMware environment (and need only be used when running checkLicense in a non-VMware environment).
- -d <*install-date*>, use to specify the date and time of the Entuity installation.
- -k, identifies the license file as one generated in an obsolete format. This is usually not applicable in live systems.

Description

This utility checks the validity of the license file, by default `license.dat`, against the license server, decoding the contents of the license file and writing them to file stdout.

This is an extract of example output, with `checkLicense` ran from `C:\Entuity\bin`:

```
checkLicense -f c:\Entuity\etc\license.dat

PRODUCT EOSDevices
```

```
  Expiry 01/Oct/2012 01:00:00
  Count  1
  OPTION 'C' - 600

PRODUCT IFA
  Expiry 01/Oct/2012 01:00:00
  Count  1
  OPTION 'C' - 2

PRODUCT IFAPremium
  Expiry 01/Oct/2012 01:00:00
  Count  1

PRODUCT EOSsnews
  Expiry 01/Oct/2012 01:00:00
  Count  2
  OPTION 'S' - 9999

PRODUCT EOSprovost
  Expiry 01/Oct/2012 01:00:00
  Count  1

PRODUCT TopologyMap
  Expiry 01/Oct/2012 01:00:00
  Count  1

PRODUCT EOSrca
  Expiry 01/Oct/2012 01:00:00
  Count  1

PRODUCT ReportServer
  Expiry 01/Oct/2012 01:00:00
  Count  1

PRODUCT TrapIntegration
  Expiry 01/Oct/2012 01:00:00
  Count  1

PRODUCT EOSobject0
  Expiry 01/Oct/2012 01:00:00
  Count  1
```

```
    OPTION 'C' - 50000
    OPTION 'P' - CISCOErrorDisableObject:0
    OPTION 'P' - CheckpointModule:0
    OPTION 'P' - HostConnectionTopoNodeEx:0
    OPTION 'P' - HostConnection:0
    OPTION 'P' - HostConnectionTopoNode:0
    OPTION 'P' - VirtualCDROM:0
    OPTION 'P' - VirtualController:0
    OPTION 'P' - VirtualNIC:0
    OPTION 'P' - VirtualDisk:0
    OPTION 'P' - VirtualMachine:0
    OPTION 'P' - ChargeableHypervisor:0:1
    OPTION 'P' - HyperVisor:0
    OPTION 'P' - VirtualizationPlatformDevice:0
    OPTION 'P' - IPSLAUDPCreator:0
    OPTION 'P' - IPSLATCPCreator:0
    OPTION 'P' - IPSLAJitterVoIPCreator:0
    OPTION 'P' - IPSLAJitterCreator:0
    OPTION 'P' - IPSLAICMPEchoPoller:0
    OPTION 'P' - IPSLAHTTPRawCreator:0
    OPTION 'P' - IPSLAHTTPCreator:0
  :
  :
  :
  :
  PRODUCT EYEVersion
    Expiry 01/Oct/2012 01:00:00
    Count  1
    OPTION 'V' - EYE:12.5:Entuity_12.5
    OPTION 'H' - 6b22bdfcc9f3193d2de813ceff89a709
```

where:

- C is the total amount of credits that the license permits.
- P is the policy group. Each group has its own rating, when set to 0 the group objects do not cost a license object.
- S is the credit value of one switch.

Error Messages

When `checkLicense` returns expiry dates of 1969 or 1970 for each process, this indicates the license file is invalid. When the license file was valid but is now expired, `checkLicense` returns the correct expiry date.

WARNING: Hardware change detected, indicates a change in the hardware setup of the Entuity server since the license was installed, e.g. a change in MAC address.

Files

`entuity.cfg`, `license.dat`, `hostIdentifier.txt`.

configure

Location	*entuity_home*\`install`
Type	Command line utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	Command line output, *entuity_home*\`log\EYEConfigure.xxx.log`

Description

`configure` sets up the Entuity server, for example `configure`:

- Configures the Entuity software, for example:
  - Database settings.
  - Activates and deactivates modules.
  - Sets module parameters.
  - Security settings.
  - Adds and updates available reports.
- Sets the ports that Entuity uses, e.g. for Event Viewer, Entuity database.
- Sets up necessary services (in a Windows environment).
- Allows you to select the license file.

You can only run `configure` after `install` has successfully completed. In a Windows environment `configure` runs as a Java wizard or through the command line. In Linux environments only the command line option is available.

Following the initial configuration of Entuity, you can run `configure` as often as is required to apply customizations to your system, for example updates to site specific files. You can only run `configure` when the Entuity server is not running.

Syntax

```
configure [[text | gui] [showportwarning]] | [defaults] | [services] |
[serverid ...]
```

Where:

- `text` instructs `configure` to run through the command line which is the default on Linux machines, but not Windows.
- `gui` instructs `configure` to run through the Java wizard which is the default on Windows machines.
- `defaults` instructs `configure` to run from the command line using the responses made the last time `configure` ran or using the settings in a specified defaults file. (See *configure defaults <file>*.)
- `showportwarning` instructs `configure` when run from the command line to display warnings when a port you are assigning to an Entuity process is already assigned to another process.
- `services` instructs `configure` to run but to only update the Windows services so that they apply to the current installation; the current installation must have been previously fully configured. The services option may be useful in a test environment where you have a number of Entuity installations. It is not recommended for use on your live installation.
- `serverid` includes a series of functions for identifying and updating the Entuity server identifier. (See *configure serverid*.)

configure defaults <file>

`configure defaults` instructs `configure` to run from the command line using the responses made the last time `configure` ran. `configure defaults` is useful when you have to re-run `configure` and do not want to amend any of the options available through `configure`, for example you have:

- Applied one or more patches.
- Upgraded Entuity.
- Amended a setting in a configuration file (for which you must run `configure` to apply them).

`configure defaults` *`file`* instructs `configure` to run using the settings in a specified text file. This can be useful when configuring multiple Entuity servers with essentially the same setup. This example uses the `defaults.cfg` file from the specified path:

```
configure defaults D:/resources/defaults.cfg
```

The defaults file uses the same parameters as listed in `entuity.cfg`. `configure` uses the default values of any option specified.

This example `defaults.cfg` file sets the web port number and configures the Atrium integration (*entuity_home*`/etc/installed_modules.cfg` includes a list of module names that can be enabled):

```
[]
webportnum=81
```

```
[modules]
Atrium_Integration=1

[JustForTemplates]
AtriumProperties_eye.server=testVal1
AtriumProperties_dataexport.dbServer=testVal2
```

configure serverid

The server identifier is used within Entuity to uniquely identify a particular Entuity server, this is especially important:

- In multi-server environments where `serverid` distinguishes one server from another.
- Restoring a database from one server to another server.

`configure serverid` has the syntax:

```
configure serverid { list | { { update | update_full } <source> }
```

Where:

- `list`, lists all of the serverids known to the server, including associated remote servers. It is useful when checking the consistency of serverid throughout the installation, for example after a cloning of a device, or restoring a database to a different server. To list the `serverids` in the install, including any remote servers enter:

  ```
  configure serverid list
  ```

- `update`, updates from the specified <*source*> the files and database with `serverid`.
- `update_full`, updates from the specified <*source*> the files and database with `serverid` but also dashboards, user selections and reports.
- <*source*> identifies to which `serverid` the server should be set:
  - `new` generates a new unique server identifier for the Entuity install. `new` can be useful when Entuity was installed to a virtual machine which you have then cloned. As part of multi-server implementation it requires a unique `serverid` which you can fully assign to the cloned install, for example:

    ```
    configure serverid update_full new
    ```

  - `from_db` uses the unique server identifier in the Entuity database install and allows you to apply it across the Entuity install. `from_db` might be useful when the database is being restored to a new machine, for example the original machine has failed and you want to maintain the remote and central relationships established with the other Entuity servers. To set an Entuity install to use the `serverid` contained in the database enter:

    ```
    configure serverid update_full from_db
    ```

  - `from_file` uses the unique server identifier in *entuity_home*`\etc\serverid.xml` and allows you to apply it across the Entuity install. `from_file` can be useful when the database is being applied to a new machine, for example you want to use the

setup from an existing server, its views, server accounts, report definitions but want it to be a unique install. To set an Entuity install to use the `serverid` contained in the *entuity_home*`\etc\serverid.xml` enter:

```
configure serverid update_full from_file
```

- *<serverid>* which is the manually entered `serverid`, for example **9a55e715-3c18-4ef1-9cc9-f1b7f29ea576**.

Entuity does not update the server identifiers associated with any physical connections. These connections are invalid and should be removed.

See Also

`install`, `serverid.xml`

dbcheck

Location	*entuity_home*`\bin`
Type	Process, runs once when Entuity is started
Invoked By	`starteye, configure`
User Invocation	Command line
Invoked Processes	`myisamchk`
Configured Through	*entuity_home*`\startup_o/s.cfg`
Log File	*entuity_home*`\log\dbcheck.log.[1..4]`

`dbcheck` verifies the last shutdown of `mysqld` completed successfully. When the shutdown was not successful it initiates a full check and, if necessary repair, of all database tables. Depending up on the size of your database this may take a significant amount of time, and so delay the start of Entuity. You can view its progress through `dbcheck.log`.

`dbcheck` is also called when `configure` runs if there is an existing database but no `mysql.error.log` which is usually the case when running an Entuity upgrade. `dbcheck` runs in fast mode (`dbcheck -F`) although you can set it to run in a more extensive mode (`dbcheck -E`).

You can run `dbcheck` from the command line but you should not run it when the database is running. `dbcheck` determines the successful shutdown of `mysqld` by scanning *entuity_home*`\log\mysqld.error.log` file for these messages:

```
081215 19:14:19 [Note] C:\entuity_z\database\bin\mysqld-nt: ready for
connections.
```

```
081215 19:14:46 [Note] C:\entuity_z\database\bin\mysqld-nt: Normal
shutdown
```

```
081215 19:14:46 [Note] C:\entuity_z\database\bin\mysqld-nt: Shutdown
complete
```

```
071022 20:50:52 [ERROR] D:\Entuity\database\bin\mysqld-nt: Incorrect
key file for table '.\dsalpha\dss_switchsystemresources.MYI'; try to
repair it
```

When dbcheck detects an error, it invokes myisamchk to perform the table check and repair. A check and repair is also run when the previous run of mysqld contains an Incorrect Key file message.

Options

- -f, forces dbcheck to run without analyzing mysql.error.log for errors.
- -Q do not scan the database rows to check for incorrect links.
- -F, dbcheck checks only tables that were not properly closed. This is the default Repair option selected when re-running configure, for example during an Entuity upgrade.
- -C, dbcheck checks only tables that have been changed since the last check or that were not properly closed.
- -M, dbcheck scans rows to verify that deleted links are valid and calculates a key checksum for the rows and verifies this with a calculated checksum for the keys.
- -E, dbcheck runs a full key lookup for all keys for each row which ensures that the table is 100% consistent. This is an extended database check and, depending on the size of the database, may take a significant length of time.
- -h, dbcheck displays the help text.

Logs

Messages are written to dbcheck.log in *entuity_home*\log. Each time dbcheck starts it scans mysql.error.log and then records its actions in the log file, for example:

```
11/07/2014     13:59:37     INFO:     (DBCheck.cpp)Scanning
"C:\Entuity\log\mysqld.error.log" to check mysqld was correctly
shutdown

11/07/2014 13:59:37  INFO: (MysqlErrorLog.cpp)mysqld last shutdown
completed successfully: 141106 13:51:22

11/07/2014 13:59:37  INFO: (DBCheck.cpp)Check/Repair complete
```

The file automatically wraps to dbcheck.log.[1-4] when the log becomes full.

devDefunct

Location	*entuity_home*\lib\tools
Type	Process, runs once a day at 00:00
Invoked By	provost
User Invocation	Command line
Invoked Processes	n/a
Configured Through	entuity.cfg

Log File	*entuity_home*\log\devdefunct.log.[1..4]

Description

It is responsible for deleting devices from Entuity that have aged out and are therefore deemed defunct. By default an age out value is not set, so devices are not automatically removed from Entuity. Through the devDefunct section in *entuity.cfg* you can set an age out value.

deviceDelete

Location	*entuity_home*\lib\tools
Type	Utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Syntax

deviceDelete *deviceName*

Description

The `deviceDelete` utility can be used to delete individual devices by name.

The output upon successful deletion is in the following format:

```
INFO: Successfully deleted deviceName
```

devpoller

Location	*entuity_home*\bin
Type	Process, run when devices are added to Entuity
Invoked By	n/a
User Invocation	Command line
Invoked Processes	`prolifsys`, `macman`, `ipman`, `vipman`, `nicman`
Configured Through	`entuity.cfg`
Log File	*entuity_home*\log\devpoller.log.[1..4]

Description

This process is run when devices are added to Entuity, calling the processes that identify device details.

devsysman

Location	*entuity_home*\bin
Type	Process, run once daily at 04:30
Invoked By	provost
User Invocation	Command line
Invoked Processes	prolifsys, macman, ipman, vipman, nicman
Configured Through	n/a
Log File	*entuity_home*\log\devsysman.log.[1..4]

Description

It is responsible for the SNMP polling of network devices for system-related information, including system location and description.

diskMonitor

Location	*entuity_home*\bin
Type	Process, runs continuously
Invoked By	starteye
User Invocation	Command line
Invoked Processes	prolifsys, macman, ipman, vipman, nicman
Configured Through	entuity.cfg
Log File	*entuity_home*\log\diskMonitor.log.[1..4]

Description

This process monitors disk space on the Entuity server and is invoked when Entuity starts up. diskMonitor polls for disk space where the Entuity database is installed. It compares this value to two thresholds, if it falls below the:

- First diskMonitor sends events to Event Viewer.
- Second diskMonitor initiates the shutdown of Entuity. This prevents corruption of the database that can occur when disk space is not available.

You can configure diskMonitor, e.g. set threshold values, period between samples, through entuity.cfg.

Logs

Messages are written to diskMonitor.log in *entuity_home*\log. Each time diskMonitor starts it writes to the log its current settings. Each time it analyzes a sample, it writes the results to the log. The file automatically wraps to diskMonitor.log.[1-4] when the log becomes full.

dnsproxy

Location	*entuity_home*\bin
Type	Process
Invoked By	provost
User Invocation	Command line
Invoked Processes	-
Configured Through	entuity.cfg
Log File	*entuity_home*\log\dnsproxy.log.[1..4]

Description

dnsproxy makes DNS requests and creates a DNS cache which is accessed by Entuity processes. The cache is limited to 10,000 per zone. This is configurable in *etc*/entuity.cfg. If the cache is exceeded Entuity drops the least recently used entries.

dnsproxy refreshes the cache:

- When the Entuity object inventory changes, e.g. a new device is managed, a device is added to a zone.
- Within 10 minutes, if a zone was unavailable.

Options

- *?*, displays help.
- *.exit*, quits the dnsproxy action.
- *.invalidate*, simulates a zone modification.
- *.list*, displays configured zones.
- *.port*, display dnsproxy port number.
- *.stats*, display dnsproxy statistics for the current zone.
- *.dump*, display dnsproxy contents for current zone.
- *.walk* <start> <cnt>, display dnsproxy contents for current zone from start limited to cnt items.
- *.purge*, discard dnsproxy contents for the current zone.
- *.zone* <zone>, set current zone by *id* or *name*.
- *.version* <4|6>, set ip version to 4 or 6.
- <*ipAddress*>, perform reverse dns lookup.
- <*host*>, perform dns look up.

domman

Location	*entuity_home*\bin
Type	Process, runs daily at 05:30

Invoked By	`provost`
User Invocation	Command line
Invoked Processes	`prolifsys`, `macman`, `ipman`, `vipman`, `nicman`
Configured Through	`entuity.cfg`
Log File	*entuity_home*`\log\domman.log.[1..4]`

Description

It is responsible for maintaining the system domain tables, including device and VLAN domains.

DsKernelStatic

Location	*entuity_home*`\bin`
Type	Process, runs continuously
Invoked By	`starteye`
User Invocation	n/a
Invoked Processes	StormWorks functionality
Configured Through	`sw_`*Name*`.cfg`. `startup_O/S.cfg`
Log File	*entuity_home*`\log\dskernel.log.[1..4]`

Description

This process actions activities for which it has been configured through `StormWorks` configure.

dumpipnettoport

Location	*entuity_home*`\lib\tools`
Type	Utility
Invoked By	Command line
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

`dumpipnettoport` exports the ipnettoport table to the command line. ipnettoport maps IP addresses to device ports.

dumpiptodev

Location	*entuity_home*\lib\tools
Type	Utility
Invoked By	Command line
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

`dumpipnettodev` exports the `ipnettodev` table to the command line. `ipnettodev` maps IP addresses to devices.

dumpvip

Location	*entuity_home*\lib\tools
Type	Utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

`dumpvip` exports virtual IP addresses to the command line.

duplexman

Location	*entuity_home*\bin
Type	Process, runs daily at 07:00
Invoked By	`provost`
User Invocation	n/a
Invoked Processes	*n/a*
Configured Through	`entuity.cfg`
Log File	*entuity_home*\log\duplexman.log.[1..4]

Description

It is responsible for maintaining the port duplex tables, so Entuity recognizes whether each managed port is full or half duplex.

encode_keychange

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

encode_keychange -t md5|sha1 [OPTIONS]

Description

encode_keychange produces a KeyChange string using the old and new passphrases as described in Section 5 of RFC 2274 "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)". -t option is mandatory and specifies the hash transform type to use.

The transform is used to convert passphrase to master key for a given user (Ku), convert master key to the localized key (Kul), and to hash the old Kul with the random bits.

Passphrases are obtained by examining a number of sources until success (in order listed):

- Command line options (see -N and -O options below);
- The file $HOME\.snmp\passphrase.ek which should only contain two lines with old and new passphrase;
- Standard input -or- user input from the terminal.

Options

- -E [0x]<engineID> EngineID used for Kul generation.

  <engineID> is interpreted as a hex string when preceded by 0x, otherwise it is treated as a text string. If no <engineID> is specified, it is constructed from the first IP address for the local host.

- -f, force passphrases to be read from standard input.

- -h, display the help message.
- -N "<new_passphrase>" , passphrase used to generate the new Ku.
- -O "<old_passphrase>" , passphrase used to generate the old Ku.
- -P, turn off the prompt for passphrases when getting data from standard input.
- -v, be verbose.
- -V, echo passphrases to terminal.

eyepoller

Location	*entuity_home*\bin
Type	Utility
Invoked By	starteye
User Invocation	n/a
Invoked Processes	n/a
Configured Through	*entuity_home*/etc/startup_o/s.cfg *entuity_home*/etc/eyepoller_overrides.cfg
Log File	*entuity_home*/log/eyepoller.log

Description

By default eyepoller polls for interface utilization, fault and congestion data at five minute intervals. eyepoller does not poll ports that adminstratively down.

eyepoller is configurable through entuity.cfg, as are associated events which monitor the accuracy of polling. These events are not enabled by default.

RFC 2863 requires interfaces that operate above 20 Mbps to support 64 bit counters; SNMP agents that support 64 bit counters are available from SNMPv2 onwards. However, eyepoller can successfully poll ports with a speed of 105Mbps or below using SNMPv1 polling of 32 bit counters. For eyepoller to collect traffic and utilization data for ports with a speed above 105Mbps there must be accompanying 64 bit counter support in the device's SNMP agent.

Entuity recommend checking devices for installation of SNMP agents that support 64 bit counters. For example you can test a device's 64 bit counter support using *entuity_home*\lib\tools\snmpwalk:

```
snmpwalk –v2c -c <community> <device> .1.3.6.1.2.1.31.1.1.1.6
```

eyepoller uses a number of 64 bit counters including IF-MIB::ifHCInOctets.

Where the device agent does not support 64 bit counters you should consider upgrading the agent.

FixNewBinVendor

Location	*entuity_home*\lib\tools
Type	Utility
Invoked By	Command line
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

Prior to Entuity's introduction of generically managed devices Entuity would, where possible, automatically generate device definitions and assign device types to devices for which it did not contain vendor definition details.

`FixNewBinVendor` allows you amend `attr.cfg`, so existing devices that are managed through `newbin.vendor` use the generic device type, rather than switch or router. This utility only requires running once.

flowCollector.bat

Location	*entuity_home*\bin
Type	Utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

With Integrated Flow Analyzer you should amend port to application mapping through the web UI. These mappings are held in *entuity_home*\etc\flow-applications-template.txt, which you can directly amend. `flowCollector.bat` is a batch file to be run when you have edited the template file.

To load the mappings, from *entuity_home*\bin enter:

```
flowCollector -loadMappings
```

You can also run the batch file to check the status of current flow collection, from *entuity_home*\bin enter:

```
flowCollector -stats
```

`flowCollector` displays a report to the screen, for example:

```
Flow Collector Started: Wed Aug 11 13:58:58 BST 2010
Receiver on port 9996 (receive buffer size = 8192 b)
  Accepted packets: 17331, bytes: 7071048
Packet queue usage: 0 from available 1000 (peak usage: 1)
  Number of accepted packets: 17331
  Number of dropped packets:  0
Packet processor (checking packet sequences: yes)
  Unrecognized packets:       0
  Missed packets:             0
  Total packets:              17331
  Total flow records decoded: 138648
Flow Buffer
  Number of flows dropped due to flush partition busy: 0
  Accepted 138648 flows from a total of 138648
  NetFlowV9 unprocessed flows:
    option flow sets:               0
    data sets due to no template:   0
    flows due to IPv6:              0
    flows due to insufficient data: 0
  Recent partition stats:
    flows received: 80, dropped: 0, grouped: 72
    flows received: 80, dropped: 0, grouped: 72
    flows received: 80, dropped: 0, grouped: 72
    flows received: 80, dropped: 0, grouped: 72
    flows received: 80, dropped: 0, grouped: 72
Flow Buffer Flusher recent writes:
    0 ms for 8 records
    0 ms for 8 records
    0 ms for 8 records
    0 ms for 8 records
    0 ms for 8 records
Performance Sampler
  Recent write times for Interface
    16 ms for 2 records
  Recent write times for Device
    0 ms for 1 records
```

```
  Recent write times for Performance
    0 ms for 23 records
Flow Filter
  perform inventory filtering: yes
  in-memory version: Thu Jul 15 13:46:01 BST 2010
  exclusion rules: 0
Application port mapper
  in-memory version: Fri Jul 09 10:43:59 BST 2010
NetFlow v9 Store
  number of templates: 0
Age Out Job recent deletes:
    0 ms for 26 records
    0 ms for 0 records
    0 ms for 8 records
    0 ms for 26 records
    16 ms for 0 records
    0 ms for 8 records
    0 ms for 26 records
    0 ms for 0 records
    0 ms for 8 records
    0 ms for 26 records
```

getDownstream

Location	*entuity_home*\lib\tools
Type	Utility
Invoked By	n/a
User Invocation	Command line, Extensible Menu
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Syntax

Entuity identifies managed objects using two different methods, each method assigns objects their own unique identifiers. These identifiers are normally only used by internal Entuity processes. However you can access these identifiers:

- ForkEvent forwards *objectID* and *objectType* as part of *eosObjectID*.

- Entuity Remedy AR System integration module uses ForkEvent to forward *objectID*, *objectType* and `StormWorks`*ID* as part of *eosObjectID*.
- Flex Reports allow you to report on `StormWorks` identifiers when you select *Show Hidden Data*.

Where you are receiving the identifiers through ForkEvent, you should use `getDownstream` with:

```
getDownStream.exe objectID objectType
```

where:

- *objectID* is the unique identifier for that managed object.
- *ObjectType* is 0 for device, and 1 for port.

Where you identify the object through running a Flex Report to find the `StormWorks` identifier use this syntax:

```
getDownStream StormWorksID
```

Description

Network Outage events indicate the number of devices impacted by a node failure. From the command line you can run `getDownstream` to view a list of the devices impacted by the failure. `getDownstream` shows the devices Availability Monitor identified as being impacted by the node failure the last time Availability Monitor ran.

`getDownstream` can also be called from a context sensitive User Action in Event Viewer.

By default availability monitor polls every two minutes. It is possible that when you run `getDownstream` to investigate an event raised in Event Viewer, Availability monitor will have already run again. Occasionally a change in the node status or network may result in `getDownstream` correctly returning a different number of impacted devices to that identified in the original event.

Files

`entuity.cfg`, `license.dat` (See *Chapter 3 - Entuity System Files*.)

hostIdent

Location	*entuity_home*\`bin`
Type	Process, runs during `Install`
Invoked By	`Install`
User Invocation	Command Line
Invoked Processes	n/a
Configured Through	n/a
Log File	*entuity_home*\`etc\hostIdentifier.txt`

Description

The license file restricts installation of Entuity to the server for which you provided a host identifier.

You must provide to your Entuity supplier the host identifier of the machine to which you want to install Entuity. You can discover this by running `hostident`:

- Before installation, by obtaining a copy of `hostident` from your Entuity contact, and running it from the command line. `hostident` displays the host identifier on the command line.
- As part of `install`, `install` displays the host identifier.
- As part of `configure`, `configure` displays the host identifier.

You can run `hostident` from the command line:

```
hostIdent
```

httpd

Location	*entuity_home*`\lib\apache\bin`
Type	Process, runs continuously
Invoked By	
User Invocation	n/a
Invoked Processes	n/a
Configured Through	`entuity.cfg, httpd_eye.conf`
Log File	*entuity_home*`\log\http.error_log, http.access_log`

Description

This process is a web server for the GUI front end. It is started and stopped automatically. The web server used is the public domain Apache web server. For details on the error and access log messages created, refer to the Apache documentation at http://www.apache.org .

install

Location	On the supplied software image
Type	Command line utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	Command line output, *entuity_home*`\log\EYEInstall.log`

Description

`install` installs the Entuity software to your server. It is the first step in setting up your server, and must successfully complete before you can configure it.

Through `install` you can specify the folders Entuity uses to build the database and locate the log files. On completion it identifies the current license file and the host identifier.

Syntax

```
install [text ¦ gui] ¦ [no-configure] ¦ [to <path>]
```

Where:

- **text** instructs `install` to run through the command line which is the default on Linux machines, but not Windows. In Windows `install` runs as a Java wizard.
- **gui** instructs `install` to run through the Java wizard which is the default on Windows machines, but not on Linux. In Linux `install` runs through the command line.
- **no-configure** instructs `install` to not trigger the running of configure.
- **to <path>** allows entry of the directory to which to install Entuity. `install` prevents installation to a system directory.

See Also

`configure`

instService

Location	*entuity_home*\bin
Type	Process, run during install
Invoked By	`configure`
User Invocation	n/a
Invoked Processes	n/a
Configured Through	`configure`
Log File	n/a

Description

`instService` creates the Entuity Windows services during Entuity server installation.

ipman

Location	*entuity_home*\bin
Type	Process, run each day at 05:00, 10:000 and 15:00
Invoked By	`provost`
User Invocation	n/a
Invoked Processes	n/a

Configured Through	`provost.conf`, `entuity.cfg`
Log File	*entuity_home*`\log\ipman.log.[1..4]`

Syntax

`ipman` usage *options*

```
ipman <ttl> [deviceName] [-d] [-f] ¦ ipman -h
```

where:

- *ttl*, sets the number of iterations of `ipman` before a MAC to IP address mapping expires.
- *deviceName*, device to poll. When not specified `ipman` polls all managed devices.
- **-d**, sets the debug level.
- **-f**, instructs `ipman` to check `ipman.devicefile` in `entuity.cfg` for a device file. In this device file you can specify routers which Entuity does not manage but from which you want to collect ARP cache information. Entuity requires ARP cache details for connected end host IP address identification.

  By default `provost` runs `ipman` with **-f** but does not reference, or require, a device file. ipman.log includes an information message reporting a device file is not specified:

  ```
  INFO: Unable to open a device file: please set ipman.devicefile in
  entuity.cfg to the full path and name of your device file.
  ```
- **-h**, calls help when used from the command line.

Description

`ipman` uses SNMP to gather ARP (Address Resolution Protocol) entries from devices. `ipman`, by default, gathers ARP information from each of the devices being managed by Entuity, checking ARP cache entries for switch and router capabilities.

You can configure `ipman` to gather ARP data from devices not managed by the Entuity server by running `ipman` against a specific device, or a list of devices specified in a configuration file.

`ipman` ignores MAC addresses in the range 00:00:0C:07:AC:00 to 00:00:0C:07:AC:FF, this range can be extended through the macman section in `entuity.cfg`.

When zones are configured `ipman` places local arp entries in the appropriate zone according to the interface on which they were seen.

Example ARP Cache Collection

In multi-server environments an Entuity server may not manage routers from which it requires ARP cache information to perform end host IP address resolution on devices it does manage. These routers may be managed by other Entuity servers. Rather than have more than one Entuity server manage the same routers, through a device file you can configure `ipman` to collect ARP cache information from these routers.

By default `provost` runs `ipman` with **-f**, but does not reference a device file. You must create a device file and through entuity.cfg identify it to `ipman`. `ipman` can then collect ARP cache information from the routers specified in the device file.

To set `ipman` to collect ARP cache information from routers an Entuity server does not manage:

1) Create a tab delimited text file containing the host names or IP addresses, and SNMP read community strings for the routers `ipman` polls.

   For example the file *entuity_home*`\etc\arp_cache_devices.cfg` contains:

   ```
   router1_hostname community_xxx
   router2_hostname community_xxx
   router3_hostname community_xxx
   ```

Entuity recommend you use the example location and name of the device file to ensure it is maintained during Entuity upgrades.

2) In entuity.cfg specify the name of the device file, `D:\Entuity\etc\entuity.cfg`:

   ```
   [ipman]
   devicefile=D:\Entuity\etc\arp_cache_devices.cfg
   ```

3) The next time `ipman` runs it references the device file.

   You can check the success of the polling through `ipman.log`:

   ```
   INFO: Opened D:\Entuity\etc\arp_cache_devices.cfg
   INFO: Got arp info for device router1_hostname
   .
   .
   ```

kill

Location	*entuity_home*`\lib\tool`
Type	Command line
Invoked By	
User Invocation	n/a
Invoked Processes	n/a
Configured Through	
Log File	

Description

Terminates the process using its process identifier, for example to kill process number 9:

```
kill 9
```

licenseSrvr

Location	*entuity_home*\bin
Type	Process, runs continuously
Invoked By	starteye
User Invocation	n/a
Invoked Processes	n/a
Configured Through	license.dat, startup_O/S.cfg
Log File	*entuity_home*\log\license.log.[1..4]

Description

This process, together with DsKernelStatic, manages the Entuity licenses. It is started with the other main system processes. Before managing a new object Entuity checks that the license allows the object to be managed. Licensing information is read from the license each time the Entuity server starts. By default licensing information is read from file *entuity_home*\etc\license.dat.

macman

Location	*entuity_home*\bin
Type	Process, runs daily at 09:30
Invoked By	provost, macScheduler
User Invocation	n/a
Invoked Processes	n/a
Configured Through	
Log File	*entuity_home*\log\macman.log.[1..4]

Description

macman gathers MAC (Media Access Control) information for the devices Entuity manages. This allows Entuity to display port end hosts.

macman ignores MAC addresses in the range 00:00:0C:07:AC:00 to 00:00:0C:07:AC:FF, these are reserved for ethernet and FFDI HSRP group virtual mac addresses. You can extend the MAC addresses macman ignores through the macman section in entuity.cfg.

macScheduler, also runs macman on devices when the port operational status of any monitored non-router port changes from inactive to active. This status change implies other changes have also occurred on the port and MAC addresses require checking.

Entuity checks for the port operational status every hour for non-router ports.

Entuity adds a five minute delay before running a MAC address check on a device, resulting from a port status change, in order to suppress many port changes occurring in a short space of time and flooding the server (and device) with requests. When port status changes occur on many devices in a short period of time (e.g. at the beginning of the day when

everyone connects and logs on), then the MAC checks for some devices may be delayed further due to the load on the server.

Entuity may not report some MAC address changes. For example, if a MAC address is seen by a port (node added or transmits a packet), and then no longer seen (node removed or no longer transmitting), and the MAC address is aged out of the device's MAC table before Entuity has polled the table for changes.

Switch ports that have more than ten MAC addresses and also have associated VLANs are identified as trunk ports. Entuity does not display the end hosts of trunk ports.

MAC addresses are aged out of the database using a 'time to live' scheme whereby a MAC address is only discarded when it has not been seen anywhere in the network for seven days. However Entuity retains MAC address change history until the number of event changes reaches a set limit, at which point Entuity discards the oldest change history record.

macScheduler

Location	*entuity_home*\bin
Type	Process, runs daily at 09:30
Invoked By	Change in port status to active.
User Invocation	n/a
Invoked Processes	macman
Configured Through	startup_O/S.cfg
Log File	*entuity_home*\log\macScheduler.log.[1..4]

Description

This process runs macman on devices when the port operational status of any monitored non-router port changes from inactive to active. This status change implies other changes have also occurred on the port and MAC addresses require checking.

mapToView

Location	*entuity_home*\install
Type	Migration utility.
Invoked By	User
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

`mapToView` migrates maps developed in versions of Entuity prior to Entuity 16.0 to the map format introduced in Entuity 16.0. Map available for migration are stored by user name beneath *entuity_home*\`maps`. Before migrating to Entuity 16.0 you should have consulted the *Entuity Migration Guide*. and prepared the maps you wanted to migrate.

To view the command line help run `mapToView` enter:

```
maptoview --help
Usage: mapToView [OPTION].. [map]..

Where valid options are:
-h --help : Print this message
-u --user <OWNER> : Convert map(s) owned by OWNER
-c --createAs <USER> : Create as USER
-p --password <PASSWORD> : Password for USER
-s --shared : Convert shared map(s)
-a --all : Convert all maps owned by OWNER
-v --verbose : Print progress messages
```

Parameters		Description
`-a`	`--all`	Convert all maps owned by the specified owner (`--user`) or are public (`--shared`).
`-c`	`--createAs`	The username of the Entuity user assigned ownership of the converted map. This user account must be a member of the Administrator user group or have the Create Views tool permission.
`-h`	`--help`	Outputs to the command line a listing and brief description of the `mapToView` parameters.
`-p`	`--password`	Password of the Entuity user assigned ownership of the converted map. A password is only required if you are using LDAP to manage user accounts.
`-s`	`--shared`	Parameter used to identify maps that were shared, set to Public. They do not have an owner.
`-u`	`--user`	Converts map(s) owned by the user. If a user account name includes spaces then enclose that name in double quotes, e.g. `"James Smith"`.
`-v`	`--verbose`	Sends to the command line progress messages on map conversion.

Table 2 `mapToView` Parameters

myisamchk

Location *entuity_home*\`database\bin`

Type	Database utility.
Invoked By	User
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Documentation	*entuity_home*\database\docs\manual.htm
Log File	n/a

Description

myisamchk gets information about your database tables or checks, repairs, or optimizes them. myisamchk works with MyISAM tables (.MYD and .MYI suffixed files).

myisampack

Location	*entuity_home*\database\bin
Type	Database utility.
Invoked By	User
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Documentation	*entuity_home*\database\docs\manual.htm
Log File	n/a

Description

myisampack compresses MyISAM tables (.MYD and .MYI suffixed files), compressing each column in the table separately.

mysql

Location	*entuity_home*\database\bin
Type	Database utility.
Invoked By	User
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Documentation	*entuity_home*\database\docs\manual.htm
Log File	n/a

Description

`mysql` is a simple SQL shell (with GNU readline capabilities). It supports interactive and non-interactive use. When used interactively, query results are presented in an ASCII-table format. When used non-interactively (for example, as a filter), the result is presented in tab-separated format. The output format can be changed using command options.

mysqladmin

Location	*entuity_home*\database\bin
Type	Database utility.
Invoked By	User
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Documentation	*entuity_home*\database\docs\manual.htm
Log File	n/a

Description

`mysqladmin` is a client for performing administrative operations. You can use it to check the server's configuration and current status, to create and drop databases, and more.

mysqlcheck

Location	*entuity_home*\database\bin
Type	Database utility.
Invoked By	User
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Documentation	*entuity_home*\database\docs\manual.htm
Log File	n/a

Description

`mysqlcheck` client checks, repairs, optimizes, and analyzes tables.

mysqld

Location	*entuity_home*\database\bin
Type	Database utility.

Invoked By	starteye
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Documentation	*entuity_home*\database\docs\manual.htm
Log File	*entuity_home*\database\data\<hostname>.err

Description

This process is the database server. It listens on a single TCP/IP port number (default 3306) through which both the Entuity database can be accessed.

mysqldump

Location	*entuity_home*\database\bin
Type	Database utility.
Invoked By	User
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Documentation	*entuity_home*\database\docs\manual.htm
Log File	n/a

Description

`mysqldump` can be used to dump a database or a collection of databases for backup or transfer to another SQL server. The dump typically contains SQL statements to create the table, populate it, or both. However, `mysqldump` can also be used to generate files in CSV, other delimited text, or XML format.

mysqlimport

Location	*entuity_home*\database\bin
Type	Database utility.
Invoked By	User
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Documentation	*entuity_home*\database\docs\manual.htm
Log File	n/a

Description

`mysqlimport` is a data import utility providing a command-line interface to the LOAD DATA INFILE SQL statement.

mysqlshow

Location	*entuity_home*\database\bin
Type	Database utility.
Invoked By	User
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Documentation	*entuity_home*\database\docs\manual.htm
Log File	n/a

Description

`mysqlshow` allows you to view which databases exist, their tables, or a table's columns or indexes.

newcommunity

Location	*entuity_home*\lib\tools
Type	Database utility.
Invoked By	User
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	*entuity_home*\log\newcommunity.log.[1..4]

Syntax

`newcommunity` *old-community new-community*

Description

The `newcommunity` utility is used to change all instances of device SNMP read community string *old-community* to *new-community*.

Once the utility has completed its processing successfully, the following confirmation message is displayed:

```
Modified community strings of n devices
```

where *n* is the number of instances that were changed.

Files

`entuity.cfg` and `bin.vendor`

See Also

`showdevs`.

nicman

Location	*entuity_home*\bin
Type	Process, run daily at 21:00
Invoked By	`provost`
User Invocation	Command line
Invoked Processes	n/a
Configured Through	`provost.conf`
Log File	*entuity_home*\log\nicman.log.[1..4]

This process combines end host MAC and IP address information held in the database, and stores it in a form suitable for use by other applications.

ObtainGenericVendor

Location	*entuity_home*\bin
Type	Process
Invoked By	-
User Invocation	Command line
Invoked Processes	n/a
Configured Through	-
Log File	*entuity_home*\log\ObtainGenericVendor.log.[1..4]

Entuity now takes under its management devices without a device support dataset (vendor file). Entuity first attempts to create a generic vendor file and if that fails devices are still polled.

```
ObtainGenericVendor -y
```

For example if Entuity is managing a device with the unsupported sysoid sysoid: .1.3.6.1.4.1.9694.1.4, `ObtainGenericVendor` would create a new device support dataset:

entuity_home\Entuity\etc\uncertified\1.3.6.1.4.1.9694.1.4.vendor

perror

Location	*entuity_home*\database\bin
Type	Database utility
Invoked By	User
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Documentation	*entuity_home*\database\docs\manual.htm
Log File	n/a

Description

`perror` prints a description for a system error code or for a storage engine (table handler) error code.

probity

Location	*entuity_home*\lib\tools
Type	Process, runs continuously
Invoked By	User
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Syntax

```
probity
```

Description

`probity` displays information about the devices currently being monitored in the Entuity management environment. It is useful for checking the integrity of the database, and can be used to troubleshoot system problems.

An example of the output produced is shown below:

```
1 routerb2 Attr:1 Prole ID:5 RawData:12
2 routerc1 Attr:1 Prole ID:5 RawData:16
3 routerc2 Attr:1 Prole ID:4 RawData:16
```

One row of information appears for each device being managed. The first column lists the device ID, and is used for internal purposes. The second column lists the device name, as defined by the System Administrator. The third column lists the number of entries this device has in the database ‘attributes’ table (this value should always be set to 1). The fourth column

displays the ID of the poller responsible for monitoring the device. If this value is set to 'INVALID', then Entuity is not polling the device, the most likely reason being that the poll time is too long. The fifth column displays the number of ports that are being monitored for the given device.

Files

`entuity.cfg`

See Also

`showdevs`

prodigy

Location	*entuity_home*\bin
Type	Process, runs on completion of `prole`
Invoked By	`provost`
User Invocation	n/a
Invoked Processes	n/a
Configured Through	`provost.conf`
Log File	*entuity_home*\log\prodigy.log.[1..4]

Description

`prodigy` is responsible for analyzing the polled data, forwarding information to the trend database for storage, and for removing any ports that are marked for deletion. It also checks there are enough license credits to manage all of the ports on the device.

profluent

Location	*entuity_home*\bin
Type	Process, runs once a day, at 04:00
Invoked By	`provost`
User Invocation	n/a
Invoked Processes	n/a
Configured Through	`provost.conf`
Log File	*entuity_home*\log\prof.log.[1..4]

Description

This process manages the relationship between the `prole` process(es) and network devices. The `profluent` process calculates the number of `proles` that need to be run, based on the variances between typical device polling times.

prole

Location	*entuity_home*\bin
Type	Process, runs every 20 minutes
Invoked By	provost
User Invocation	n/a
Invoked Processes	setupProle
Configured Through	provost.conf
Log File	*entuity_home*\log\prole.log.[1..4]

Description

This process is responsible for SNMP polling a pre-defined list of networking devices. Multiple proles may be started simultaneously, depending on the number of devices being managed.

The SNMP response data is forwarded to the database for storage and subsequent analysis by prodigy. For hubs prole also artificially creates outbound data (octets/packets), which hubs do not provide.

prole uses vendor information, supplied through bin.vendor individual vendor files, and mib.txt.

proliferate

Location	*entuity_home*\bin
Type	Process
Invoked By	autoDiscovery
User Invocation	Command line, Extensible Menu
Invoked Processes	prolifsys and prolifmodule
Configured Through	device file, command line, startup_O/S.cfg
Log File	Output is usually to stdout, unless autoDiscovery is run from the command line with the appropriate settings then output is to *entuity_home*\log\proliferate.log.[1..4]

Description

proliferate compares the SNMP devices Entuity currently manages against those you specify it should be managing, attempting to add devices when found. When adding devices to Entuity proliferate:

- Attempts to identify the device type, first using individual vendor and then bin.vendor files.
- Identifies whether the device supports router, switch, router/switch or none of these system capabilities.

- Has an extensive set of switches that you can use to tailor its behavior for each device:
  - The communication protocol Entuity uses to manage a device, i.e. IPv4 (default), IPv6.
  - The Entuity device management level, e.g. **Full**, **Full (Mgmt Port Only)**, **Full Management (No Ports)**.

You can set the device(s) `proliferate` attempts to add to Entuity using:

- A device file, `proliferate` compares the devices detailed in the file to the devices Entuity currently manages. You can create your own device file, or use `autoDiscovery`. When `autoDiscovery` runs it creates a device file, `autodisc.cfg`, ready for `proliferate`.
- A single IP address and community string that `proliferate` compares to the devices Entuity currently manages.
- The options available through Inventory Administration.

So, before a device is added to Entuity, `proliferate` verifies that it:

- Has no existing interface IP addresses (if there are already any addresses for the device, then it is assumed to exist under a different name, unless the **-I** parameter is set).
- Is responding to SNMP requests.
- Passes a poll check.
- Is of a recognized device type for management (based on the device sysOID being included to individual vendor or `bin.vendor` files).

When a device:

- Passes all of the checks `proliferate` adds it to Entuity, with devices of a type listed in *entuity_home*`\etc\uncertified` being added as Unclassified devices.
- Fails any of the first three checks, then it is rejected.
- Only fails the final check, then `proliferate` adds the device to Entuity as an Unclassified device. From the web UI you can run an Extensible Menu function to manage the device, which runs `proliferate` with -g. Alternatively where you have a number of devices to add you may want to add them through a device file.

`proliferate` automatically runs `prolifsys` and `prolifmodule`.

Usage, Syntax and Options

Parameters

`-g` is a powerful, resource intensive option and should only be used when specifying an IP address or community string. Adding one device can take fifteen minutes; Entuity do not recommend you use it with large device files.

Switch Short / Long		Description
-a	--auth=	SNMPv3 specific parameter. Sets the authentication protocol, valid values are: ■ MD5 (Message-Digest algorithm 5). ■ SHA (Secure Hash Algorithm).
-A	--auth-pass=	SNMPv3 specific parameter. Sets the authentication password, valid values must be between eight and thirty-two characters long.
-c	--community=	Sets the device community string.
-d	--device=	Sets the device name or IP address Entuity uses when polling the device.
-D	--name=	Specify a name to identify the device. This overrides -N.
-e	--engine-id=	SNMPv3 specific parameter. Specifies the SNMP engine identifier.
-E	--exitMessage	Displays a machine readable exit message.
-f	--file=	Instructs `proliferate` to get the device information from the specified device file.
-g	--unrecognized-into-generic	When `proliferate` cannot identify a device type and this option is: ■ Specified, `proliferate` uses the closest matching attributes of the device types defined in bin.vendor to create a generically managed device type. `proliferate` adds the new device type to *entuity_home*`\etc\uncertified`, and adds the device to Entuity as a generically managed device type. Generically managed device types should be considered a temporary measure, only being used until Entuity Support provide you with the appropriate vendor definition. If a new device type cannot be built then `proliferate` adds the device to Entuity as an Unclassified device. ■ Not specified the SNMP pollable device is added to Entuity as an Unclassified device.
-h	--help	Run from the command line displays command help.
-i	--ignore-interfaces	Takes the device under management but not its interfaces, i.e. ignore all interfaces.
-I	--allow-duplicate-ip	If this option (**capitalized i**) is not specified, then proliferate runs in 'unique IP address enforcement' mode, disregarding any devices with one or more IP addresses that already exist in the main database. If this option is specified, then proliferate ignores 'unique IP address enforcement' mode. This means, for example, that Cisco routers can be added even though they share IP addresses through HSRP (Hot Standby Routing Protocol).

Table 3 proliferate Switches

Switch Short / Long		Description
-l	--level	This option (**lowercase L**) specifies the device management level, e.g. **pingOnly**, **basic**, **full**, **fullMgmtOnly** and **fullNoPorts**. Entuity also includes **web**, for use by proliferate when adding VM platforms to Entuity.
-k	--keep-slow-devices	By default `proliferate` does not add to Entuity devices that take longer to poll than the 300 seconds maximum allowed (configurable through `proliferate.maxpolltime` in `entuity.cfg` and through -K). With this option you can run `proliferate` so it accepts slow devices.
-K	--killafter=	By default `proliferate` does not add to Entuity devices that take longer to poll than the 300 seconds maximum allowed. This timeout period is configurable here and through `proliferate.maxpolltime` in `entuity.cfg`.
-m	--managed-interface-only	Running `proliferate` -m on a device results in Entuity only managing the management port. When a management port is not found then no ports are monitored. If new ports appear on the device Entuity does not manage them.
-N	--name-using=	The display name used in Entuity which when set to: ■ PolledName displays the identifier Entuity uses to poll the device. ■ SystemName displays the administrator set device system name, ■ IpAddress displays the management IP address. ■ ResolvableName displays the resolved host name of the device. ■ ResolvableNameFQ displays the fully resolved host name of the device.
-O	--owner	The owner of the `proliferate` action.
-p	--protocol=	Sets the communication protocol Entuity uses to manage a device, either IPv4 (default) or IPv6. These are the valid formats **4**, **V4**, **IPV4**, **6**, **V6**, **IPV6**.
-P	--pdu-size=	Sets the maximum PDU size.
-r	--retry=	Sets the number of SNMP poll retries.
–R	--reevaluate-device-type	This option enables a refresh of device vendor file information. For example, a device using the Not Classified Generically Managed device type, should be updated to use the appropriate vendor file as soon as you receive the vendor definition from Entuity Support. As part of the refresh the device would be assigned an appropriate device type, e.g. router, switch.
-s	--suspend-polling	Stops SNMP polling of the specified device(s).
-t	--timeout=	Sets the SNMP request timeout, in seconds. It is configurable through `eostimeoutsnmp`, by default 300 seconds.

Table 3 proliferate Switches

Switch Short / Long		Description
-T	--override-type	Associates the numeric internal Entuity identifier device type to the specified device.
-u	--user=	SNMPv3 specific parameter. Sets user security name.
-U	--update-view-membership	Updates the managed object map used as the basis for objects viewed through the web interface.
-v	--version=	Sets the SNMP version used to manage the device, where: ■ **1**, is SNMPv1. Devices solely managed through the SNMPv1 protocol will have limited device support within Entuity. ■ **2**, is SNMPv2c. ■ **2c**, is SNMPv2c. ■ **3**, is SNMPv3. ■ **default** includes both SNMPv1 and SNMPv2c support.
-V	--verbose	Puts `proliferate` into 'verbose' mode, so that it produces detailed diagnostic information.
-w	--web-polling-details	Specifies connection details for web service polling, for example for use with VM Platform device types. Enter the parameters in this order '[type],[url],[user],[password]' where type can be 2(esx)|3(oracle). You can use Escape commas where present in any of the four parts.
-x	--priv=	SNMPv3 specific parameter. Sets the privacy protocol, valid values are DES (Data Encryption Standard), 3DES, AES, AES192, AES256.
-X	--priv-pass=	SNMPv3 specific parameter. Sets the privacy password, valid values must be between eight and thirty-two characters long.
-y	--createVendorForExisting	Create a vendor file for the specified device, a device which is already in the database.
-Z	--zone	Devices can be added to a particular zone.

Table 3 proliferate Switches

Usage 1: Running with a Device File

`proliferate` compares the devices held in the current version of the device file against those that are already being managed, and adds any new devices to the Entuity database for monitoring.

```
proliferate [-v] [-I] [-t] [-f DeviceFile]
```

For example if you enter the command:

```
proliferate -I dev.txt
```

`proliferate`:

- Compares the devices in the device file, `dev.txt`, to the devices Entuity manages.

- Runs in ‘unique IP address enforcement’ mode, disregarding any devices with IP addresses that already exist in the main database.

When you run `autoDiscovey` it creates in *entuity_home*`\etc\deviceFiles` a device file, `autodisc.cfg`. This file is used by `proliferate` when adding discovered devices to Entuity.

Usage 2: Running with a Single Device

`proliferate` compares the specified IP address and community string against those Entuity already manages. `proliferate` adds new devices to the Entuity database for monitoring.

```
proliferate [-g] -d IpAddress [-c CommunityString]
```

When you add a new or updated vendor file to Entuity you should instruct Entuity to refresh the devices that use that definition so they use the latest vendor file.

`-g` is a powerful, resource intensive option and should only be used when specifying an IP address or community string. Adding one device can take fifteen minutes; Entuity do not recommend you use it with large device files.

Example 1 - Adding a device that does not have a vendor file:

When you attempt to add to Entuity a new device that is also of a new device type for which Entuity does not have a vendor file, then after entering the command:

```
proliferate -g -d 187.15.70.155 -c public
```

`proliferate`:

- Compares the device 187.15.70.155 with the devices Entuity manages.
- Attempts to create a new `bin.vendor` file definition, and adds the device to Entuity, as a Not Classified Generically Managed device.

Example 2 - Adding a device and only its management port:

You can add a device to Entuity and limit Entuity’s management of it to its management port by entering:

```
proliferate -m -d 10.25.90.155 -c public
```

`proliferate`:

- Compares the device 10.25.90.155 with the devices Entuity manages.
- Adds only the device’s management port to Entuity.

Usage 3: Adding VM Platforms

Entuity manages VM platforms through their SDK which necessitates a different set of connection attributes to other device types. Entuity recommend VM platforms are added through the web UI, but when you want to add VM platforms from the command line, the format is:

```
proliferate -d IpAddress -l manLevel -w type,url,user,password -T deviceType
```

where:

- -d *IpAddress*, identifies the device name or IP address.
- -l *manLevel*, must be set to the management level **web**.
- -w sets the web connection details, which must be comma delimited and entered in this order:
  - *type*, enter **2** for a VMware ESXi or **3** for an Oracle VM platform.
  - *url*, the url to the VM platform's SDK.
  - *user*, user account Entuity uses to access the SDK.
  - *password*, user account password.
- -T, sets the device to the internal Entuity identifier for a VM platform, i.e. 1144.

For example to add the VM platform blade to Entuity you can enter:

```
proliferate -d blade -l web -w 2,https://blade/sdk,devuser,232neree -T
1144
```

Files

`entuity.cfg`, `mib.txt`, `bin.vendor`, Device File (Seed File) and `autodisc.cfg`.

See Also

`autoDiscovery`, `showdevs`, `prolifsys` and `prolifmodule`.

prolifsys

Location	*entuity_home*\bin
Type	Process,
Invoked By	`proliferate`
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Log File	*entuity_home*\log\prolifsys.log.[1..4]

Description

Process internal to Entuity used when adding devices.

prologV2

Location	*entuity_home*\bin
Type	Process,
Invoked By	starteye
User Invocation	n/a
Invoked Processes	n/a
Configured Through	startup_O/S.cfg, entuity.cfg
Log File	*entuity_home*\log\prologV2.log.[1..4]

Description

It receives SNMPv1 and SNMPv2c traps from managed network devices and forwards them to the event system as events.

prologV2 also caches credentials for SNMPv3 devices, both managed and unmanaged. SNMPv3 traps are decoded through a configuration file, which the system administrator must manually maintain. Only successfully decrypted and authenticated traps are forwarded, all other traps are dropped.

prologV2 listens for IPv4 and IPv6 traps and informs. For IPv6 traps, when the source address:

- Matches the management IP address of the device Entuity can raise an event against the managed device.
- Does not match the device management address Entuity cannot identify the device as a managed device. Entuity raises the event as though it is against an unmanaged device, using the IPv6 address as the source of the event.

By default prologV2 listens on UDP port 162, although this can be changed using the *trapportnum* variable set in entuity.cfg.

prologV2 trap handling settings can also be configured through the Traps section of entuity.cfg. For example, *enterpriseFormat* allows you to configure Entuity to include more information to enterprise traps, *replaceEventDetailsAction* to replace problematic characters from the event details. The remaining parameters allow you to amend the setup of prologV2 to handle the rate of incoming traps.

prologV2 supports HP OpenView style expansions in trap description strings, i.e. $A $E $e $G $S $O $o $T $# $$ $*. Wildcard specific trap numbers and sub-oid matching are also supported.

protean

Location	*entuity_home*\bin
Type	Process,
Invoked By	provost, runs once a day at 02:00

User Invocation	n/a
Invoked Processes	n/a
Configured Through	`provost.conf`
Log File	*entuity_home*`\log\protean.log.[1..4]`

Description

`protean` updates the IP and VLAN network information used by other processes. `protean` uses SNMP to gather new addressing information from each device managed by Entuity, and forwards this information to the main database for storage.

provost

Location	*entuity_home*`\bin`
Type	Process,
Invoked By	`starteye`
User Invocation	n/a
Invoked Processes	All those specified in `provost.conf`
Configured Through	`startup_O/S.cfg`
Log File	*entuity_home*`\log\provost.log.[1..4]`

Description

This process is responsible for the scheduling of non-Event Stream Manager processes within the Entuity environment. `provost` is only stopped when Entuity closes down.

replace

Location	*entuity_home*`\database\bin`
Type	Utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Documentation	*entuity_home*`\database\docs\manual.htm`
Log File	n/a

Description

`replace` utility changes strings in place in files or on the standard input.

restore

Location	*entuity_home*\bin
Type	Process,
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Syntax

restore [-f]

The command can only be run if the database server `mysqld` is running without the rest of Entuity.

The **-f** parameter will suppress the prompt for confirmation immediately prior to the removal of the existing databases.

Description

`restore` destroys existing Entuity databases, and any existing `mysql.user` table, builds new ones, and recreates the tables and data from the backup files, which will have been created via the `backup` command. After running `restore`, and before restarting the Entuity server, you should run *`swmaint`* to audit and maintain the database.

You cannot restore the databases individually.

You are informed whether or not the restore has been successful.

Files

Messages relating to start, failure and completion are written to the file `restore.log` in the *entuity_home*\`log` directory (where *entuity_home* is the Entuity installation directory) This wraps to `restore.log.[1-4]` when the log becomes full. The database output is also written to `restore.log`.

See Also

`backup`

rollLog

Location	*entuity_home*\lib\tools

Type	Process
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

RollLog copies or moves a file and adds a timestamp to the filename.

Syntax

```
RollLog.exe Y|M|D|H|N|S M|C FileName [DestDir]
```

where:

- **Y**, specifies only the year (YY).
- **M**, specifies year and month (YYMM).
- **D**, specifies year, month and day (YYMMDD).
- **H**, specifies year, month, day and hours (YYMMDDHH).
- **N**, specifies year, month, day, hours and minutes (YYMMDDHHMM).
- **S**, specifies year, month, day, hours, minutes and seconds (YYMMDDHHMMSS).
- **M|C**, specifies whether to move or copy the file.
- *FileName*, full name of the file to copy or move. When the file name includes spaces use double quotes.
- *DestDir*, is an optional destination directory. When not specified the copy is done to the same directory as the source file.

runbg

Location	*entuity_home*\bin
Type	Process
Invoked By	User
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

`runbg` allows you to run Entuity binaries in the background from the command line.

setupProle

Location	*entuity_home*\bin
Type	Process,
Invoked By	startup
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Description

`setupProle` is an internal Entuity process involved in setting up `proles`.

showdevs

Location	*entuity_home*\lib\tools
Type	Process,
Invoked By	User
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

Syntax

showdevs > *outputfile.txt*

Description

The `showdevs` utility displays the devices currently being monitored in the Entuity management environment, together with their SNMP read community strings. An example of the output produced is shown below:

```
# VM Platform blade
-d 10.44.1.249 -D 10.44.1.249 -l full -c public
-d apcr1 -D apcr1 -l full -c public
-d entlonsw03 -D entlonsw03 -l full -c public
-d 10.66.24.1 -N IpAddress -l full -c public
-d 10.66.13.25 -N PolledName -l full -c public
-d 10.66.13.22 -N PolledName -l full -c public
```

One row of information appears for each device being managed. The first column displays the device name, as defined by the System Administrator. The second column displays the device community string, used for SNMP polling of the device.

This is the only tool for checking the current SNMP community string of a device. Individual community strings can be changed by using the 'Modify Attributes' option in the Administration part of the GUI, whilst global changes can be effected via the `newcommunity` utility.

Files

`entuity.cfg` (See *Chapter 3 - Entuity System Files*.)

See Also

`newcommunity` and `probity`.

slalogger

Location	*entuity_home*\`bin`
Type	Process,
Invoked By	`provost`, runs every 60 minutes
User Invocation	n/a
Invoked Processes	n/a
Configured Through	`provost.conf`, `entuity.cfg`
Log File	*entuity_home*\`log\slalogger.log.[1..4]`

`slalogger` handles the roll-up of availability data collected by `applicationMonitor`, the roll-up parameters are set through `entuity.cfg`. Roll-up information is available through reports and the availability graphs.

snmpbulkget

Location	*entuity_home*\`lib\tools`
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\`licenseTerms\Net-SNMP`.

Syntax

snmpbulkget [APPLICATION OPTIONS] [COMMON OPTIONS] OID [OID]...

Description

snmpbulkget is an SNMP application that uses the SNMP GETBULK request to query a network entity efficiently for information. One or more object identifiers (OIDs) may be given as arguments on the command line. Each variable name is given in the format specified in variables(5).

If the network entity has an error processing the request packet, an error packet will be returned and a message will be shown, helping to pinpoint why the request was malformed.

Options

- -Cn<NUM>

  Set the non-repeaters field in the GETBULK PDU. This specifies the number of supplied variables that should not be iterated over. The default is 0.

- -Cr<NUM>

  Set the max-repetitions field in the GETBULK PDU. This specifies the maximum number of iterations over the repeating variables. The default is 10.

  In addition to these options, snmpbulkget takes the common options described in the snmpcmd(1) manual page.

Example

The command:

```
snmpbulkget -v2c -Cn1 -Cr5 -Os -c public zeus system ifTable
```

retrieves the variable system.sysDescr.0 (which is the lexicographically next object to system) and the first 5 objects in the ifTable:

```
sysDescr.0 = STRING: "SunOS zeus.net.cmu.edu 4.1.3_U1 1 sun4m"
ifIndex.1 = INTEGER: 1
ifIndex.2 = INTEGER: 2
ifDescr.1 = STRING: "lo0"
et cetera.
```

As the name implies, snmpbulkget utilizes the SNMP GETBULK message, which is not available in SNMPv1.

snmpcmd

Location n/a

Type	This Man page is only available when your system administrator has separately installed man pages to the Entuity server.
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	n/a
Log File	n/a

This documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

```
snmpcmd [OPTIONS] AGENT [PARAMETERS]
```

Description

This section describes the common options for the SNMP commands: `snmpbulkget`, `snmpbulkwalk`, `snmpdelta`, `snmpget`, `snmpgetnext`, `snmpset`, `snmpstatus`, `snmptable`, `snmptest`, `snmptrap`, `snmpdf`, `snmpusm`, `snmpwalk`. The command line applications use the SNMP protocol to communicate with an SNMP capable network entity, an agent. Individual applications typically (but not necessarily) take additional parameters that are given after the agent specification. These parameters are documented in the manual pages for each application.

Options

- -3[MmKk] 0xHEXKEY

  Sets the keys to be used for SNMPv3 transactions. These options allow you to set the master authentication and encryption keys (-3m and -3M respectively) or set the localized authentication and encryption keys (-3k and -3K respectively). SNMPv3 keys can be either passed in by hand using these flags, or by the use of keys generated from passwords using the -A and -X flags discussed below. For further details on SNMPv3 and its usage of keying information, see the Net-SNMP tutorial web site (http://www.Net-SNMP.org/tutorial-5/commands/). Overrides the defAuthMasterKey (-3m), defPrivMasterKey (-3M), defAuthLocalizedKey (-3k) or defPrivLocalizedKey (-3K) tokens, respectively, in the snmp.conf file, see snmp.conf(5).

- -a authProtocol

  Set the authentication protocol (MD5 or SHA) used for authenticated SNMPv3 messages. Overrides the defAuthType token in the snmp.conf file.

- -A authPassword

  Set the authentication pass phrase used for authenticated SNMPv3 messages. Overrides the defAuthPassphrase token in the snmp.conf file. It is insecure to specify pass phrases on the command line, see snmp.conf(5).

- -c community

  Set the community string for SNMPv1/v2c transactions. Overrides the defCommunity token in the snmp.conf file.

- -d

  Dump (in hexadecimal) the raw SNMP packets sent and received.

- -D TOKEN[,...]

  Turn on debugging output for the given TOKEN(s). Try ALL for extremely verbose output.

- -e engineID

  Set the authoritative (security) engineID used for SNMPv3 REQUEST messages. It is typically not necessary to specify this, as it will usually be discovered automatically.

- -E engineID

  Set the context engineID used for SNMPv3 REQUEST messages scopedPdu. If not specified, this will default to the authoritative engineID.

- -h, --help

  Display a brief usage message and then exit.

- -H

  Display a list of configuration file directives understood by the command and then exit.

- -I [brRhu]

  Specifies input parsing options. See INPUT OPTIONS below.

- -l secLevel

  Set the security level used for SNMPv3 messages (noAuthNoPriv|authNoPriv|authPriv). Appropriate pass phrase(s) must provided when using any level higher than noAuthNoPriv. Overrides the defSecurityLevel token in the snmp.conf file.

- -L [eEfFoOsS]

  Specifies output logging options.

- -m MIBLIST

  Specifies a colon separated list of MIB modules (not files) to load for this application. This overrides (or augments) the environment variable MIBS, the snmp.conf directive mibs, and the list of MIBs hardcoded into the Net-SNMP library.

  If MIBLIST has a leading '-' or '+' character, then the MIB modules listed are loaded in addition to the default list, coming before or after this list respectively. Otherwise, the specified MIBs are loaded instead of this default list.

  The special keyword ALL is used to load all MIB modules in the MIB directory search list. Every file whose name does not begin with "." will be parsed as if it were a MIB file.

- -M DIRLIST

  Specifies a colon separated list of directories to search for MIBs. This overrides (or augments) the environment variable MIBDIRS, the snmp.conf directive mibdirs, and the

default directory hardcoded into the Net-SNMP library (`/usr/local/share/snmp/mibs`).

If DIRLIST has a leading '-' or '+' character, then the given directories are added to the default list, being searched before or after the directories on this list respectively. Otherwise, the specified directories are searched instead of this default list.

Note that the directories appearing later in the list take precedence over earlier ones. To avoid searching any MIB directories, set the MIBDIRS environment variable to the empty string ("").

Note that MIBs specified using the -m option or the mibs configuration directive will be loaded from one of the directories listed by the -M option (or equivalents). The mibfile directive takes a full path to the specified MIB file, so this does not need to be in the MIB directory search list.

- -n contextName

  Set the contextName used for SNMPv3 messages. The default contextName is the empty string "". Overrides the defContext token in the snmp.conf file.

- -O [abeEfnqQsStTuUvxX]

  Specifies output printing options.

- -P [cdeRuwW]

  Specifies MIB parsing options.

- -r retries

  Specifies the number of retries to be used in the requests. The default is 5.

- -t timeout

  Specifies the timeout in seconds between retries. The default is 1.

- -u secName

  Set the securityName used for authenticated SNMPv3 messages. Overrides the defSecurityName token in the snmp.conf file.

- -v 1 | 2c | 3

  Specifies the protocol version to use: 1 (RFCs 1155-1157), 2c (RFCs 1901-1908), or 3 (RFCs 2571-2574). The default is typically version 3. Overrides the defVersion token in the snmp.conf file.

- -V, --version

  Display version information for the application and then exit.

- -x privProtocol

Set the privacy protocol (DES or AES) used for encrypted SNMPv3 messages. Overrides the defPrivType token in the `snmp.conf` file. This option is only valid if the Net-SNMP software was build to use OpenSSL.

- -X privPassword

  Set the privacy pass phrase used for encrypted SNMPv3 messages. Overrides the defPrivPassphrase token in the snmp.conf file. It is insecure to specify pass phrases on the command line, see snmp.conf(5).

- -Z boots,time

  Set the engineBoots and engineTime used for authenticated SNMPv3 messages. This will initialize the local notion of the agents boots/time with an authenticated value stored in the LCD. It is typically not necessary to specify this option, as these values will usually be discovered automatically.

- -Yname=value

  `--name=value`

  Allows to specify any token ("name") supported in the snmp.conf file and sets its value to "value". Overrides the corresponding token in the snmp.conf file. See snmp.conf(5) for the full list of tokens.

Agent Specification

The string AGENT in the SYNOPSIS above specifies the remote SNMP entity with which to communicate. This specification takes the form:

```
[<transport-specifier>:]<transport-address>
```

At its simplest, the AGENT specification may consist of a hostname, or an IPv4 address in the standard "dotted quad" notation. In this case, communication will be attempted using UDP/IPv4 to port 161 of the given host. Otherwise, the <transport-address> part of the specification is parsed according to the following table:

```
<transport-specifier>
<transport-address> format
udp hostname[:port] or IPv4-address[:port]
tcp hostname[:port] or IPv4-address[:port]
unix pathname
ipx [network]:node[/port]
aal5pvc or pvc
[interface.][VPI.]VCI
udp6 or udpv6 or udpipv6
           hostname[:port] or IPv6-address:port or
           '['IPv6-address']'[:port]
tcp6 or tcpv6 or tcpipv6
           hostname[:port] or IPv6-address:port or
           '['IPv6-address']'[:port]
```

<transport-specifier> strings are case-insensitive so that, for example, "tcp" and "TCP" are equivalent. Here are some examples, along with their interpretation:

- hostname:161

  perform query using UDP/IPv4 datagrams to hostname on port 161. The ":161" is redundant here since that is the default SNMP port in any case.

- udp:hostname

  identical to the previous specification. The "udp:" is redundant here since UDP/IPv4 is the default transport.

- TCP:hostname:1161

  connect to hostname on port 1161 using TCP/IPv4 and perform query over that connection.

- ipx::00D0B7AAE308

  perform query using IPX datagrams to node number 00D0B7AAE308 on the default network, and using the default IPX port of 36879 (900F hexadecimal), as suggested in RFC 1906.

- ipx:0AE43409:00D0B721C6C0/1161

  perform query using IPX datagrams to port 1161 on node number 00D0B721C6C0 on network number 0AE43409.

- unix:/tmp/local-agent

  connect to the Unix domain socket /tmp/local-agent, and perform the query over that connection.

- /tmp/local-agent

  identical to the previous specification, since the Unix domain is the default transport if the first character of the <transport-address> is a '/'.

- AAL5PVC:100

  perform the query using AAL5 PDUs sent on the permanent virtual circuit with VPI=0 and VCI=100 (decimal) on the first ATM adapter in the machine.

- PVC:1.10.32

  perform the query using AAL5 PDUs sent on the permanent virtual circuit with VPI=10 (decimal) and VCI=32 (decimal) on the second ATM adapter in the machine. Note that "PVC" is a synonym for "AAL5PVC".

- udp6:hostname:10161

  perform the query using UDP/IPv6 datagrams to port 10161 on hostname (which will be looked up as an AAAA record).

- UDP6:[fe80::2d0:b7ff:fe21:c6c0]

  perform the query using UDP/IPv6 datagrams to port 161 at address fe80::2d0:b7ff:fe21:c6c0.

- tcpipv6:[::1]:1611

  connect to port 1611 on the local host (::1 in IPv6 parlance) using TCP/IPv6 and perform query over that connection.

Not all the transport domains listed above will always be available; for instance, hosts with no IPv6 support will not be able to use udp6 transport addresses, and attempts to do so will result in the error "Unknown host". Likewise, since AAL5 PVC support is only currently available on Linux, it will fail with the same error on other platforms.

MIB Parsing Options

The Net-SNMP MIB parser mostly adheres to the Structure of Management Information (SMI). As that specification has changed through time, and in recognition of the diversity in compliance expressed in MIB files, additional options provide more flexibility in reading MIB files.

- -Pc

  Allow ASN.1 comments to extend to the end of the MIB source line. Strictly speaking, a second appearance of "--" should terminate the comment, but this breaks some MIB files. This behaviour can also be set with the configuration token strictCommentTerm.

- -Pd

  Disables saving the DESCRIPTION of MIB objects when parsing MIB files, reducing the amount of memory used by the running application.

- -Pe

  Show errors encountered when parsing MIB files. These include references to IMPORTed modules and MIB objects that cannot be located in the MIB directory search list. This can also be set with the configuration token showMibErrors.

- -PR

  If the same MIB object (parent name and sub-identifier) appears multiple times in the list of MIB definitions loaded, use the last version to be read in. By default, the first version will be used, and any duplicates discarded. This behaviour can also be set with the configuration token mibReplaceWithLatest.

  Such ordering is normally only relevant if there are two MIB files with conflicting object definitions for the same OID (or different revisions of the same basic MIB object).

- -Pu

  Allow the underline character in MIB object names and other symbols. Strictly speaking, this is not valid SMI syntax, but some vendor MIB files define such names. This can also be set with the configuration token mibAllowUnderline.

- -Pw

  Show various warning messages in parsing MIB files and building the overall OID tree. This can also be set with the configuration directive mibWarningLevel 1.

- -PW

Show some additional warning messages, mostly relating to parsing individual MIB objects. This can also be set with the configuration directive mibWarningLevel 2.

Output Options

The format of the output from SNMP commands can be controlled using various parameters of the -O flag. The effects of these sub-options can be seen by comparison with the following default output (unless otherwise specified):

```
$ snmpget -c public -v 1 localhost sysUpTime.0
SNMPv2-MIB::sysUpTime.0 = Timeticks: (14096763) 1 day, 15:09:27.63
```

- -Oa

  Display string values as ASCII strings (unless there is a DISPLAY-HINT defined for the corresponding MIB object). By default, the library attempts to determine whether the value is a printable or binary string, and displays it accordingly.

  This option does not affect objects that do have a Display Hint.

- -Ob

  Display table indexes numerically, rather than trying to interpret the instance subidentifiers as string or OID values:

```
$ snmpgetnext -c public -v 1 localhost vacmSecurityModel
SNMP-VIEW-BASED-ACM-MIB::vacmSecurityModel.0."wes" = xxx
$ snmpgetnext -c public -v 1 -Ob localhost vacmSecurityModel
SNMP-VIEW-BASED-ACM-MIB::vacmSecurityModel.0.3.119.101.115 = xxx
```

- -Oe

  Removes the symbolic labels from enumeration values:

```
$ snmpget -c public -v 1 localhost ipForwarding.0
IP-MIB::ipForwarding.0 = INTEGER: forwarding(1)
$ snmpget -c public -v 1 -Oe localhost ipForwarding.0
IP-MIB::ipForwarding.0 = INTEGER: 1
```

- -OE

  Modifies index strings to escape the quote characters:

```
$ snmpgetnext -c public -v 1 localhost vacmSecurityModel
SNMP-VIEW-BASED-ACM-MIB::vacmSecurityModel.0."wes" = xxx
$ snmpgetnext -c public -v 1 -OE localhost vacmSecurityModel
SNMP-VIEW-BASED-ACM-MIB::vacmSecurityModel.0.\"wes\" = xxx
```

  This allows the output to be reused in shell commands.

- -Of

  Include the full list of MIB objects when displaying an OID:

```
.iso.org.dod.internet.mgmt.mib-2.system.sysUpTime.0 =
```

```
Timeticks: (14096763) 1 day, 15:09:27.63
```

- -On

  Displays the OID numerically:

  ```
  .1.3.6.1.2.1.1.3.0 = Timeticks: (14096763) 1 day, 15:09:27.63
  ```

- -Oq

  Removes the equal sign and type information when displaying varbind values:

  ```
  SNMPv2-MIB::sysUpTime.0 1:15:09:27.63
  ```

- -OQ

  Removes the type information when displaying varbind values:

  ```
  SNMPv2-MIB::sysUpTime.0 = 1:15:09:27.63
  ```

- -Os

  Display the MIB object name (plus any instance or other subidentifiers):

  ```
  sysUpTime.0 = Timeticks: (14096763) 1 day, 15:09:27.63
  ```

- -OS

  Display the name of the MIB, as well as the object name:

  ```
  SNMPv2-MIB::sysUpTime.0 = Timeticks: (14096763) 1 day, 15:09:27.63
  ```

  This is the default OID output format.

- -Ot

  Display TimeTicks values as raw numbers:

  ```
  SNMPv2-MIB::sysUpTime.0 = 14096763
  ```

- -OT

  If values are printed as Hex strings, display a printable version as well.

- -Ou

  Display the OID in the traditional UCD-style (inherited from the original CMU code). That means removing a series of "standard" prefixes from the OID, and displaying the remaining list of MIB object names (plus any other subidentifiers):

  system.sysUpTime.0 = Timeticks: (14096763) 1 day, 15:09:27.63

- -OU

  Do not print the UNITS suffix at the end of the value.

- -Ov

  Display the varbind value only, not the OID:

  ```
  $ snmpget -c public -v 1 -Oe localhost ipForwarding.0
  INTEGER: forwarding(1)
  ```

- -Ox

Display string values as Hex strings (unless there is a DISPLAY-HINT defined for the corresponding MIB object). By default, the library attempts to determine whether the value is a printable or binary string, and displays it accordingly.

This option does not affect objects that do have a Display Hint.

- -OX

  Display table indexes in a more "program like" output, imitating a traditional array-style index format:

```
$ snmpgetnext -c public -v 1 localhost ipv6RouteTable
IPv6-MIB::ipv6RouteIfIndex.63.254.1.0.255.0.0.0.0.0.0.0.0.0.0.64.1 =
INTEGER: 2
$ snmpgetnext -c public -v 1 -OE localhost ipv6RouteTable
IPv6-MIB::ipv6RouteIfIndex[3ffe:100:ff00:0:0:0:0:0][64][1] = INTEGER:
2
```

Most of these options can also be configured via configuration tokens. See the snmp.conf(5) manual page for details.

Logging Options

The mechanism and destination to use for logging of warning and error messages can be controlled by passing various parameters to the -L flag.

- -Le

  Log messages to the standard error stream.

- -Lf FILE

  Log messages to the specified file.

- -Lo

  Log messages to the standard output stream.

- -Ls FACILITY

  Log messages via syslog, using the specified facility ('d' for LOG_DAEMON, 'u' for LOG_USER, or '0'-'7' for LOG_LOCAL0 through LOG_LOCAL7).

  There are also "upper case" versions of each of these options, which allow the corresponding logging mechanism to be restricted to certain priorities of message. Using standard error logging as an example:

```
-LE pri
```

  will log messages of priority 'pri' and above to standard error.

- -LE p1-p2

  will log messages with priority between 'p1' and 'p2' (inclusive) to standard error.

  For -LF and -LS the priority specification comes before the file or facility token. The priorities recognised are:

- 0 or ! for LOG_EMERG,
- 1 or a for LOG_ALERT,
- 2 or c for LOG_CRIT,
- 3 or e for LOG_ERR,
- 4 or w for LOG_WARNING,
- 5 or n for LOG_NOTICE,
- 6 or i for LOG_INFO, and
- 7 or d for LOG_DEBUG.

Normal output is (or will be!) logged at a priority level of LOG_NOTICE

Input Options

The interpretation of input object names and the values to be assigned can be controlled using various parameters of the -I flag. The default behaviour will be described at the end of this section.

- -Ib

  Specifies that the given name should be regarded as a regular expression, to match (case-insensitively) against object names in the MIB tree. The "best" match will be used - calculated as the one that matches the closest to the beginning of the node name and the highest in the tree. For example, the MIB object vacmSecurityModel could be matched by the expression vacmsecuritymodel (full name, but different case), or vacm.*model (regexp pattern).

'.' is a special character in regular expression patterns, so the expression cannot specify instance subidentifiers or more than one object name. A "best match" expression will only be applied against single MIB object names. For example, the expression sys*ontact.0 would not match the instance sysContact.0 (although sys*ontact would match sysContact). Similarly, specifying a MIB module name will not succeed (so SNMPv2-MIB::sys.*ontact would not match either).

- -Ih

  Disables the use of DISPLAY-HINT information when assigning values. This would then require providing the raw value:

  ```
  snmpset ... HOST-RESOURCES-MIB::hrSystemData.0
  x "07 D2 0C 0A 02 04 06 08"
  ```

  instead of a formatted version:

  ```
  snmpset ... HOST-RESOURCES-MIB::hrSystemDate.0
  = 2002-12-10,2:4:6.8
  ```

- -Ir

Disables checking table indexes and the value to be assigned against the relevant MIB definitions. This will (hopefully) result in the remote agent reporting an invalid request, rather than checking (and rejecting) this before it is sent to the remote agent.

Local checks are more efficient (and the diagnostics provided also tend to be more precise), but disabling this behaviour is particularly useful when testing the remote agent.

- -IR

  Enables "random access" lookup of MIB names. Rather than providing a full OID path to the desired MIB object (or qualifying this object with an explicit MIB module name), the MIB tree will be searched for the matching object name. Thus .iso.org.dod.internet.mib-2.system.sysDescr.0 (or SNMPv2-MIB::sysDescr.0) can be specified simply as sysDescr.0.

Since MIB object names are not globally unique, this approach may return a different MIB object depending on which MIB files have been loaded. The MIB-MODULE::objectName syntax has the advantage of uniquely identifying a particular MIB object, as well as being slightly more efficient (and automatically loading the necessary MIB file if necessary).

- -Is SUFFIX

  Adds the specified suffix to each textual OID given on the command line. This can be used to retrieve multiple objects from the same row of a table, by specifying a common index value.

- -IS PREFIX

  Adds the specified prefix to each textual OID given on the command line. This can be used to specify an explicit MIB module name for all objects being retrieved (or for incurably lazy typists).

- -Iu

  Enables the traditional UCD-style approach to interpreting input OIDs. This assumes that OIDs are rooted at the 'mib-2' point in the tree (unless they start with an explicit '.' or include a MIB module name). So the sysDescr instance above would be referenced as system.sysDescr.0.

Object names specified with a leading '.' are always interpreted as "fully qualified" OIDs, listing the sequence of MIB objects from the root of the MIB tree. Such objects and those qualified by an explicit MIB module name are unaffected by the -Ib, -IR and -Iu flags.

Otherwise, if none of the above input options are specified, the default behaviour for a "relative" OID is to try and interpret it as an (implicitly) fully qualified OID, then apply "random access" lookup (-IR), followed by "best match" pattern matching (-Ib).

Environment Variables

PREFIX

The standard prefix for object identifiers (when using UCD-style output). Defaults to .iso.org.dod.internet.mgmt.mib-2

MIBS

The list of MIBs to load. Defaults to SNMPv2-TC:SNMPv2-MIB:IF-MIB:IP-MIB:TCP-MIB:UDP-MIB:SNMP-VACM-MIB. Overridden by the -m option.

MIBDIRS

The list of directories to search for MIBs. Defaults to `/usr/local/share/snmp/mibs`. Overridden by the -M option.

See Also

`snmpget`, `snmpgetnext`, `snmpset`, `snmpbulkget`, `snmpbulkwalk`, `snmpwalk`, snmptable, `snmpdelta`, `snmptrap`, `snmpinform`, `snmpusm`, `snmpstatus`, `snmptest(1)`, `snmp.conf`.

snmpdelta

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*`\licenseTerms\Net-SNMP`.

Syntax

snmpdelta [common options] [-Cf] [-Ct] [-Cs] [-CS] [-Cm] [-CF configfile] [-Cl] [-Cp period] [-CP Peaks] [-Ck] [-CT] AGENT OID [OID ...]

Description

snmpdelta will monitor the specified integer valued OIDs, and report changes over time.

AGENT identifies a target SNMP agent, which is instrumented to monitor the given objects. At its simplest, the AGENT specification will consist of a hostname or an IPv4 address. In this situation, the command will attempt communication with the agent, using UDP/IPv4 to port 161 of the given target host. See snmpcmd(1) for a full list of the possible formats for AGENT.

OID is an object identifier which uniquely identifies the object type within a MIB. Multiple OIDs can be specified on a single snmpdelta command.

Options

COMMON OPTIONS

Please see snmpcmd for a list of possible values for COMMON OPTIONS as well as their descriptions.

- -Cf

  Don't fix errors and retry the request. Without this option, if multiple oids have been specified for a single request and if the request for one or more of the oids fails, snmpdelta will retry the request so that data for oids apart from the ones that failed will still be returned. Specifying -Cf tells snmpdelta not to retry a request, even if there are multiple oids specified.

- -Ct

  Flag will determine time interval from the monitored entity.

- -Cs

  Flag will display a timestamp.

- -CS

  Generates a "sum count" in addition to the individual instance counts. The "sum count" is the total of all the individual deltas for each time period.

- -Cm

  Prints the maximum value ever attained.

- -CF configfile

  Tells snmpdelta to read it's configuration from the specified file. This options allows the input to be set up in advance rather than having to be specified on the command line.

- -Cl

  Tells snmpdelta to write it's configuration to files whose names correspond to the MIB instances monitored. For example, snmpdelta -Cl localhost ifInOctets.1 will create a file "localhost-ifInOctets.1".

- -Cp

  Specifies the number of seconds between polling periods. Polling constitutes sending a request to the agent. The default polling period is one second.

- -CP peaks

  Specifies the reporting period in number of polling periods. If this option is specified, snmpdelta polls the agent peaks number of times before reporting the results. The result reported includes the average value over the reporting period. In addition, the highest polled value within the reporting period is shown.

- -Ck

  When the polling period (-Cp) is an increment of 60 seconds and the timestamp is displayed in the output (-Cs), then the default display shows the timestamp in the format hh:mm mm/dd. This option causes the timestamp format to be hh:mm:ss mm/dd.

- -CT

  Makes snmpdelta print its output in tabular form.

  -Cv vars/pkt

Specifies the maximum number of oids allowed to be packaged in a single PDU. Multiple PDUs can be created in a single request. The default value of variables per packet is 60. This option is useful if a request response results in an error because the packet is too big.

Examples

```
$ snmpdelta -c public -v 1 -Cs localhost IF-MIB::ifInUcastPkts.3 IF-
MIB::ifOutUcastPkts.3
[20:15:43 6/14] ifInUcastPkts.3 /1 sec: 158
[20:15:43 6/14] ifOutUcastPkts.3 /1 sec: 158
[20:15:44 6/14] ifInUcastPkts.3 /1 sec: 184
[20:15:44 6/14] ifOutUcastPkts.3 /1 sec: 184
[20:15:45 6/14] ifInUcastPkts.3 /1 sec: 184
[20:15:45 6/14] ifOutUcastPkts.3 /1 sec: 184
[20:15:46 6/14] ifInUcastPkts.3 /1 sec: 158
[20:15:46 6/14] ifOutUcastPkts.3 /1 sec: 158
[20:15:47 6/14] ifInUcastPkts.3 /1 sec: 184
[20:15:47 6/14] ifOutUcastPkts.3 /1 sec: 184
[20:15:48 6/14] ifInUcastPkts.3 /1 sec: 184
[20:15:48 6/14] ifOutUcastPkts.3 /1 sec: 184
[20:15:49 6/14] ifInUcastPkts.3 /1 sec: 158
[20:15:49 6/14] ifOutUcastPkts.3 /1 sec: 158
^C
$ snmpdelta -c public -v 1 -Cs -CT localhost IF-MIB:ifInUcastPkts.3
IF-MIB:ifOutcastPkts.3
localhost       ifInUcastPkts.3 ifOutUcastPkts.3
[20:15:59 6/14] 184.00  184.00
[20:16:00 6/14] 158.00  158.00
[20:16:01 6/14] 184.00  184.00
[20:16:02 6/14] 184.00  184.00
[20:16:03 6/14] 158.00  158.00
[20:16:04 6/14] 184.00  184.00
[20:16:05 6/14] 184.00  184.00
[20:16:06 6/14] 158.00  158.00
^C
```

The following example uses a number of options. Since the Cl option is specified, the output is sent to a file and not to the screen.

```
$ snmpdelta -c public -v 1 -Ct -Cs -CS -Cm -Cl -Cp 60 -CP 60
```

```
        interlink.sw.net.cmu.edu    .1.3.6.1.2.1.2.2.1.16.3
.1.3.6.1.2.1.2.2.1.16.4
fi
```

snmpdf

Location	*entuity_home*/lib/tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

```
snmpdf [COMMON OPTIONS] [-Cu] AGENT
```

Description

snmpdf is simply a networked version of the typical df command. It checks the disk space on the remote machine by examining the HOST-RESOURCES-MIB's hrStorageTable or the UCD-SNMP-MIB's dskTable. By default, the hrStorageTable is preferred as it typically contains more information. However, the -Cu argument can be passed to snmpdf to force the usage of the dskTable.

AGENT identifies a target SNMP agent, which is instrumented to monitor the given objects. At its simplest, the AGENT specification will consist of a hostname or an IPv4 address. In this situation, the command will attempt communication with the agent, using UDP/IPv4 to port 161 of the given target host. See the snmpcmd(1) manual page for a full list of the possible formats for AGENT.

See the snmpd.conf(5) manual page on setting up the dskTable using the disk directive in the snmpd.conf file.

Options

Please see snmpcmd(1) for a list of possible values for COMMON OPTIONS as well as their descriptions.

- -Cu

  Forces the command to use dskTable in mib UCD-SNMP-MIB instead of the default to determine the storage information. Generally, the default use of hrStorageTable in mib HOST-RESOURCES-MIB is preferred because it typically contains more information.

Examples

```
% snmpdf -v 2c -c public localhost
Description       size (kB)         Used    Available Used%
/                   7524587      2186910      5337677   29%
/proc                     0            0            0    0%
/etc/mnttab               0            0            0    0%
/var/run            1223088           32      1223056    0%
/tmp                1289904        66848      1223056    5%
/cache               124330         2416       121914    1%
/vol                      0            0            0    0%
Real Memory          524288       447456        76832   85%
Swap Space          1420296       195192      1225104   13%
```

snmpdump

Location	*entuity_home*\lib\tools
Type	Utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

Syntax

```
snmpdump [OPTIONS] hostname
```

Description

snmpdump is an SNMP application that uses SNMP GETNEXT requests to query and return the full MIB, including the enterprise section (unlike the OID specific snmpwalk). snmpdump is more tolerant of faults and loops than snmpwalk. By default snmpdump:

- Has 6 retries with a 10 second timeout.
- Assumes SNMP version 2c, with public as the community string.
- Uses MIB Loop detection, which you can turn off using -C switch.

When snmpdump completes it displays an "End of MIB" message, number of variables and the time taken.

You can configure snmpdump to work with SNMP v3.

Options

These options are available with snmpdump:

- -h, --help

  Display a brief usage message and then exit.

- -C

  Turn off loop checking.

- -v 1 | 2c | 3

  Specifies the protocol version to use: 1 (RFCs 1155-1157), 2c (RFCs 1901-1908), or 3 (RFCs 2571-2574). The default is version 2c.

- -V, --version

  Display version information for the application and then exit.

This is a SNMP version 1 and 2c specific option:

- -c COMMUNITY

  Set the community string for SNMPv1/v2c transactions, default public.

These are SNMP version 3 specific options:

- -a PROTOCOL

  Set the authentication protocol (MD5 or SHA) used for authenticated SNMPv3 messages.

- -A PASSPHRASE

  Set the authentication pass phrase used for authenticated SNMPv3 messages.

- -e ENGINE-ID

  Set the authoritative (security) engineID used for SNMPv3 REQUEST messages. It is typically not necessary to specify this, as it will usually be discovered automatically.

- -E ENGINE-ID

  Set the context engineID used for SNMPv3 REQUEST messages scopedPdu. If not specified, this will default to the authoritative engineID.

- -l LEVEL

  Set the security level used for SNMPv3 messages (noAuthNoPriv|authNoPriv|authPriv). Appropriate pass phrase(s) must provided when using any level higher than noAuthNoPriv.

- -n CONTEXT

  Set the contextName used for SNMPv3 messages. The default contextName is the empty string "".

- -u USER-NAME

  Set the securityName used for authenticated SNMPv3 messages.

- -x PROTOCOL

  Set the privacy protocol (DES or AES) used for encrypted SNMPv3 messages.

- -X PASSPHRASE

  Set the privacy pass phrase used for encrypted SNMPv3 messages.

- -Z BOOTS,TIME

  Set the engineBoots and engineTime used for authenticated SNMPv3 messages. This will initialize the local notion of the agents boots/time with an authenticated value stored in the LCD. It is typically not necessary to specify this option, as these values will usually be discovered automatically.

These are general communication options:

- -r RETRIES

  Specifies the number of retries to be used in the requests. The default is 5.

- -t TIMEOUT

  Specifies the timeout in seconds between retries. The default is 1.

Example

The command:

```
snmpdump 10.1.1.1
```

will retrieve the full MIB.

snmpget

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

snmpget [COMMON OPTIONS] [-Cf] OID [OID]...

Description

snmpget is an SNMP application that uses the SNMP GET request to query for information on a network entity. One or more object identifiers (OIDs) may be given as arguments on the command line. Each variable name is given in the format specified in variables(5).

For example:

```
snmpget -c public zeus system.sysDescr.0
```

will retrieve the variable system.sysDescr.0:

```
system.sysDescr.0 = "SunOS zeus.net.cmu.edu 4.1.3_U1 1 sun4m"
```

If the network entity has an error processing the request packet, an error packet will be returned and a message will be shown, helping to pinpoint in what way the request was malformed. If there were other variables in the request, the request will be resent without the bad variable.

Options

- -Cf

  If -Cf is not specified, some applications (snmpdelta, snmpget, snmpgetnext and snmpstatus) will try to fix errors returned by the agent that you were talking to and resend the request. The only time this is really useful is if you specified a OID that didn't exist in your request and you're using SNMPv1 which requires "all or nothing" kinds of requests. Here is an example (note that system.sysUpTime is an incomplete OID as it needs the .0 index appended to it):

```
snmpget -v1 -Cf -c public localhost system.sysUpTime
system.sysContact.0
Errorinpacket
Reason: (noSuchName) There is no such variable name in this MIB.
This name doesn't exist: system.sysUpTime


snmpget -v1 -c public localhost system.sysUpTime system.sysContact.0
Error in packet
Reason: (noSuchName) There is no such variable name in this MIB.
This name doesn't exist: system.sysUpTime


system.sysContact.0 = STRING: root@localhost
```

  With the -Cf specified the application will not try to fix the PDU for you.

  In addition to this option, snmpget takes the common options described in the snmpcmd(1) manual page.

snmpgetnext

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line

Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

snmpgetnext [COMMON OPTIONS] [-Cf] OID [OID]...

Description

snmpget is an SNMP application that uses the SNMP GETNEXT request to query for information on a network entity. One or more object identifiers (OIDs) may be given as arguments on the command line. Each variable name is given in the format specified in variables(5). For each one, the variable that is lexicographically "next" in the remote entity's MIB will be returned.

For example:

```
snmpgetnext -c public zeus interfaces.ifTable.ifEntry.ifType.1
```

will retrieve the variable interfaces.ifTable.ifEntry.ifType.2:

```
interfaces.ifTable.ifEntry.ifType.2 = softwareLoopback(24)
```

If the network entity has an error processing the request packet, an error message will be shown, helping to pinpoint in what way the request was malformed.

Options

snmpgetnext takes the common options described in the snmpcmd(1) manual page and also the -Cf option described in the snmpget(1) manual page

snmpset

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

snmpset [COMMON OPTIONS] OID TYPE VALUE [OID TYPE VALUE]...

Description

`snmpset` is an SNMP application that uses the SNMP SET request to set information on a network entity. One or more object identifiers (OIDs) must be given as arguments on the command line. A type and a value to be set must accompany each object identifier. Each variable name is given in the format specified in variables(5).

The TYPE is a single character, one of:

i	INTEGER
u	UNSIGNED
s	STRING
x	HEX STRING
d	DECIMAL STRING
n	NULLOBJ
o	OBJID
t	TIMETICKS
a	IPADDRESS
b	BITS

Most of these will use the obvious corresponding ASN.1 type. 's', 'x', 'd' and 'b' are all different ways of specifying an OCTET STRING value, and the 'u' unsigned type is also used for handling Gauge32 values.

If you have the proper MIB file loaded, you can, in most cases, replace the type with an '=' sign. For an object of type OCTET STRING this will assume a string like the 's' type notation. For other types it will do "The Right Thing".

For example:

```
snmpset -c private -v 1 test-hub system.sysContact.0 s
dpz@noc.rutgers.edu ip.ipforwarding.0 = 2
```

will set the variables sysContact.0 and ipForwarding.0:

```
system.sysContact.0 = STRING: "pgp@entuity.com"
ip.ipForwarding.0 = INTEGER: not-forwarding(2)
```

If the network entity has an error processing the request packet, an error packet will be returned and a message will be shown, helping to pinpoint in what way the request was malformed.

Options

- Common options

  See `snmpcmd` for a list of possible values for common options.

snmpstatus

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation are provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

```
snmpstatus [COMMON OPTIONS] [-Cf] AGENT
```

Description

snmpstatus is an SNMP application that retrieves several important statistics from a network entity.

AGENT identifies a target SNMP agent, which is instrumented to monitor the given objects. At its simplest, the AGENT specification will consist of a hostname or an IPv4 address. In this situation, the command will attempt communication with the agent, using UDP/IPv4 to port 161 of the given target host.

See the snmpcmd for a full list of the possible formats for AGENT.

The information returned is:

- The IP address of the entity.
- A textual description of the entity (sysDescr.0).
- The uptime of the entity's SNMP agent (sysUpTime.0).
- The sum of received packets on all interfaces (ifInUCastPkts.* + ifInNUCastPkts.*).
- The sum of transmitted packets on all interfaces (ifOutUCastPkts.* + ifOutNUCastPkts.*).
- The number of IP input packets (ipInReceives.0).
- The number of IP output packets (ipOutRequests.0).

For example:

```
snmpstatus -c public -v 1 netdev-kbox.cc.cmu.edu
```

will produce output similar to the following:

```
[128.2.56.220]=>[Kinetics FastPath2] Up: 1 day, 4:43:31
Interfaces: 1, Recv/Trans packets: 262874/39867 | IP: 31603/15805
```

snmpstatus also checks the operational status of all interfaces (ifOperStatus.*), and if it finds any that are not running, it will report in a manner similar to this:

```
2 interfaces are down!
```

If the network entity has an error processing the request packet, an error packet will be returned and a message will be shown, helping to pinpoint in what way the request was malformed. snmpstatus will attempt to reform its request to eliminate the malformed variable (unless the -Cf option is given, see below), but this variable will then be missing from the displayed data.

Options

- Common options

  See snmpcmd for a list of possible values for common options.

- -Cf

  By default, snmpstatus will try to fix errors returned by the agent and retry a request. In this situation, the command will display the data that it can. If the -Cf option is specified, then snmpstatus will not try to fix errors, and the error will cause the command to terminate.

snmptable

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

snmptable [COMMON OPTIONS] [-Cb] [-CB] [-Ch] [-CH] [-Ci] [-Cf STRING] [-Cw WIDTH] AGENT TABLE-OID

Description

snmptable is an SNMP application that repeatedly uses the SNMP GETNEXT or GETBULK requests to query for information on a network entity. The parameter TABLE-OID must specify an SNMP table.

snmptable is an SNMP application that repeatedly uses the SNMP GETNEXT or GETBULK requests to query for information on a network entity. The parameter TABLE-OID must specify an SNMP table.

AGENT identifies a target SNMP agent, which is instrumented to monitor the given objects. At its simplest, the AGENT specification will consist of a hostname or an IPv4 address. In this situation, the command will attempt communication with the agent, using UDP/IPv4 to port 161 of the given target host. See snmpcmd(1) for a full list of the possible formats for AGENT.

Options

- Common options

  See snmpcmd for a list of possible values for common options.

- -Cb

  Display only a brief heading. Any common prefix of the table field names will be deleted.

- -CB

  Do not use GETBULK requests to retrieve data, only GETNEXT.

- -Cc CHARS

  Print table in columns of CHARS characters width.

- -Cf STRING

  The string STRING is used to separate table columns. With this option, each table entry will be printed in compact form, just with the string given to separate the columns (useful if you want to import it into a database). Otherwise it is printed in nicely aligned columns.

- -Ch

  Display only the column headings.

- -CH

  Do not display the column headings.

- -Ci

  This option prepends the index of the entry to all printed lines.

- -Cl

  Left justify the data in each column.

- -Cr REPEATERS

  For GETBULK requests, REPEATERS specifies the max-repeaters value to use. For GETNEXT requests, REPEATERS specifies the number of entries to retrieve at a time.

- -Cw WIDTH

  Specifies the width of the lines when the table is printed. If the lines will be longer, the table will be printed in sections of at most WIDTH characters. If WIDTH is less than the length of the contents of a single column, then that single column will still be printed.

Examples

$ snmptable -v 2c -c public localhost at.atTable

SNMP table: at.atTable RFC1213-MIB::atTable

atIfIndex atPhysAddress atNetAddress

```
 1  8:0:20:20:0:ab 130.225.243.33
$ snmptable -v 2c -c public -Cf + localhost at.atTable
SNMP table: at.atTable
atIfIndex+atPhysAddress+atNetAddress 1+8:0:20:20:0:ab+130.225.243.33
$ snmptable localhost -Cl -CB -Ci -OX -Cb -Cc 16 -Cw 64 ifTable
SNMP table: ifTable
Index           Descr           Type            Mtu
Speed           PhysAddress     AdminStatus     OperStatus
LastChange      InOctets        InUcastPkts     InNUcastPkts
InDiscards      InErrors        InUnknownProtos OutOctets
OutUcastPkts    OutNUcastPkts   OutDiscards     OutErrors
OutQLen         Specific
index: [1]
1               lo              softwareLoopbac 16436
10000000                        up              up
?               2837283786      3052466         ?
0               0               ?               2837283786
3052466         ?               0               0
0               zeroDotZero
index: [2]
2               eth0            ethernetCsmacd  1500
10000000        0:5:5d:d1:f7:cf up              up
?               2052604234      44252973        ?
0               0               ?               149778187
65897282        ?               0               0
0               zeroDotZero
```

snmptest

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

snmptest [COMMON OPTIONS] AGENT

Description

snmptest is a flexible SNMP application that can monitor and manage information on a network entity.

After invoking the program, a command line interpreter proceeds to accept commands. This interpreter enables the user to send different types of SNMP requests to target agents.

AGENT identifies a target SNMP agent, which is instrumented to monitor the given objects. At its simplest, the AGENT specification will consist of a hostname or an IPv4 address. In this situation, the command will attempt communication with the agent, using UDP/IPv4 to port 161 of the given target host. See snmpcmd(1) for a full list of the possible formats for AGENT.

Once snmptest is invoked, the command line interpreter will prompt with:

```
Variable:
```

At this point you can enter one or more variable names, one per line. A blank line ends the parameter input and will send the request (variables entered) in a single packet, to the remote entity. Each variable name is given in the format specified in variables(5). For example:

```
snmptest -c public -v 1 zeus
Variable: system.sysDescr.0
Variable:
```

will return some information about the request and reply packets, as well as the information:

```
requestid 0x5992478A errstat 0x0 errindex 0x0
system.sysDescr.0 = STRING: "Unix 4.3BSD"
```

The errstatus value shows the error status code for the call. The possible values for errstat are in the header file snmp.h. The errindex value identifies the variable that has the given error. Index values are assigned to all the variables entered at the "Variable": prompt. The first value is assigned an index of 1.

Upon startup, the program defaults to sending a GET request packet. The type of request can be changed by typing one of the following commands at the "Variable:" prompt:

- $G - send a GET request
- $N - send a GETNEXT request
- $S - send a SET request
- $B - send a GETBULK request

GETBULK is not available in SNMPv1

- $I - send an Inform request
- $T - send an SNMPv2 Trap request

Other values that can be entered at the "Variable:" prompt are:

- $D - toggle the dumping of each sent and received packet
- $QP - toggle a quicker, less verbose output form
- $Q - Quit the program

Request Types:

- GET Request:

  When in "GET request" mode ($G or default), the user can enter an OID at the "Variable:" prompt. The user can enter multiple OIDs, one per prompt. The user enters a blank line to send the GET request.

- GETNEXT Request:

  The "GETNEXT request" mode ($N) is similar to the "Get request" mode, described above.

- SET Request:

  When in the "SET request" mode ($S), more information is requested by the prompt for each variable. The prompt:

  ```
  Type [i|s|x|d|n|o|t|a]:
  ```

  requests the type of the variable be entered. Depending on the type of value you want to set, you can type one of the following:

  i - integer

  u - unsigned integer

  s - octet string in ASCII

  x - octet string in hex bytes, separated by whitespace

  d - octet string as decimal bytes, separated by whitespace

  a - ip address in dotted IP notation

  o - object identifier

  n - null

  t - timeticks

  At this point a value will be prompted for:

  ```
  Value:
  ```

  If this is an integer value, just type the integer (in decimal). If it is a decimal string, type in white-space separated decimal numbers, one per byte of the string. Again type a blank line at the prompt for the variable name to send the packet.

- GETBULK Request:

The "GETBULK request" mode ($B) is similar to the "Set request" mode. GETBULK, however, is not available in SNMPv1.

- Inform Request:

  The "Inform request" mode ($I) is similar to the "Set request" mode. This type of request, however, is not available in SNMPv1. Also, the _agent_ specified on the snmptest command should correspond to the target snmptrapd agent.

- SNMPv2 Trap Request:

  The "SNMPv2 Trap Request" mode ($T) is similar to the "Set request" mode. This type of request, however, is not available in SNMPv1. Also, the _agent_ specified on the snmptest command should correspond to the target snmptrapd agent.

Options

- Common options

  See snmpcmd for a list of possible values for common options.

Examples

The following is an example of sending a GET request for two OIDs:

```
% snmptest -v 2c -c public testhost:9999


Variable: system.sysDescr.0
Variable: system.sysContact.0
Variable:
Received Get Response from 128.2.56.220
requestid 0x7D9FCD63 errstat 0x0 errindex 0x0
SNMPv2-MIB::sysDescr.0 = STRING: SunOS testhost 5.9 Generic_112233-02
sun4u
SNMPv2-MIB::sysContact.0 = STRING: x1111
```

The following is an example of sending a GETNEXT request:

```
Variable: SNMPv2-MIB::sysORUpTime
Variable:
Received Get Response from 128.2.56.220
requestid 0x7D9FCD64 errstat 0x0 errindex 0x0
SNMPv2-MIB::sysORUpTime.1 = Timeticks: (6) 0:00:00.06
Variable:
```

The following is an example of sending a SET request:

```
Variable: $S
Request type is Set Request
Variable: system.sysLocation.0
Type [i|u|s|x|d|n|o|t|a]: s
```

```
Value: building 17
Variable:
Received Get Response from 128.2.56.220
requestid 0x7D9FCD65 errstat 0x0 errindex 0x0
SNMPv2-MIB::sysLocation.0 = STRING: building A
Variable:
```

The following is an example of sending a GETBULK request:

```
Variable: $B
Request type is Bulk Request
Enter a blank line to terminate the list of non-repeaters
and to begin the repeating variables
Variable:
Now input the repeating variables
Variable: system.sysContact.0
Variable: system.sysLocation.0
Variable:
What repeat count? 2
Received Get Response from 128.2.56.220
requestid 0x2EA7942A errstat 0x0 errindex 0x0
SNMPv2-MIB::sysName.0 = STRING: testhost
SNMPv2-MIB::sysORLastChange.0 = Timeticks: (58) 0:00:00.58
SNMPv2-MIB::sysLocation.0 = STRING: bldg A
SNMPv2-MIB::sysORID.1 = OID: IF-MIB::ifMIB
Variable:
```

The following is an example of sending an Inform request:

```
snmptest -v 2c -c public snmptrapd_host
Variable: $I
Request type is Inform Request
(Are you sending to the right port?)
Variable: system.sysContact.0
Type [i|u|sIx|d|n|o|t|a]: s
Value: x12345
Variable:
Inform Acknowledged
Variable:
```

The snmptrapd_host will show:

```
snmptrapd_host [<ip address>]: Trap SNMPv2-MIB::sysContact.0 = STRING:
x12345
```

The following is an example of sending an SNMPv2 Trap request:

```
snmptest -v 2c -c public snmptrapd_host
Variable: $T
Request type is SNMPv2 Trap Request
(Are you sending to the right port?)
Variable: system.sysLocation.0
Type [i|u|s|x|d|n|o|t|a]: s
Value: building a
Variable:
```

The snmptrapd_host will show:

```
snmptrapd_host [<ip address>]: Trap SNMPv2-MIB::sys.0 = STRING:
building a
```

snmptranslate

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

snmptranslate [OPTIONS] OID [OID]...

Description

snmptranslate is an application that translates one or more SNMP object identifier values from their symbolic (textual) forms into their numerical forms (or vice versa).

OID is either a numeric or textual object identifier.

Options

- -D TOKEN[,...]

  Turn on debugging output for the given TOKEN(s). Try ALL for extremely verbose output.

- -h

  Display a brief usage message and then exit.

- -m MIBLIST

  Specifies a colon separated list of MIB modules to load for this application. This overrides the environment variable MIBS.

  The special keyword ALL is used to specify all modules in all directories when searching for MIB files. Every file whose name does not begin with "." will be parsed as if it were a MIB file.

- -M DIRLIST

  Specifies a colon separated list of directories to search for MIBs. This overrides the environment variable MIBDIRS.

- -T TRANSOPTS

  Provides control over the translation of the OID values. The following TRANSOPTS are available:

- -Td

  Print full details of the specified OID.

- -Tp

  Print a graphical tree, rooted at the specified OID.

- -Ta

  Dump the loaded MIB in a trivial form.

- -Tl

  Dump a labelled form of all objects.

- -To

  Dump a numeric form of all objects.

- -Ts

  Dump a symbolic form of all objects.

- -Tt

  Dump a tree form of the loaded MIBs (mostly useful for debugging).

- -Tz

  Dump a numeric and labelled form of all objects (compatible with MIB2SCHEMA format).

- -V

  Display version information for the application and then exit.

- -w WIDTH

  Specifies the width of -Tp and -Td output. The default is very large.

In addition to the above options, snmptranslate takes the OID input (-I), MIB parsing (-M) and OID output (-O) options described in the INPUT OPTIONS, MIB PARSING OPTIONS and OUTPUT OPTIONS sections of the snmpcmd(1) manual page.

Examples

* snmptranslate -On -IR seceders

will translate "sysDescr" to a more qualified form:

```
system.sysDescr
```

* snmptranslate -Onf -IR sysDescr

will translate "sysDecr" to:

```
.iso.org.dod.internet.mgmt.mib-2.system.sysDescr
```

* snmptranslate -Td -OS system.sysDescr

will translate "sysDecr" into:

```
SNMPv2-MIB::sysDescr
sysDescr OBJECT-TYPE
  -- FROM SNMPv2-MIB
  -- TEXTUAL CONVENTION DisplayString
  SYNTAX OCTET STRING (0..255)
  DISPLAY-HINT "255a"
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION "A textual description of the entity. This
               value should include the full name and
               version identification of the system's
               hardware type, software operating-system,
               and networking software."
::= { iso(1) org(3) dod(6) internet(1) mgmt(2) mib-2(1) system(1) 1 }
```

* snmptranslate -Tp -OS system

will print the following tree:

```
+--system(1)
   |
   +-- -R-- String    sysDescr(1)
   |        Textual Convention: DisplayString
   |        Size: 0..255
   +-- -R-- ObjID     sysObjectID(2)
   +-- -R-- TimeTicks sysUpTime(3)
   +-- -RW- String    sysContact(4)
```

```
|        Textual Convention: DisplayString
|        Size: 0..255
+-- -RW- String    sysName(5)
|        Textual Convention: DisplayString
|        Size: 0..255
+-- -RW- String    sysLocation(6)
|        Textual Convention: DisplayString
|        Size: 0..255
+-- -R-- Integer   sysServices(7)
+-- -R-- TimeTicks sysORLastChange(8)
|        Textual Convention: TimeStamp
|
+--sysORTable(9)
   |
   +--sysOREntry(1)
      |
      +-- ---- Integer   sysORIndex(1)
      +-- -R-- ObjID     sysORID(2)
      +-- -R-- String    sysORDescr(3)
      |        Textual Convention: DisplayString
      |        Size: 0..255
      +-- -R-- TimeTicks sysORUpTime(4)
               Textual Convention: TimeStamp
```

* snmptranslate -Ta | head

will produce the following dump:

```
dump DEFINITIONS ::= BEGIN
org ::= { iso 3 }
dod ::= { org 6 }
internet ::= { dod 1 }
directory ::= { internet 1 }
mgmt ::= { internet 2 }
experimental ::= { internet 3 }
private ::= { internet 4 }
security ::= { internet 5 }
snmpV2 ::= { internet 6 }
```

* snmptranslate -Tl | head

will produce the following dump:

```
.iso(1).org(3)
.iso(1).org(3).dod(6)
.iso(1).org(3).dod(6).internet(1)
.iso(1).org(3).dod(6).internet(1).directory(1)
.iso(1).org(3).dod(6).internet(1).mgmt(2)
.iso(1).org(3).dod(6).internet(1).mgmt(2).mib-2(1)
.iso(1).org(3).dod(6).internet(1).mgmt(2).mib-2(1).system(1)
.iso(1).org(3).dod(6).internet(1).mgmt(2).mib-
2(1).system(1).sysDescr(1)
.iso(1).org(3).dod(6).internet(1).mgmt(2).mib-2(1).system(1).sysOb-
jectID(2)
.iso(1).org(3).dod(6).internet(1).mgmt(2).mib-
2(1).system(1).sysUpTime(3)
```

* snmptranslate -To | head

will produce the following dump

```
.1.3
.1.3.6
.1.3.6.1
.1.3.6.1.1
.1.3.6.1.2
.1.3.6.1.2.1
.1.3.6.1.2.1.1
.1.3.6.1.2.1.1.1
.1.3.6.1.2.1.1.2
.1.3.6.1.2.1.1.3
```

* snmptranslate -Ts | head

will produce the following dump

```
.iso.org
.iso.org.dod
.iso.org.dod.internet
.iso.org.dod.internet.directory
.iso.org.dod.internet.mgmt
.iso.org.dod.internet.mgmt.mib-2
.iso.org.dod.internet.mgmt.mib-2.system
.iso.org.dod.internet.mgmt.mib-2.system.sysDescr
.iso.org.dod.internet.mgmt.mib-2.system.sysObjectID
```

```
.iso.org.dod.internet.mgmt.mib-2.system.sysUpTime
```

* snmptranslate -Tt | head

will produce the following dump

```
org(3) type=0
  dod(6) type=0
    internet(1) type=0
      directory(1) type=0
      mgmt(2) type=0
        mib-2(1) type=0
          system(1) type=0
            sysDescr(1) type=2 tc=4 hint=255a
            sysObjectID(2) type=1
            sysUpTime(3) type=8
```

snmptrap

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

```
snmptrap -v 1 [COMMON OPTIONS] [-Ci] enterprise-oid agent generic-trap
specific-trap uptime [OID TYPE VALUE]...

snmptrap -v [2c|3] [COMMON OPTIONS] [-Ci] uptime trap-oid [OID TYPE
VALUE]...

snmpinform -v [2c|3] [COMMON OPTIONS] uptime trap-oid [OID TYPE
VALUE]...
```

Description

snmptrap is an SNMP application that uses the SNMP TRAP operation to send information to a network manager. One or more object identifiers (OIDs) can be given as arguments on the

command line. A type and a value must accompany each object identifier. Each variable name is given in the format specified in variables(5).

When invoked as snmpinform, or when -Ci is added to the command line flags of snmptrap, it sends an INFORM-PDU, expecting a response from the trap receiver, retransmitting if required. Otherwise it sends an TRAP-PDU or TRAP2-PDU.

If any of the required version 1 parameters, enterprise-oid, agent, and uptime are specified as empty, it defaults to 1.3.6.1.4.1.3.1.1 (enterprises.cmu.1.1), hostname, and host-uptime respectively.

The TYPE is a single character, one of:

i - integer

c - counter 32

u - unsigned integer

s - octet string in ASCII

x - octet string in hex bytes, separated by whitespace

d - octet string as decimal bytes, separated by whitespace

a - ip address in dotted IP notation

o - object identifier

b - bits

n - null

t - timeticks

which are handled in the same way as the snmpset command.

For example:

```
snmptrap -v 1 -c public manager enterprises.spider test-hub 3 0 ''
interfaces.iftable.ifentry.ifindex.1 i 1
```

will send a generic linkUp trap to manager, for interface 1.

Options

- Common options

  See snmpcmd for a list of possible values for common options.

- -Ci.

snmpusm

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a

User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

```
snmpusm [COMMON OPTIONS] create USER [CLONEFROM-USER]
snmpusm [COMMON OPTIONS] delete USER
snmpusm [COMMON OPTIONS] cloneFrom USER CLONEFROM-USER
snmpusm [COMMON OPTIONS] [-Ca] [-Cx] passwd OLD-PASSPHRASE NEW-PASSPHRASE [USER]
snmpusm [COMMON OPTIONS] <-Ca | -Cx> -Ck passwd OLD-KEY-OR-PASSPHRASE NEW-KEY-OR-PASSPHRASE [USER]
snmpusm [COMMON OPTIONS] [-Ca] [-Cx] changekey [USER]
```

Description

snmpusm is an SNMP application that can be used to do simple maintenance on the users known to an SNMP agent, by manipulating the agent's User-based Security Module (USM) table. The user needs write access to the usmUserTable MIB table. This tool can be used to create, delete, clone, and change the passphrase of users configured on a running SNMP agent.

Options

- Common options

  See snmpcmd for a list of possible values for common options.

- -CE ENGINE-ID

  Set usmUserEngineID to be used as part of the index of the usmUserTable. Default is to use the contextEngineID (set via -E or probed) as the usmUserEngineID.

- -Cp STRING

  Set the usmUserPublic value of the (new) user to the specified STRING.

Options for the passwd and changekey commands:

- -Ca

  Change the authentication key.

- -Cx

  Change the privacy key.

- -Ck

Allows to use localized key (must start with 0x) instead of passphrase. When this option is used, either the -Ca or -Cx option (but not both) must also be used.

Creating Users

An unauthenticated SNMPv3 user can be created using the command

```
snmpusm [OPTIONS] create USER
```

This constructs an (inactive) entry in the usmUserTable, with no authentication or privacy settings. In principle, this user should be useable for 'noAuthNoPriv' requests, but in practise the Net-SNMP agent will not allow such an entry to be made active.

In order to activate this entry, it is necessary to "clone" an existing user, using the command

```
snmpusm [OPTIONS] cloneFrom USER CLONEFROM-USER
```

The USER entry then inherits the same authentication and privacy settings (including pass phrases) as the CLONEFROM user.

These two steps can be combined into one, by using the command

```
snmpusm [OPTIONS] create USER CLONEFROM-USER
```

The two forms of the create sub-command require that the user being created does not already exist. The cloneFrom sub-command requires that the user being cloned to does already exist.

Cloning is the only way to specify which authentication and privacy protocols to use for a given user, and it is only possible to do this once. Subsequent attempts to reclone onto the same user will appear to succeed, but will be silently ignored. This (somewhat unexpected) behaviour is mandated by the SNMPv3 USM specifications (RFC 3414). To change the authentication and privacy settings for a given user, it is necessary to delete and recreate the user entry. This is not necessary for simply changing the pass phrases (see below). This means that the agent must be initialized with at least one user for each combination of authentication and privacy protocols. See the snmpd.conf(5) manual page for details of the createUser configuration directive.

Deleting Users

A user can be deleted from the usmUserTable using the command

```
snmpusm [OPTIONS] delete USER
```

Changing Password Phrases

User profiles contain private keys that are never transmitted over the wire in clear text (regardless of whether the administration requests are encrypted or not). To change the secret key for a user, it is necessary to specify the user's old passphrase as well as the new one. This uses the command

```
snmpusm [OPTIONS] [-Ca] [-Cx] passwd OLD-PASSPHRASE NEW-PASSPHRASE
[USER]
```

After cloning a new user entry from the appropriate template, you should immediately change the new user's passphrase.

If USER is not specified, this command will change the passphrase of the (SNMPv3) user issuing the command. If the -Ca or -Cx options are specified, then only the authentication or

privacy keys are changed. If these options are not specified, then both the authentication and privacy keys are changed.

```
snmpusm [OPTIONS] [-Ca] [-Cx] changekey [USER]
```

This command changes the key in a perfect-forward-secrecy compliant way through a diffie-helman exchange. The remote agent must support the SNMP-USM-DH-OBJECTS-MIB for this command to work. The resulting keys are printed to the console and may be then set in future command invocations using the --defAuthLocalizedKey and --defPrivLocalizedKey options or in your snmp.conf file using the defAuthLocalizedKey and defPrivLocalizedKey keywords.

Since these keys are randomly generated based on a diffie helman exchange, they are no longer derived from a more easily typed password. They are, however, much more secure.

To change from a localized key back to a password, the following variant of the passwd sub-command is used:

```
snmpusm [OPTIONS] <-Ca | -Cx> -Ck passwd OLD-KEY-OR-PASSPHRASE NEW-
KEY-OR-PASSPHRASE [USER]
```

Either the -Ca or the -Cx option must be specified. The OLD-KEY-OR-PASSPHRASE and/or NEW-KEY-OR-PASSPHRASE arguments can either be a passphrase or a localized key starting with "0x", e.g. as printed out by the changekey sub-command.

Examples

Let's assume for our examples that the following VACM and USM configurations lines were in the snmpd.conf file for a Net-SNMP agent. These lines set up a default user called "initial" with the authentication passphrase "setup_passphrase" so that we can perform the initial set up of an agent:

```
# VACM configuration entries
rwuser initial
# lets add the new user we'll create too:
rwuser wes
# USM configuration entries
createUser initial MD5 setup_passphrase DES
```

The "initial" user's setup should be removed after creating a real user that you grant administrative privileges to.

passphrases must have a minimum length of 8 characters.

Create a new user

```
snmpusm -v3 -u initial -n "" -l authNoPriv -a MD5 -A setup_passphrase
localhost create wes initial
```

Creates a new user, here named "wes" using the user "initial" to do it. "wes" is cloned from "initial" in the process, so he inherits that user's passphrase ("setup_passphrase").

Change the user's passphrase

```
snmpusm -v 3 -u wes -n "" -l authNoPriv -a MD5 -A setup_passphrase
localhost passwd setup_passphrase new_passphrase
```

After creating the user "wes" with the same passphrase as the "initial" user, we need to change his passphrase for him. The above command changes it from "setup_passphrase", which was inherited from the initial user, to "new_passphrase".

Test the new user

```
snmpget -v 3 -u wes -n "" -l authNoPriv -a MD5 -A new_passphrase
localhost sysUpTime.0
```

If the above commands were successful, this command should have properly performed an authenticated SNMPv3 GET request to the agent.

Now, go remove the vacm "group" snmpd.conf entry for the "initial" user and you have a valid user 'wes' that you can use for future transactions instead of initial.

Manipulating the usmUserTable using this command can only be done using SNMPv3. This command will not work with the community-based versions, even if they have write access to the table.

snmpvacm

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

```
snmpvacm [COMMON OPTIONS] createSec2Group MODEL SECURITYNAME GROUPNAME
snmpvacm [COMMON OPTIONS] deleteSec2Group MODEL SECURITYNAME
snmpvacm [COMMON OPTIONS] createView [-Ce] NAME SUBTREE MASK
snmpvacm [COMMON OPTIONS] deleteView NAME SUBTREE
```

```
snmpvacm [COMMON OPTIONS] createAccess GROUPNAME [CONTEXTPREFIX] MODEL
LEVEL CONTEXTMATCH READVIEW WRITEVIEW NOTIFYVIEW

snmpvacm [COMMON OPTIONS] deleteAccess GROUPNAME [CONTEXTPREFIX] MODEL
LEVEL

snmpvacm [COMMON OPTIONS] createAuth GROUPNAME [CONTEXTPREFIX] MODEL
LEVEL AUTHTYPE CONTEXTMATCH VIEW

snmpvacm [COMMON OPTIONS] deleteAuth GROUPNAME [CONTEXTPREFIX] MODEL
LEVEL AUTHTYPE
```

Description

snmpvacm is an SNMP application that can be used to do simple maintenance on the View-based Control Module (VACM) tables of an SNMP agent. The SNMPv3 VACM specifications (see RFC2575) define assorted tables to specify groups of users, MIB views, and authorized access settings. These snmpvacm commands effectively create or delete rows in the appropriate one of these tables, and match the equivalent configure directives which are documented in the snmpd.conf(5) man page.

Sub-Commands

```
createSec2Group MODEL SECURITYNAME GROUPNAME
```

Create an entry in the SNMPv3 security name to group table. This table allows a single access control entry to be applied to a number of users (or 'principals'), and is indexed by the security model and security name values.

- MODEL, An integer representing the security model, taking one of the following values:
  - 1 - reserved for SNMPv1
  - 2 - reserved for SNMPv2c
  - 3 - User-based Security Model (USM)
- SECURITYNAME, A string representing the security name for a principal (represented in a security-model-independent format). For USM-based requests, the security name is the same as the username.
- GROUPNAME, A string identifying the group that this entry (i.e. security name/model pair) should belong to. This group name will then be referenced in the access table (see createAccess below).

```
deleteSec2Group MODEL SECURITYNAME
```

Delete an entry from the SNMPv3 security name to group table, thus removing access control settings for the given principal. The entry to be removed is indexed by the MODEL and SECURITYNAME values, which should match those used in the corresponding createSec2Group command (or equivalent).

```
createView [-Ce] NAME SUBTREE MASK
```

Create an entry in the SNMPv3 MIB view table. A MIB view consists of a family of view subtrees which may be individually included in or (occasionally) excluded from the view.

Each view subtree is defined by a combination of an OID subtree together with a bit string mask. The view table is indexed by the view name and subtree OID values.

- [-Ce], an optional flag to indicate that this view subtree should be excluded from the named view. If not specified, the default is to include the subtree in the view. When constructing a view from a mixture of included and excluded subtrees, the excluded subtrees should be defined first - particularly if the named view is already referenced in one or more access entries.
- NAME, a string identifying a particular MIB view, of which this OID subtree/mask forms part (possibly the only part).
- SUBTREE, the OID defining the root of the subtree to add to (or exclude from) the named view.
- MASK, a bit mask indicating which sub-identifiers of the associated subtree OID should be regarded as significant.

```
deleteView NAME SUBTREE
```

Delete an entry from the SNMPv3 view table, thus removing the subtree from the given MIB view. Removing the final (or only) subtree will result in the deletion of the view. The entry to be removed is indexed by the NAME and SUBTREE values, which should match those used in the corresponding createView command (or equivalent).

When removing subtrees from a mixed view (i.e. containing both included and excluded subtrees), the included subtrees should be removed first.

```
createAccess GROUPNAME [CONTEXTPREFIX] MODEL LEVEL CONTEXTMATCH
READVIEW WRITEVIEW NOTIFYVIEW
```

Create an entry in the SNMPv3 access table, thus allowing a certain level of access to particular MIB views for the principals in the specified group (given suitable security model and levels in the request). The access table is indexed by the group name, context prefix, security model and security level values.

- GROUPNAME, the name of the group that this access entry applies to (as set up by a createSec2Group command, or equivalent)
- CONTEXTPREFIX, a string representing a context name (or collection of context names) which this access entry applies to. The interpretation of this string depends on the value of the CONTEXTMATCH field (see below).

  If omitted, this will default to the null context "".
- MODEL, an integer representing the security model, taking one of the following values:
  - 1 - reserved for SNMPv1
  - 2 - reserved for SNMPv2c
  - 3 - User-based Security Model (USM)
- LEVEL, an integer representing the minimal security level, taking one of the following values:
  - 1 - noAuthNoPriv
  - 2 - authNoPriv

- 3 - authPriv

  This access entry will be applied to requests of this level or higher (where authPriv is higher than authNoPriv which is in turn higher than noAuthNoPriv).

- CONTEXTMATCH, indicates how to interpret the CONTEXTPREFIX value. If this field has the value '1' (representing 'exact') then the context name of a request must match the CONTEXTPREFIX value exactly for this access entry to be applicable to that request.

  If this field has the value '2' (representing 'prefix') then the initial substring of the context name of a request must match the CONTEXTPREFIX value for this access entry to be applicable to that request. This provides a simple form of wildcarding.

- READVIEW, the name of the MIB view (as set up by createView or equivalent) defining the MIB objects for which this request may request the current values.

  If there is no view with this name, then read access is not granted.

- WRITEVIEW, the name of the MIB view (as set up by createView or equivalent) defining the MIB objects for which this request may potentially SET new values.

  If there is no view with this name, then read access is not granted.

- NOTIFYVIEW, the name of the MIB view (as set up by createView or equivalent) defining the MIB objects which may be included in notification request.

This aspect of access control is not currently supported.

```
deleteAccess GROUPNAME [CONTEXTPREFIX] MODEL LEVEL
```

Delete an entry from the SNMPv3 access table, thus removing the specified access control settings. The entry to be removed is indexed by the group name, context prefix, security model and security level values, which should match those used in the corresponding createAccess command (or equivalent).

```
createAuth GROUPNAME [CONTEXTPREFIX] MODEL LEVEL AUTHTYPE CONTEXTMATCH
VIEW
```

Create an entry in the Net-SNMP extension to the standard access table, thus allowing a certain type of access to the MIB view for the principals in the specified group. The interpretation of GROUPNAME, CONTEXTPREFIX, MODEL, LEVEL and CONTEXTMATCH are the same as for the createAccess directive. The extension access table is indexed by the group name, context prefix, security model, security level and authtype values.

- AUTHTYPE, the style of access that this entry should be applied to. See snmpd.conf(5) and snmptrapd.conf(5) for details of valid tokens.

- VIEW, the name of the MIB view (as set up by createView or equivalent) defining the MIB objects for which this style of access is authorized.

```
deleteAuth GROUPNAME [CONTEXTPREFIX] MODEL LEVEL AUTHTYPE
```

Delete an entry from the extension access table, thus removing the specified access control settings. The entry to be removed is indexed by the group name, context prefix, security

model, security level and authtype values, which should match those used in the corresponding createAuth command (or equivalent).

Examples

Given a pre-existing user dave (which could be set up using the snmpusm(1) command), we could configure full read-write access to the whole OID tree using the commands:

```
snmpvacm localhost createSec2Group 3 dave RWGroup
snmpvacm localhost createView all .1 80
snmpvacm localhost createAccess RWGroup 3 1 1 all all none
```

This creates a new security group named "RWGroup" containing the SNMPv3 user "dave", a new view "all" containing the full OID tree based on .iso(1) , and then allows those users in the group "RWGroup" (i.e. "dave") both read- and write-access to the view "all" (i.e. the full OID tree) when using authenticated SNMPv3 requests.

As a second example, we could set up read-only access to a portion of the OID tree using the commands:

```
snmpvacm localhost createSec2Group 3 wes ROGroup
snmpvacm localhost createView sysView system fe
snmpvacm localhost createAccess ROGroup 3 0 1 sysView none none
```

This creates a new security group named "ROGroup" containing the (pre-existing) user "wes", a new view "sysView" containing just the OID tree based on .iso(1).org(3).dod(6).inet(1).mgmt(2).mib-2(1).system(1) , and then allows those users in the group "ROGroup" (i.e. "wes") read-access, but not write-access to the view "sysView" (i.e. the system group).

Exit Status

The following exit values are returned:

- 0 - Successful completion
- 1 - A usage syntax error (which displays a suitable usage message) or a request timeout.
- 2 - An error occurred while executing the command (which also displays a suitable error message).

Limitations

- This utility does not support the configuration of new community strings, so is only of use for setting up new access control for SNMPv3 requests. It can be used to amend the access settings for existing community strings, but not to set up new ones.
- The use of numeric parameters for secLevel and contextMatch parameters is less than intuitive. These commands do not provide the full flexibility of the equivalent config file directives.
- There is (currently) no equivalent to the one-shot configure directives rouser and rwuser.

snmpwalk

Location	*entuity_home*\lib\tools
Type	Third party utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

This utility and documentation is provided according to its license terms, which can be viewed under *entuity_home*\licenseTerms\Net-SNMP.

Syntax

```
snmpwalk [APPLICATION OPTIONS] [COMMON OPTIONS] [OID]
```

Description

snmpwalk is an SNMP application that uses SNMP GETNEXT requests to query a network entity for a tree of information.

An object identifier (OID) may be given on the command line. This OID specifies which portion of the object identifier space will be searched using GETNEXT requests. All variables in the subtree below the given OID are queried and their values presented to the user. Each variable name is given in the format specified in variables(5).

If no OID argument is present, snmpwalk will search the subtree rooted at SNMPv2-SMI::mib-2 (including any MIB object values from other MIB modules, that are defined as lying within this subtree). If the network entity has an error processing the request packet, an error packet will be returned and a message will be shown, helping to pinpoint why the request was malformed.

If the tree search causes attempts to search beyond the end of the MIB, the message "End of MIB" will be displayed.

Options

- Common options

  See snmpcmd for a list of possible values for common options.

- -Cc, do not check whether the returned OIDs are increasing. Some agents (LaserJets are an example) return OIDs out of order, but can complete the walk anyway. Other agents return OIDs that are out of order and can cause snmpwalk to loop indefinitely. By default, snmpwalk tries to detect this behavior and warns you when it hits an agent acting illegally. Use -Cc to turn off this check.
- -Ci, include the given OID in the search range. Normally snmpwalk uses GETNEXT requests starting with the OID you specified and returns all results in the MIB subtree

rooted at that OID. Sometimes, you may wish to include the OID specified on the command line in the printed results if it is a valid OID in the tree itself. This option lets you do this explicitly.

- -Cl, in fact, the given OID will be retrieved automatically if the main subtree walk returns no useable values. This allows a walk of a single instance to behave as generally expected, and return the specified instance value. This option turns off this final GET request, so a walk of a single instance will return nothing.
- -Cp, upon completion of the walk, print the number of variables found.
- -Ct, upon completion of the walk, print the total wall-clock time it took to collect the data (in seconds). Note that the timer is started just before the beginning of the data request series and stopped just after it finishes. Most importantly, this means that it does not include snmp library initialization, shutdown, argument processing, and any other overhead.

Example

The command:

```
snmpwalk -Os -c public -v 1 zeus system
```

will retrieve all of the variables under system:

```
sysDescr.0 = STRING: "SunOS zeus.net.cmu.edu 4.1.3_U1 1 sun4m"
sysObjectID.0 = OID: enterprises.hp.nm.hpsystem.10.1.1
sysUpTime.0 = Timeticks: (155274552) 17 days, 23:19:05
sysContact.0 = STRING: ""
sysName.0 = STRING: "zeus.net.cmu.edu"
sysLocation.0 = STRING: ""
sysServices.0 = INTEGER: 72
```

start

Location	*entuity_home*\bin
Type	Process,
Invoked By	Command line
User Invocation	Command line
Invoked Processes	n/a
Configured Through	Command line
Log File	n/a

Syntax

```
start database
```

Description

This command starts the Entuity database server `mysqld` in readiness for a `restore` from the previous `backup`.

In Windows `start` is also the name of a Windows command. To use `start` specify the full path:

```
C:\Entuity\bin\start database
```

See Also

`stop`.

starteye

Location	*entuity_home*\bin
Type	Process,
Invoked By	startup
User Invocation	Command line
Invoked Processes	Entuity processes
Configured Through	n/a
Log File	n/a

Syntax

```
starteye
```

Description

`starteye` starts and monitors processes specified in `startup_o/s.cfg`. When a process stops starteye attempts to re-start the process, if four re-start attempts fail then `starteye` shuts down Entuity.

Files

`entuity.cfg` (see *Chapter 3 - Entuity System Files*), and `/top/start.log`.

See Also

`stopeye`.

starteotssvr

Location	*entuity_home*\bin
Type	Process,
Invoked By	`starteye`
User Invocation	Windows Service

Invoked Processes	n/a
Configured Through	`startup_WIN32.cfg`
Log File	*entuity_home*`\log\starteyesvr.log.[1..4]`

Description

This process is a Windows service that controls the starting and continued running of processes specified through `startup_WIN32.cfg`. When `starteotssvr` fails to restart a process four times within five minutes then Entuity is shutdown.

Logs

Messages are written to the file `systemcontrol.log` in the *entuity_home*`\log` directory.

On UNIX and Linux system administrators should replicate `starteotssvr` by defining a chron job that starts the processes specified in startup_*o*/*s*.cfg.

stop

Location	*entuity_home*`\bin`
Type	Process,
Invoked By	n/a
User Invocation	Command line
Invoked Processes	Context dependant
Configured Through	n/a
Log File	n/a

Syntax

```
stop database
```

Description

This command stops the Entuity database server `mysqld` following a `restore` from the previous `backup`.

See Also

`start`.

stopeye

Location	*entuity_home*`\bin`
Type	Process,
Invoked By	n/a

User Invocation	Command line
Invoked Processes	Entuity processes
Configured Through	n/a
Log File	n/a

Syntax

```
stopeye
```

Description

The `stopeye` script stops the:

- web server
- scheduler
- database
- license server.

The prompt returns when Entuity is successfully shutdown.

Files

`entuity.cfg` (see *Chapter 3 - Entuity System Files*), and `\tmp\start.log`.

See Also

`starteye`.

stpman

Location	*entuity_home*\bin
Type	Process, run daily, at 05:15
Invoked By	provost
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Log File	*entuity_home*\log\stpman.log.[1..4]

Description

When you have the Device News (Device Network Early Warning System) module installed, is responsible for gathering STP (Spanning Tree Protocol)-related information from each switch and hub in the network. The information is gathered using SNMP, and includes the root switch, STP port status (blocking, forwarding, etc.), and STP timers.

swmaint

Location	*entuity_home*\bin
Type	Utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	None
Configured Through	Command line parameters
Log File	No

Syntax

```
swmaint[-c <connection string>] [-d <days>] [-force] [-n] [-o] [-q]
[-s] [-v]
```

Description

`swmaint` removes inconsistencies between StormWorks objects, associations and streams. It can also delete stale objects and optimize database tables, so incorporate `swmaint` into your Entuity housekeeping process. When restoring an Entuity database you should also always run `swmaint` before restarting the Entuity server.

You must not run `swmaint` when Entuity is polling your network, otherwise it will corrupt your database. Only run `swmaint` when the Entuity database is the only Entuity process running.

Options

- **-c**, database connection string. When `swmaint` is run from the Entuityserver it is not required. The database connection string has the format:
  `HOST=<host>;UID=<user>;PWD=<password>;DB=<database>;PORT=<port>`
- **-d**, objects that are considered stale for more days than this value are deleted. On long running systems the number of stale objects can impact database performance. By default `swmaint` does not delete stale objects.
- **-force**, continue `swmaint` even when the previous run failed or the Entuity server is running.

-force may result in data loss or corruption.

- **-n**, no update. `swmaint` does not modify the database but does report on the state of the database. To view the number of stale objects you must always use this setting with **-d**.
- **-o**, deletes object data with incomplete StormWorks associations (the default). This option is only useful with **-q**.
- **-p**, optimize database tables. This calls the database command to optimize each table, and may take sometime.

- **-q**, quick mode. Quick mode does not delete or optimize object and sample data (dso_.. and dss_.. . You should use quick mode when wanting to quickly restart Entuity's management of your network.
- **-s**, deletes sample data with incomplete associations (the default). This option is only useful with **-q**.
- **-v**, verbose mode provides a full set of progress messages.

Examples

This example optimizes the database, deletes StormWorks objects with incomplete associations and delivers a full set of progress messages:

```
swmaint -v
```

This example reports on the number of stale objects that have been in the system more than seven days:

```
swmaint -n -d 7
```

This removes all stale objects from the database:

```
swmaint -d 0
```

sysLogger

Location	*entuity_home*\bin
Type	Process,
Invoked By	starteots
User Invocation	n/a
Invoked Processes	None
Configured Through	*entuity_home*\etc\startup_0/S.cfg entuity.cfg
Log File	*entuity_home*\log\syslogger.log.[1..4]

Description

Invoked during system startup and continues to run until the system closes. It receives device syslog messages, discards those from devices not managed by Entuity and forwards to the Event Viewer as events those it does. sysLogger uses the Entuity database to identify the device and possibly add additional information, e.g. CPU utilization, buffer capacity and mismatches in protocol.

Through the syslogger section of entuity.cfg you can use *replaceEventDetailsAction* to replace problematic characters from the event details.

ticker

Location	*entuity_home*\bin
Type	Process

Invoked By	`starteots`
User Invocation	n/a
Invoked Processes	None
Configured Through	*entuity_home*\`etc\startup_O/S.cfg`
Log File	*entuity_home*\`log\ticker.log.[1..4]`

Description

This is the `ticker` process. It is a daemon process, invoked during software startup, that allows you to view real time output at the device and port level, viewing data changes as they occur.

trapsplit

Location	*entuity_home*\`bin`
Type	Process
Invoked By	Command line, `starteots`
User Invocation	Yes
Invoked Processes	None
Configured Through	Command line, trapsplit configuration file, startup_o/s.cfg,
Log File	*entuity_home*\log\trapsplit.log.[1..4]

Syntax

trapsplit [-p portnumber] [-l logfilename] configfilename

Description

This is a daemon process that can be started by the System Administrator. It is a trap receiver that forwards traps on to a user specified list of recipient hosts on user-definable UDP ports.

Listens for SNMP traps on UDP port 162. It then forwards the traps to one or more ports specified through the configuration file.

- **-p** *portnumber* is the UDP port on which trapsplit listens for traps. The default is UDP port 162. To amend the port, for example to listen on port 2162, enter:

  ```
  trapsplit -p 2162 trapconfig.cfg
  ```

- **-l** *logfilename* enables logging and specifies the name and path of the trapsplit log file. By default logging is not enabled. To output the trapsplit messages to the Entuity log folder enter:

  ```
  trapsplit -l ..\log\trapsplit.log trapconfig.cfg
  ```

- *configfilename* is the trapsplit configuration file. Each entry should be on a separate line and have the format:

```
host [port]
```

Where:

- *host* specifies the destination host, either the hostname or IP address.
- *port* specifies the destination port. When not entered the default is UDP port 162.

updateNames

Location	*entuity_home*\bin
Type	Utility
Invoked By	n/a
User Invocation	Process, Command line
Invoked Processes	None
Configured Through	Command line parameters
Log File	*entuity_home*\log\updateNames.log

Syntax

`updateNames`.

Description

Entuity distinguishes between the *Polled Name*/*IP address* it uses to manage a device and the *Display Name* it displays for you to identify the device.

Display Name can be derived from **Polled Name** / **IP Address**, **System Name**, **Resolved Name**, **Resolved Name (fully qualified)** and **IP Address**, i.e. the source of the name is external to Entuity and derived either from SNMP or DNS. If this external value changes then `updateNames` updates *Display Name*.

`updateNames` compares the device *Display Name* in Entuity against the value on the device. If there is a difference `updateNames` updates the *Display Name.* However if the new name clashes with an existing name Entuity appends its *Device ID* in brackets after it. If this would make the name longer than the maximum name length (59 characters) then the original name is shortened prior to appending the *Device ID*.

If a device display name includes its *Device ID*, `updateNames` does not remove it, even when it is no longer required to make the device name unique.

`updateNames` is scheduled, by default, to run at 03:00 every day. It can be disabled through a setting in `entuity.cfg`:

```
[updateNames]
disabled=true
```

`updateNames` can also be run from the command line from `/bin/updateNames`.

`updateNames.log` records changes to device inventory and also when `updateNames` is run.

vendinfo

Location	*entuity_home*\lib\tools
Type	Utility
Invoked By	n/a
User Invocation	Command line
Invoked Processes	None
Configured Through	Command line parameters
Log File	No

Description

`vendinfo` identifies the vendor device support datasets available to Entuity and the decisions made when more than one vendor file is available for a particular sysoid; which device support dataset Entuity uses to manage that device type (as identified through its sysoid).

Each device support dataset is associated with a specific device sysoid. Where there:

- Is only one available device support dataset for a given sysoid, Entuity uses that dataset when managing a device with that sysoid.
- Are two or more device support datasets for a given sysoid, Entuity uses the dataset with the highest priority.

Datasets are available through four types of vendor files, all have a `.vendor` extension. These vendor files are, listed in ascending order of priority:

1) uncertified device definitions in *entuity_home*\`etc\uncertified` folder when Entuity discovers devices with sysoids for which there is not a device support dataset. These generic device support datasets should be considered temporary definitions, and only used until Entuity supply an appropriate vendor file.

2) `bin.vendor`, which is installed to *entuity_home*\`etc`. It contains multiple device support datasets, many of which are also listed in their individual vanilla vendor files. `bin.vendor` has the second lowest priority when Entuity is determining the source of device information.

   Device support datasets in `bin.vendor` have the second lowest priority when Entuity is determining which of those available to use to manage a device type.

3) vanilla vendor files are installed to *entuity_home*\`etc` during Entuity installation and configuration.

   Device support datasets in vanilla vendor files have the second highest priority when Entuity is determining which vendor device definition to use to manage a device type.

4) exotica vendor files are installed to *entuity_home*\`etc\exotica`. Exotica files are only

used by Entuity when they are copied to *entuity_home*\etc, either manually or during Entuity configuration, e.g. when selecting a module.

Device support datasets in exotica vendor files have the highest priority when Entuity is determining which vendor device definition to use to manage a device type.These files use a simple naming convention, using the vanilla filename, with a plus sign in the filename and identifying name, e.g. SOLSERV+managed Host.vendor.

Entuity does not make operational use of vendor files from the etc\exotica; these files are primarily reference resources. Entuity only uses vendor files in the active configuration directory, by default *entuity_home*\etc, when determining how to manage a device type.

vendinfo Switches

vendinfo is supplied with a number of case sensitive switches, that you can use individually, or combine to investigate vendor information:

- -e *directory*, instructs vendinfo to consider device support datasets in the specified folder as though they are in the active configuration folder, by default *entuity_home*\etc directory. You must specify the full path e.g.

  ```
  vendinfo -e c:\entuity\etc -I 1.3.6.1.4.1.42.2.1.1
  ```

- -V *directory*, instructs vendinfo to consider device support datasets in the specified folder as though they are vanilla vendor files. You must specify the full path e.g.

  ```
  vendinfo -V c:\entuity_resources\vanilla -I 1.3.6.1.4.1.42.2.1.1
  ```

- -E *directory*, instructs vendinfo to consider device support datasets in the specified folder as though they are in the exotica vendor file reference folder, by default *entuity_home*\etc\exotica directory. You must specify the full path e.g.

  ```
  vendinfo -E c:\entuity\etc\exotica -I 1.3.6.1.4.1.42.2.1.1
  ```

- -B *directory*, instructs vendinfo to take the specified folder as the root folder for relative path folders specified with other switches, e.g.:

  ```
  vendinfo -B c:\entuity -e etc -I 1.3.6.1.4.1.42.2.1.1
  ```

- -H *directory*, instructs vendinfo to take the specified folder as the root folder. Unlike the -B switch you do not need to specify a path to the exotica folder, e.g.:

  ```
  vendinfo -H c:\entuity -I 1.3.6.1.4.1.42.2.1.1
  ```

- -n *filename*, forces vendinfo to use newbin.vendor format when reading the specified file (newbin.vendor is a deprecated file):

  ```
  vendinfo -n c:\entuity\etc\newbin.vendor
  ```

- -c *filename*, forces vendinfo to use classic.vendor format when reading the specified file:

  ```
  vendinfo -c c:\entuity\etc\bin.vendor
  ```

- -x *prefix*, exclude data for sysoids starting with the entered prefix:

  ```
  vendinfo -H c:\entuity -x 1.3.6.1.4.1.9 -x 1.3.6.1.4.1.42
  ```

- -X *sysoid*, excludes data for the entered sysoid:

  ```
  vendinfo -H c:\entuity -X 1.3.6.1.4.1.42.2.1.1 -X 1.3.6.1.4.1.9.1.8
  ```

- -i *prefix*, includes data for sysoids starting with the entered prefix:

  ```
  vendinfo -H c:\entuity -x 1.3.6.1.4.1.42  -x 1.3.6.1.4.1.9
  ```

- -I *sysoid*, allows you to specify the particular sysoid in which you are interested:

  ```
  vendinfo -H c:\entuity -I 1.3.6.1.4.1.42.2.1.1
  ```

- -m, restricts `vendinfo` output to sysoids for devices currently under Entuity management.

  ```
  vendinfo -H c:\entuity -m
  ```

- -q, restricts `vendinfo` output to sysoids with concerns or questionable status. This is useful when investigating the current status of your system's device support datasets.

  ```
  vendinfo -H c:\entuity -q
  ```

- -h, displays command line help.

  ```
  vendinfo -h
  ```

- -u, displays command line help.

  ```
  vendinfo -u
  ```

- -v, displays `vendinfo` version number:

  ```
  vendinfo version 1.7    [@(#)buildstamp.h $Revision: 6.48 $]
  ```

Understanding the Results

In this example output, `vendinfo` is flagging a concern about the provenance of an operational device support dataset. This was most likely a consequence of mistakenly moving, rather than copying, an exotica device support file from *entuity_home*`\etc\exotica` to *entuity_home*`\etc`.

```
lib\tools\vendinfo -q
795 datasets read from 188 files ( 12 null files, 332 others):
c:\Entuity\TRUNKref30a\etc
25 datasets read from  25 files (  5 null files,   2 others):
c:\Entuity\TRUNKref30a\etc\exotica
.1.3.6.1.4.1.42.2.1.1          -          -       -
?provenance        winner     1.3     etc\SOLSERV+managedHost.vendor
variation=1        loser      1.3     etc\SOLSERV.vendor
variation=1        loser      1.3     etc\bin.vendor
variation=1         reference  1.3     etc\SOLSERV.vendor
```

When you run `vendinfo` it returns a report on device support datasets it has processed:

- exotica and uncertified vendor files contain one dataset each, `bin.vendor` contains multiple datasets.
- Null files are old, deprecated vendor files that no longer contain vendor definitions. They are supplied to prevent older Entuity installations continuing to use these definitions.
- Others, are files in the *entuity_home*`\etc` and *entuity_home*`\etc\exotica` folders that do not have the vendor extension and so Entuity, and `vendinfo` do not consider them as

device support files.

The results for each sysoid all have the same format:

sysoid

VendorStatus Variation=n ResultStatus VersionNumber PathName

where:

- *sysoid* identifies the sysoid to which the subsequent vendor information relates.
- *VendorStatus* indicates the status of the vendor file, and can be:
  - **?Provenance**, indicates a winner, or loser, entry in `etc\` does not have a matching reference dataset, i.e. in *entuity_home*`\etc\exotica`. This does not necessarily indicate an immediate operational problem, only that it may indicate a problem in maintaining reference file information.
  - **?fluke**, indicates you need to check the vendor files in *entuity_home*`\etc` for competing vendor definitions from the same reference folder. For example, you may have copied from the *entuity_home*`\etc\exotica` to *entuity_home*`\etc` two Nokia3.8.1-build28 firewall definitions. Entuity cannot determine which you want to use to manage your devices, and so selects one on the basis of their filename's ASCII alphabetic values.
  - **?version**, indicates vendor files with the same name have different operational characteristics. You should investigate that the correct vendor file is in use and ensure all vendor files with the same name have the same device definition.
  - **?Name**, indicates vendor files with different names have the same operational characteristics. You should investigate that the correct vendor file is in use and ensure all vendor files with the same definition have the same filename.
  - **?rootName**, indicates a deviation from the supplied naming convention. You must not amend vendor filenames as Entuity uses the naming convention when determining which vendor definition to use to manage a device.
  - **?wrongDir**, indicates vanilla or exotica file definitions are in the wrong folder, e.g. a vanilla vendor file is in the exotica folder.

When `vendinfo` is only run against one folder, *VendorStatus* indicators that rely on comparisons across folders, e.g. **?Provenance**, are not meaningful.

- *Variation=n*, is only used where there is more than one vendor entry that would yield different operational behavior for the sysoid. Vendor definitions with the same variation value would exhibit the same operational behavior.
- *ResultStatus* can be:
  - **winner**, the device support dataset identified as being the highest ranked available in *entuity_home*`\etc` for that sysoid.
  - **loser**, a device support dataset for which there is another higher ranked dataset available in *entuity_home*`\etc` for that sysoid.
  - **reference**, device support datasets that are not in operational use but held in the

resource folder *entuity_home*\etc\exotica. Usually for every winner and loser there is an equivalent reference file for that sysoid.

- **alternate,** is applied to entries from etc\exotica whose behavior would not match any winner or loser from etc for the current sysoid.

- *VersionNumber* is an internal, non-mandatory Entuity reference number. Different version numbers between two files does not necessarily indicate differences in the vendor definition information.
- *PathName*, indicates the name and location of the file holding the vendor information.

viewserver

Location	*entuity_home*\bin
Type	Process
Invoked By	starteots
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Log File	*entuity_home*\log\viewserver.log.[1..4]

Description

By default event management process uses the internal Entuity mechanism, viewserver for view membership checks. viewserver checks object-view and content filter settings, by default every twenty minutes, or when a view is amended.

vipman

Location	*entuity_home*\bin
Type	Process, run at 19:00 and 02:00
Invoked By	provost
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Log File	*entuity_home*\log\vipman.log.[1..4]

Description

It is responsible for ascertaining which ports in the network are deemed to be infrastructure ports, i. e.

- router ports
- trunk ports (i.e. ports connecting switches together)
- uplinks (i.e. ports connecting routers with switches).

Entuity uses three methods to identify trunk ports; through the MIB, by counting the number of MAC addresses on the port and then identifying whether there are associated VLANs and lastly through Cisco's CDP trunk discovery protocol. Through the VIPMAN Trunk Promote module you can also identify to vipman ports you want Entuity to manage as trunk ports.

vtpDomainTool

Location	*entuity_home*\lib\tools
Type	Utility,
Invoked By	`provost`, user
User Invocation	User, Command line
Invoked Processes	n/a
Configured Through	provost.conf
Log File	

Syntax

```
vtpDomainTool [-c] [-d] [-h] [-p] [-b]
```

where:

- -c, deletes the Regional by VTP view
- -d, sets debug logging level
- -h, displays help information
- -p, preserves user tags
- -b, preserves blank domain.

Description

`vtpDomainTool` automatically assigns aliases for use in Entuity, enabling Entuity to distinguish between VLANs that have the same name but are members of a different VTP domain. The VLAN alias is built by combining the VTP Domain Name with the VLAN name. `vtpDomainTool` also generates a view called **Regional by VTP**, which shows devices and VLANs grouped by VTP domain name.

`vtpDomainTool` can be run from the command line, or scheduled and run by `provost`. It uses information collected by `vtpman` to identify devices and VLANs, and their correct VTP domains. To maintain the accuracy of the view, you should schedule `vtpDomainTool` to run after `vtpman` has completed. Scheduling is set through `provost.conf`, for example:

```
job vtpDomainTool {
 count 1, start @06:15:00, repeat forever, interval 24h, command
'${entuity_home}/lib/tools/vtpDomainTool'
}
```

Changes to configuration files are not maintained after upgrading Entuity, and so `VTPDomainTool` would have to be rescheduled in `provost.conf`.

vtpman

Location	*entuity_home*\bin
Type	Process
Invoked By	`provost,` run daily, at 05:15
User Invocation	n/a
Invoked Processes	n/a
Configured Through	n/a
Log File	*entuity_home*\log\vtpman.log.[1..4]

Description

`vtpman` is responsible for gathering VTP (VLAN Trunking Protocol)-related information from each switch in the network. The information is gathered using SNMP, and includes the VTP server, VTP domain name, and pruning status.

3 Entuity System Files

This section describes the main system files used in the Entuity environment. These files should not be moved, deleted or modified unless otherwise stated.

Directory names are given in Linux/Unix format. The names still apply if you are a Windows user, reverse the slashes to enter them in DOS format.

bin.vendor

Location

entuity_home/`etc`

Format

Internal use only.

Description

Contains MIB-related information for each networking vendor supported by the Entuity environment. The file is used by various SNMP polling processes, including `prole`.

This MIB information is also detailed in individual device type vendor files, installed by default to *entuity_home*/`etc`. Additional device type definitions, not detailed in `bin.vendor` are held in *entuity_home*/`etc/exotica`. Device definitions held in these folders are only used by Entuity, when you copy them to *entuity_home*/`etc`. Also `proliferate` can generate new device types, called Unclassified, and these are held in *entuity_home*\`etc\uncertified`. (See the *Entuity User and System Administrator Guide*.)

Through `vendinfo` you can check the current status of your device vendor files. `vendinfo` identifies the vendor device support datasets available to Entuity and the decisions made when more than one vendor file is available for a particular sysoid; which device support dataset Entuity uses to manage that device type (as identified through its sysoid).

Status

Read-only.

Device File (Seed File)

Location

User defined location and name. Historically this import device file was known as `dev.txt` and was expected in *entuity_home*\`etc`.

Format

Text file containing lines in two possible formats. The older format which only applies when adding SNMPv1 and SNMPv2 devices is:

```
# comment line
device-name     community-string      #optional comment
```

The recommended format supports SNMP1v1/v2 and SNMPv3 devices, for example:

- SNMP1v1/v2:

  `-d jupiter -D jupiter -l full -c public`

- SNMPv3:

  `-d` *10.44.2.44* `-u` *paul* `-a MD5 -A` *xyy1232h* `-x DES -X` *fgdgg34g*

Description

The device file is also known as the seed file, it contains instructions used by `proliferate` when adding devices to Entuity, e.g. device identifiers, authentication details, SNMP version.

A device file can be created by:

- System Administrators who specify in it the list of devices they want to import to Entuity through the Inventory Administration Import Devices function.

  Historically this import device file was known as `dev.txt` and was expected in *entuity_home*\`etc`, however both name and location are user definable.

- Entuity, specifically as part of `autoDiscovery`. It is then used by `proliferate` to add devices to the Entuity database, i.e. it contains the same list of devices and options as displayed through the Inventory Administration Inventory Candidates page.

  This file is called `autodisc.txt` and is located in *entuity_home*`/etc/deviceFiles` .

Devices can either be referred to by an IP address or a host name. Host names should either be added to the local `/etc/hosts` file, or be present within the DNS (Domain Name System). Once a device is added to the Entuity management environment, it continues to be referenced by the name specified in the device file.This is an example extract from a device file using the new format for an SNMPv1c/v2 device:

```
-d 10.44.1.40 -c public # sysoid ".1.3.6.1.4.1.9.1.716" sysDescr
"Cisco IOS Software, C2960 Software (C2960-LANBASE-M), Version
12.2(25)FX, RELEASE SOFTWARE (fc1) Copyright (c) 1986-2005 by Cisco
Systems, Inc. Compiled Wed 12-Oct-05 22:05 by yenanh".
```

where:

- *#*, indicates the subsequent text on that line is a comment. Comments can inform you:
  - That the device is already managed by Entuity using another interface.
  - Of the current device's IP address, sysoid and system description.
  - Of a device that could not be managed.
- *-d*, indicates the following value is the device name.

- *-c*, indicates the following value is the device community string.

This is an example extract from a device file, using the new format for an SNMPv3 device:

```
-d 10.44.2.44 -u paul -a MD5 -A xyy1232h -x DES -X fgdgg34g
```

For SNMPv3 devices the format is:

```
-d <deviceIdentifier> -u <UserName> -a MD5 -A <Auth passwd> -x DES -X
<Privacy passwd>
```

where:

- *-d*, indicates the following value is the device name.
- *deviceIdentifier* is the management interface on hubs and switches, and a single interface on a router.

  You should be able to resolve each of the device names into an IP address on the Entuity server using one of the following methods:

  - Static hosts file (e.g. `\etc\hosts`).
  - NIS (Network Information System) or NIS+ .
  - DNS (Domain Name System).

  This resolution is not required if the device identifier is itself the IP address of the device. The choice of identifier is important as it is the primary method of identifying devices in Entuity.
- *-u <UserName>*, requires a valid user name to access the device.
- *-a* MD5, sets the authentication protocol, valid values are MD5 (Message-Digest algorithm 5), SHA (Secure Hash Algorithm).
- *-A <Auth passwd>*, sets the authentication password, valid values must be between eight and thirty-two characters long. If the password contains spaces double quotes must be placed around the password.
- *-x* DES, sets the privacy protocol, valid values are DES (Data Encryption Standard), AES.
- *-X <Privacy passwd>*, sets the privacy password, valid values must be between eight and thirty-two characters long. If the password contains spaces double quotes must be placed around the password.

It is `proliferate` that adds devices to Entuity and so the switches used within the device file configure `proliferate`.

Adding VM Platforms

Entuity manages VM platforms through their SDK which necessitates a different set of connection attributes to other device types. To specify a VM platform the format is:

```
-d IpAddress -l manLevel -w type,url,user,password -T deviceType
```

where:

- -d *IpAddress*, identifies the device name or IP address.
- -l *manLevel*, must be set to the management level **web.**

- -w sets the web connection details, which must be comma delimited and entered in this order:
  - *type*, enter **2** for a VMware ESXi or **3** for an Oracle VM platform.
  - *url*, the url to the VM platform's SDK.
  - *user*, user account Entuity uses to access the SDK.
  - *password*, user account password.
- -T, sets the device to the internal Entuity identifier for a VM platform, i.e. 1144.

For example to add the VM platform blade to Entuity you can enter:

```
-d blade -l web -w 2,https://blade/sdk,devuser,232neree -T 1144
```

Status

Created and maintained by the System Administrator, name and location are user definable. Also created each time `autoDiscovery` runs, being saved to *entuity_home*/`etc/deviceFiles` as `autodisc.txt`.

entuity.cfg

Location

entuity_home/`etc`

Format

Text file containing lines in the format: *systemVariable*=*value*, under headings denoted by square brackets [].

`entuity.cfg` is white space sensitive, therefore do not, for example, enter spaces at the start of a line or before or after the equals sign.

Description

This file holds the key information about the Entuity configuration. You must only use the configuration procedure described in the *Entuity Getting Started Guide* to reconfigure the software.

Do not directly modify the parameters in `entuity.cfg`, Entuity cannot be held responsible for the consequences. If you want to further amend these settings contact your Entuity Support representative.

Status

Maintained by the System Administrator.

entuity.cfg Sections

Within `entuity.cfg` related parameters are grouped together within sections, for example:

```
[autodiscovery]
config=${ENTUITY_HOME}${FPS}etc${FPS}autodisc.cfg
automatic=1
```

where:

- *[autodiscovery]* is the section header for autoDiscovery, identified as it is placed between square brackets.
- *config* and *automatic* are autoDiscovery parameters.

Parameters must follow the correct section headings to have the required effect. Therefore variable names must only be unique within a section, e.g. config is used in a number of different sections.

Following is a list of some of the sections and parameters available within `entuity.cfg`. If you require changes to the default settings please contact your Entuity representative.

[]

Most parameters are held within sections that relate to particular Entuity functionality. These parameters are of a more general application and are in the first section of `entuity.cfg` (it has the square brackets that denote a section, but no section name):

- *activeuser* is the user login used to start Entuity.
- *alternatelicensefile* is the location and name of alternate Entuity license files. You can specify a comma delimited list of license files.
- *auditLogKeepTime* is the keep time for audit log entries, by default set to 60 days.
- *configured*, indicates whether Entuity is configured, **1**, or not **0**.
- *dbconfigured* indicates whether the Entuity database is configured, **1**, or not **0**.
- *dbdir* is the directory containing the database (typically, *entuity_home*`/database`).
- *dbportnum* is the port number used by the database server (typically, 3306).
- *destination* is the directory into which the software was copied (i.e. *entuity_home*).
- *devicefile* is the master device file used by Entuity, by default `dev.txt`.
- *eosretrysnmp* is the number of times Entuity attempts to make an SNMP connection, by default 5. Each retry timeout value is the same, derived from *eostimeoutsnmp*.
- *eostimeoutsnmp* is the time in milliseconds Entuity waits for a response from a device before considering it a timeout, by default 1500.
- *etcdir* is the location of the directory which contains the active configuration files, by default *entuity_home*`\etc`.
- *fps* holds the correct slash, (forward or backward) for your operating system and is placed into default file paths given `entuity.cfg`.

- *hostname* must be the valid hostname of the Entuity server. If wrongly set then enter the correct value here, or if appropriate reset the value in the server host file.
- *installed*, indicates whether Entuity `install` successfully completed, **1**, or not, **0**.
- *installtime*, time the Entuity server was installed.
- *installid*, the unique Entuity server identifier. In multi Entuity server environments it is used to distinguish one Entuity server from another. Also third party integrations may use it as part of the URL to access an Entuity server.
- *Licensefile* is the location of the Entuity license file.
- *logdir* is the directory containing the log files (typically, /log).
- *macttl* is the time to live of a MAC address discovered by the `provost` scheduled `macman`. By default set to 7, i.e. seven days after last polled on the device Entuity removes it.
- *mallocArenaMax* is a Linux specific configuration setting. It sets the maximum number of arenas available for allocation to Entuity threads. By default Entuity limits the number of arenas to 16:

  ```
  mallocArenaMax=16
  ```

  In multi-core environments with appropriate memory resources you can increase the number of arenas and improve Entuity performance. Linux arenas are allocated memory in, as a minimum, 64mb chunks.

- *snmpMaxPduSize* limits the length of SNMP request packets, by default PDU length is set to 1408:

  ```
  snmpMaxPduSize=1408
  ```

  You can configure Entuity so it does not limit PDU size, however some devices may report over length packets as too big or silently ignore them. To set PDU size to unrestricted set:

  ```
  snmpMaxPduSize=0
  ```

- *snmpMaxPduSizeOverridesfile* sets the name of the file, by default `snmpMaxPduOverrides.cfg`, containing sysoids with the maximum PDU size for devices with that sysoid.(See *snmpMaxPduOverrides.cfg*.)

  You can amend the name of the PDU override configuration file, useful when a customer wants to add their own override values and preserve them during upgrades:

  ```
  snmpMaxPduSizeOverridesfile=snmpMaxPDUoveride.cfg
  ```

- *snmpVlanContextPrefix* is for use with SNMPv3 devices configured to provide VLAN information using an SNMPv3 context. When you have configured these devices Entuity can convert any characters in the SNMP v1/v2c community string into SNMPv3 context by comparing the community string provided in the SNMP request with the stored community string. Any difference, excluding a leading **@,** is appended to a string, by default **vlan-**. You can change the **vlan-** prefix by setting

  ```
  snmpVlanContextPrefix=cVLAN-
  ```

- *source* is the directory from which the software was copied (i.e. the CD-ROM directory).

- *StartupProperties=-Djava.rmi.dummy=dummy*
- *trapportnum* is the port used for receiving SNMP traps, by default port 162.
- *trendconfigured=0*
- *version* is the Entuity software version number.
- *webportnum* is the port number used by the web server (typically, 80).

[auditlog]

Parameters in this section are applicable to the audit log. The default is:

```
[auditlog]
rowlimit=1000
```

where:

- *rowlimit* determines the maximum number of log entries displayed through the Audit Log page, by default set to 1000.

[AuthLog]

Parameters in this section are applicable to the login authorization log file. The default is:

```
[AuthLog]
FailureOnly=0
```

where:

- *FailureOnly* is set to:
  - 0, all login events are recorded in `auth.log`.
  - 1, only when login fails are events recorded in `auth.log`.

[autodiscovery]

Parameters in this section are applicable to autoDiscovery:

- *automatic* when set to:
  - 0, autoDiscovery is not automatically started. When it is already running manually then this value is ignored. When it is already running automatically then autoDiscovery is stopped.
  - 1, autoDiscovery runs each Sunday at 01:00 hours. autoDiscovery uses the specified configuration file. Where the file does not exist, autoDiscovery searches for devices on the network(s) to which the current host is attached.

During the configuration of Entuity if you created your device file using autoDiscovery *automatic* is set to 1, otherwise it is set to 0.

- *config* holds the path and name of the default autoDiscovery configuration file, *entuity_home*`/etc/autodisc.cfg`.
- *duplicateIpCheck* when set to:
  - 1, autoDiscovery checks that discovered devices do not have the same IP address as

devices already under management. Entuity hides devices with duplicate IP addresses from the list of candidate devices, you can view them by selecting **show devices already in inventory**.

- 0 (default), autoDiscovery displays in the Inventory Candidates page devices with duplicate IP addresses to those already under management. When you attempt to add them to Entuity, Entuity reports them as already under management and does not add then again.

- *suppressNotRecognized*, controls how `autoDiscovery` handles unrecognized device types. When *suppressNotRecognized* is set to:
  - 0 (default), autoDiscovery adds unrecognized device types to Entuity as non-classified devices.
  - 1, autoDiscovery does not add unrecognized device types to Entuity.

[AvailabilityMonitor]

Parameters in this section are applicable to the Availability Monitor, and all Entuity functions that use ping.

```
[AvailabilityMonitor]
maxReportedEffectedItems=32
maxThreads=256
pingMaxTTL=32
pingTimeout=5
ignoreIfType=59,60,70
```

Where:

- *maxReportedEffectedItems*, restricts the total number of impacted items that can be displayed for an event to, by default, 32. On Network Outage events calling from the context menu Impacted Items, Entuity can only display Nodes, Applications and Servers up to this maximum.
- *maxThreads* is the maximum number of concurrent traceroute threads, default 256.
- *pingMaxTTL* is the maximum ICMP TTL to use, default 32.
- *pingTimeout* is the maximum number of pings to a device that Entuity sends before timing out, default 5.0.
- *ignoreIfType*, instructs Availability Monitor to ignore interfaces of the specified type. These are detailed in *Appendix C - Port Interface Types*.
- *ignorevirtualaddress* when set to:
  - **0**, `applicationMonitor` would ping HSRP virtual IP addresses. These pinged addresses would be included when Entuity is determining the state of a device, or performing root cause analysis, with potentially misleading consequences.
  - **1** (default), `applicationMonitor` does not ping HSRP virtual addresses. For newly added devices there would be a short period between a port being taken under management and its IP address being recognized as an HSRP virtual IP address.

- *loglevel*, level of error reporting written to the log file, i.e. **error**, **warning**, **info**, **debug** and **all**.
- *resetstatsinterval*, sets the reporting period of availability statistics used for, for example, SLA reporting. by default this is hourly, i.e. 3600 seconds.
- *tracecoreinterval*, (default 120 seconds) controls how frequently Availability Monitor attempts to do full traceroutes to non-edge ip addresses and shortcut traceroutes to edge ip addresses.

[bem]

Parameters in this section are applicable to forwarding events and incidents to BMC TrueSight Infrastructure Management Server:

```
[bem]
connection_username=admin
connection_view=All Objects
consolidation_server_name=entlonppvm01
consolidation_server_web_port=81
```

Where:

- *connection_username* is the Entuity user account used to access the Entuity server from the associated event or incident URL available from the BMC TrueSight Infrastructure Management Server.
- *connection_view* is the Entuity view used to access the Entuity server from the associated event or incident URL available from the BMC TrueSight Infrastructure Management Server.

Which events and incidents are forwarded to the BMC TrueSight Infrastructure Management Server is determined by the conditions added to rules or triggers. The *connection_username* and *connection_view* settings must allow access to the data associated with those events and incidents for the associated URL to succeed. For example **admin** and **All Objects** provide access to all managed objects on a server, however Entuity support recommend using a non-administrator account.

- *consolidation_server_name* is the resolved name of the Entuity consolidation server that you want to use to access the event or incident data. This replaces the name of the Entuity server that actually raised the event or incident.
- *consolidation_server_web_port* is the port number of the Entuity consolidation server that you want to use to access the event or incident data. By default it is port 80.

You can use *consolidation_server_name* and *consolidation_server_web_port* if you are not using a consolidation server but you have configured the Entuity server forwarding events and incidents with a non-default web port.

[bemSender]

Parameters in this section are applicable to forwarding events and incidents to BMC TrueSight Infrastructure Management Server:

```
[bemSender]
MaxSendingThreads=1
EventQueueSize=10000
```

Where:

- *MaxSendingThreads* should not be amended. It is set to meet the BMC Impact Manager multi-threading requirements.
- *EventQueueSize* sets the maximum size of the sending tasks queue waiting to be processed by the BMC TrueSight Infrastructure Management Server sender. The default value is 10000.

[configurationmanager]

Parameters in this section are applicable to the Entuity Configuration Management module. The default parameter values are:

```
[configurationmanager]
historyrowlimit=1000
jobhistorykeeptime=30
jobhistoryupdateinterval=10
jobmonitorflushtime=36000
jobmonitorsleepinterval=10
jobmonitorsleepstartup=5
subjobhistoryupdateinterval=10
```

where:

- *historyrowlimit* determines the maximum number of entries displayed through the Job History page, by default set to 1000.
- *jobhistorykeeptime* sets the number of days to retain the history of a job, by default 30 days. `jobmonitor` runs each day at 00:10 and deletes the history of jobs that ran more than the set number of days ago.
- *jobhistoryupdateinterval* sets the number of seconds between updates of an open Job History page, by default 10 seconds.
- *jobmonitorflushtime* sets the number of seconds since the last response from a designated Script Engine after which Entuity will stop requesting a job update from that Script Engine. By default this is 10 hours. Entuity does not update the state of a job that was in progress, although through the log file the Script Engine will be marked as ignored.
- *jobmonitorsleepinterval* sets the number of seconds between `jobmonitor` queries of the Scrip Engine for the states of sub-jobs, by default 10 seconds.

- *jobmonitorsleepstartup* sets a delay in the startup of `jobmonitor` subsequent to the startup of TomCat. By default this delay is 5 seconds and should not be amended.
- *subjobhistoryupdateinterval* sets the number of seconds between updates of an open sub job history dialog, by default every 10 seconds.

[database]

Parameters in this section are used when configuring Entuity's database. This is the default setting:

```
[database]
key_buffer=192M
```

Where:

- *key_buffer* defines the size of the buffer that holds details of recently used keys. On large sites, and where the Entuity server machine has available resources, performance can be improved by increasing the size of the key buffer.

[datastream]

Parameters in this section are used when configuring `StormWorks`. These are the default settings and must not be amended:

```
[datastream]
connection=HOST=127.0.0.1;UID=root;PWD=;DB=DSALPHA; PORT=${dbportnum}
```

Where:

- *connection* defines the link to the `StormWorks` database and
  - *HOST* is the IP address of the machine holding the database.
  - *UID* is the database login.
  - *PWD* is the database password.
  - *DB* is the database.
  - *PORT* is the default database port, usually 3306.

[devdefunct]

`devDefunct` removes devices from Entuity that have aged out. `devDefunct` is configured through:

- *ageout,* the number of days after which a device is deemed to be defunct and can be removed via the daily run `devDefunct`. When a value is not entered `devDefunct` does not delete any devices. This is the default state.

[discovery]

By default the details of newly added devices and ports are given priority in the discovery queue. When you do not want to interrupt Entuity's normal discovery cycle, you can turn off the priority setting through:

```
[discovery]
noPrioritiseNewInProliferate=1
noPrioritiseNewInGUI=1
noRefreshViewMapInProliferate=0
HostNameFormat=Qualified
```

where:

- *noPrioritiseNewInProliferate* when set to 1 does not move devices and ports newly added using autodiscovery, to the top of the discovery queue.
- *noPrioritiseNewInGUI* when set to 1 does not move devices and ports newly added through the web interface, to the top of the discovery queue.
- *noRefreshViewMapInProliferate* when set to:
  - **0** (default), changes made from the web UI to the devices Entuity manages trigger a refresh of the underlying object map used by the Entuity web interface.
  - **1**, changes made from the web UI to the devices Entuity manages do not trigger a refresh of the underlying object map used by the Entuity web interface. The changes are only visible after the next refresh.

The length of time it takes to refresh the object map partly depends upon the size of the managed network. As there is a overhead to regenerating the map, `proliferate` only allows a queue of two refresh requests.

- *HostNameFormat* determines the device name used by Entuity when adding a device through auto discovery. When it is set to:
  - **Mixed** (default), discovery uses the qualified DNS name when possible. When the name is too long, over 59 characters, then Entuity uses the unqualified name and when that is not available Entuity uses the device IP address.
  - **Qualified**, discovery uses the qualified DNS name when possible. When the name is too long, over 59 characters, then Entuity uses the unqualified name and if that is not available Entuity uses the device IP address.
  - **Unqualified**, discovery uses the unqualified DNS name and when that is not available the device IP address.
  - **IpAddress**, Entuity uses the device IP address.

When adding devices using a seed file, Entuity uses the device name as it appears in the file.

[diskmonitor]

Parameters in this section configure `diskMonitor` which monitors the available disk space on the Entuity server. This is an example section:

```
[diskmonitor]
sample_period=60
```

```
message_period=600
message_threshold=200
shutdown_threshold=100
```

where:

- *sample_period* is the period in seconds between monitoring of the disk space. The default is 60, i.e. disk space is measured every minute.
- *message_period* is the interval, in seconds, between `diskMonitor` generating disk space low warning events that appear on Event Viewer. The default is 600, i.e. an event is generated every ten minutes when disk space reaches the *messagethreshold*.
- *message_threshold* is compared to the available disk space. When that value falls below the *messagethreshold* `diskMonitor` generates a disk space warning event. The default value, is 200Mb, setting it to 0 turns off this feature.
- *shutdown_threshold* is compared to the available disk space. When that value falls below the *shutdownthreshold* `diskMonitor` initiates Entuity shutdown. The default value is 100Mb, setting it to 0 turns off this feature.

On UNIX systems the disk space value is unreliable for NFS partitions. When Entuity and its database are on different machines disabling `diskMonitor` is recommended.

[dns]

The parameter in this section configures frequency of hostname resolution.

```
[dns]
positivestaletime=86400
```

where:

- *positivestaletime*, determines how long Entuity retains resolved IP address and hostname information in both memory and the database, by default 86400 seconds (twenty-four hours). It therefore also determines how quickly Entuity identifies a change in hostname resolution.

[dnsproxy]

Parameters in this section configure `dnsproxy`.

```
[dnsproxy]
maxCacheSize=10000
```

Where:

- *maxCacheSize* is the number of entries in the DNS cache for each zone.

[Events]

Parameters in this section extend Entuity functionality.

```
[Events]
```

```
engineIdOverwrite=a2fed1312070f4dcc9eb2b483318ef317
portEventsForDevices=false
excludeGiants=1
enableDeviceUnreachableEvents=1
SnmpTimeoutFilterByReachability=1
jmxserver_port=12122
jmxFile=eventEngineJmxUrl.jmx
licenseLowWarningThreshold=100
# RPC timeout for calls into DsKernel
swRpcTimeout=60
# Maximum number of threads executing external processes
processExecutorMaxCount=4
# Number of threads processing events in parallel
workerMaxCount=10
# Number of times event can be derived or forwarded between event
engines
maxEventProcessingDepth=10
# Number of seconds between e-mails to the same address
emailThrottlingPeriodSec=300
# For how long to store events
dbKeepDays=14
dbPartitionDurationHours=24
# Receiver settings
receiverPort=19194
receiverHostname=localhost
receiverBacklog=10
receiverThreads=10
receiverTimeout=60
# Request listener settings
requestListenerPort=19193
requestListenerHostname=localhost
requestListenerBacklog=10
requestListenerThreads=10
requestListenerTimeout=60
requestListenerEventsInBatch=100
# Delete expired event suppression rules
deleteExpiredEventSuppressionsPeriodSeconds=0
```

Where:

- *engineIdOverwrite* is an hexadecimal string that when defined would override the default engine ID used by the Entuity server when forwarding SNMPv3 traps. The default engine ID is derived from the Entuity `server.id`. You may want to override the default string when there is a conflict with another device's engine id.

  Engine ID is represented by a hexadecimal string including just 0-9 and A-F. It must be at least 5 bytes long but no more than 32 bytes. If you enter a string with an invalid length or one that contains invalid characters Entuity records the error in *entuity_home*`\log\groovyEvents.log`.

  engineIdOverwrite can have one of these formats:

  - `xxxxxxxxxx`, no separator.
  - `xx:xx:xx:xx:xx`, separated by colon.
  - `xx xx xx xx xx`, separated by space.
- *portEventsForDevices* when set to:
  - **true**, events raised against a port contribute to the event status of its device.
  - **false** (default), events raised against a port do not contribute to the event status of its device.
- *excludeGiants* when set to:
  - **1** (default) excludes giants from error calculations, therefore Packet Corruption events cannot be raised by giants. When excluded Entuity writes to `prodigy.log` Excluding Giants, (`prodigy` calculates packet corruption errors).
  - **0**, giants are included as part of error calculations.
- *enableDeviceUnreachableEvents* controls when Entuity raises the Device Reachability Degraded, Device Unreachable and Device Unreachable Cleared events and the Device Reachability incident. The Network Outage event is independent of this parameter.

  When set to:

  - **1** Entuity raises the Device Reachability Degraded event when the device is the root cause of the network outage, and Device Unreachable when the device is unavailable but not the root cause.
  - **0** (default) the device unreachable events and incident are not configured.
- *licenseLowWarningThreshold* sets the threshold for the number Entuity Server License Alert event. By default when there are fewer than 100 device or object credits available Entuity raises the event.
- *SnmpTimeoutFilterByReachability*, controls how Entuity manages SNMP Agent Not Responding events. When set to:
  - **1** (default), SNMP Agent Not Responding events are only generated when the device is reachable.
  - **0**, SNMP Agent Not Responding events are generated regardless of whether Entuity can reach the device. With this setting Entuity does not generate the clearing SNMP Agent Responding events.
  - **mix**, allows generation of SNMP Agent Not Responding events regardless of whether

the device is reachable by Entuity. It also raises the clearing SNMP Agent Responding events. This setting is for test purposes only.

- *deleteExpiredEventSuppressionsPeriodSeconds* when set to:
  - 0 (default), Entuity does not automatically delete expired event suppression rules.
  - 1 or greater Entuity does delete expired event suppression rules.

  Entuity does not delete a rule as it expires but instead regularly checks for expired event suppression rules. This value sets, in seconds, the period between those checks.

  Entuity does enforce a lower boundary of one hour (3600 seconds) between checks for expired events which guards against this action becoming too resource intensive. For example if you enter a value of 2 Entuity checks every hour and not every 2 seconds. Entuity takes the server start time as the starting point of its event suppression period.

  Entuity records as a separate entry in the audit log each deleted expired event suppression rule with *User* set to **System**.

[eyepoller]

These parameters control configuration of eyepoller. Misconfiguration of some eyepoller parameters can result in poor Entuity performance, including missing polling of data. Always consult with Entuity Support before amending the eyepoller configuration.

Changes to the polling frequency must always be multiples of five minutes for the polled data to meaningfully integrate with the Entuity roll-up processes.

```
[eyepoller]
pollerEventsEnable=1
workers=25
backlog=2
timeSkewTolPercent=2.0
timeSkewTolAbsSecs=5.0
wrapDetectionMarginSecsCrit=2.0
wrapDetectionMarginSecsWarn=5
disableEventGrouping=0
fetchUpdatesRetryLimit=5
fetchUpdatesItemsPerReq=100
```

Where:

- *pollerEventsEnable*, controls whether these events which report on the efficacy of eyepoller, are enabled or disabled:
  - Device Port(s) Utilization Accuracy Lost
  - Device Port(s) Utilization Accuracy At Risk
  - Device Clock Inconsistency
  - Device Port(s) Utilization Missed Due to Slow Response.

When set to:

- **1**, (default), Entuity can raise events that indicate problems with eyepoller.
- **0**, Entuity cannot raise events that indicate problems with eyepoller. The only indication of problems with eyepoller would be when data is missing from the managed object's history.

- *workers*, the maximum number of working threads eyepoller can use. Too few threads and eyepoller may not have enough time to complete all of its polling, too many and resources on the server may not be sufficient.

  By default *workers* is set to 25, valid values range from 1 to 500.

  Do not amend this setting unless specifically advised to do so by your Entuity Support contact.

- *backlog*, influences creation of additional eyepoller work threads. By default set to 2, while valid values range from 1 to 5.

  Do not amend this setting unless specifically advised to do so by your Entuity Support contact.

- *timeSkewTolPercent*, the proportional setting for the tolerated difference between the poll interval as measured by device sysUpTime and poll interval as measured by the Entuity server system clock. When the clocks differ by a proportion greater than *timeSkewTolPercent* plus *timeSkewTolAbsSecs* Entuity raises a Device Clock Inconsistency (when it is enabled) and discards the polled sample.

  By default *timeSkewTolPercent* is set to 2.0, while valid values range from 0.0 to 20.0.

- *timeSkewTolAbsSecs*, the fixed value, in seconds, for the tolerated difference between the poll interval as measured by device sysUpTime and poll interval as measured by the Entuity server system clock. When the clocks differ by a proportion greater than *timeSkewTolPercent* plus *timeSkewTolAbsSecs* Entuity raises a Device Clock Inconsistency (when it is enabled) and discards the polled sample.

  A lower tolerance level implies more sensitive checking, which could also lead to a greater number of Device Clock Inconsistency events (when enabled).

  By default *timeSkewTolAbsSecs* is set to 5.0, while valid values range from 0.0 to 30.0.

- *wrapDetectionMarginSecsCrit*, sets the margin, in seconds, for Entuity to identify potential undetected 32 bit counter wraps as the interval between pollings is too great. When the margin threshold is crossed Entuity:
  - Discards the polled data, resulting in a gap in the history data for the managed object
  - Raises a Device Port(s) Utilization Accuracy Lost event (when it is enabled).

  By default *wrapDetectionMarginSecsCrit* is set to 2.0, while valid values range from 0.0 to 10.0. A larger margin implies more sensitive checking, and potentially more discarded samples and more Device Port(s) Utilization Accuracy Lost events (when enabled).

- *wrapDetectionMarginSecsWarn*, sets the margin, in seconds, for Entuity to identify potential undetected 32 bit counter wraps as the interval between pollings is too great.

When the margin threshold is crossed Entuity raises a Device Port(s) Utilization Accuracy At Risk event (when it is enabled).

By default *wrapDetectionMarginSecsWarn* is set to 5.0, while valid values range from 0.0 to 10.0. A larger margin implies more sensitive checking, and potentially the raising of more Device Port(s) Utilization Accuracy At Risk events (when enabled).

- *disableEventGrouping*, controls whether polling problem events are raised against the device or the port. When set to:
  - 0 (default), events associated with eyepoller are raised against the device
  - 1, polling problem events are raised at the port level. Only use this setting under guidance from Entuity Support as the consequences are likely to be a great increase in events.
- *fetchUpdatesRetryLimit*, controls the number of attempts `eyepoller` makes to obtain polling duty updates from dskernel, before abandoning the attempt.
- By default *fetchUpdatesRetryLimit* is set to 5, while valid values range from 0 to 20.

  Do not amend this setting unless specifically advised to do so by your Entuity Support contact.
- *fetchUpdatesItemsPerReq*, determines the maximum amount of data per response when eyepoller is requesting polling duty updates from dskernel. By default *fetchUpdatesItemsPerReq* is set to 100, while valid values range from 10 to 1000.
- *useCounter32_ifTypeList*, identifies the interface type as using 32 bit counters, the default is ifType 24 (loopback).
- *useCounter64_ifTypeList*, identifies interface types for which Entuity performs 64 bit counter polling. By default this list is empty.

  Do not amend the useCounter settings unless specifically advised to do so by your Entuity Support contact.

[FlexReporting]

Parameters in this section enable additional Flex Report functionality.

- *EnableExpressionBuilder* when set to 1 switches on, and when set to 0 switches off, Expression Builder. This example section switches on Expression Builder:

```
[FlexReporting]
EnableExpressionBuilder=1
```

[Flow]

Parameters in this section configure Integrated Flow Analyzer (IFA) ports and data rollups.

```
[flow]
port=9996
managementport=12121
compression=1
maxCountValue=10000
```

```
defaultQueryResultLimit=100
collectorWindowSec=300
# rollup0 is an IFA Premium setting, inactive by default
rollup0=0,1800,1Minute
rollup1=300,7200,5Minute
rollup2=3600,172800,1Hour
rollup3=21600,604800,6Hour
rollup4=86400,3024000,1Day
```

where:

- *port* is the port on which the Entuity server receives NetFlow, Netstream and JFlow data, by default 9996. Entuity IFA requires the exporting router to be configured with the IP address of the target Entuity server and a port number.

  You can set *Flow Port* during `configure` and through `flowcfg.properties`, where you can set multiple receiving ports.

  Entuity IFA receives sFlow and IPFIX packets through 2 non-configurable ports, for:

  - IPFIX you must set your router to export IPFIX to port 2055 of the Entuity server.
  - sFlow you must set your router to export sFlow to port 6343 of the Entuity server.

- *managementport*, the port Entuity uses to manage, e.g. stop, the flow collector process. You can set *Flow Management Port* during `configure`, by default to 12121.

When set, port values in `flowcfg-template.properties` and `flowcfg.properties` take precedence over the values set during `configure` and stored here in `entuity.cfg`.

- *compression*, sets how Entuity stores flow data. When set to:
  - **0**, Entuity retains the raw flow data, which implicitly increases the size of its database.
  - **1** (default), Entuity compresses the flow data.
- *maxCountValue*, sets the count limit, by default 10000, a value over 10000 and Entuity displays in the Count column 10k>. Count is an option available through Custom Breakdowns, an IFA Premium feature.
- *defaultQueryResultLimit*, sets the maximum number of results that can be returned to a Flow Analysis table.
- *collectorWindSec*, sets how Entuity handles flow records to account for differences in flow receipt and collection intervals creating spurious data spikes.

  For example, assume flow records are sent every two minutes for a continuous data transfer of 1k per second. You would expect a five minute sample to show 300k total. However since five is not a multiple of two the collector would alternate between two and three records per sample causing the displayed data to flip between 240k and 360k. Even if the flow records were sent every one minute there would still occasionally be a spike when the receipt of the flow records coincided with the collection boundaries.

To overcome this incoming records are apportioned into multiple buckets using the concept of a collection window which defaults to five minutes (300 seconds).

- *rollupN*, Entuity rolls up data to extract the most meaningful information and save it in a form that can be efficiently used to graph and report on over a longer period. For flow data there are five levels of rollup, with three attributes:
  - *frequency*, is the frequency of flow collection in seconds. When set to 0 Entuity does not collect flow data.
  - *keeptime*, is the length of time measured in seconds Entuity retains the rolled up data.
  - *directory*, is the name of the directory holding the rolled up data, for example **5minute**. You should only amend the name when you amend the rollup frequency.

Through the **rollup1** definition IFA supports a maximum flow collection frequency of five minutes. However through the **rollup0** definition IFA Premium can support a one minute flow collection frequency and retain that data for 30 minutes (1800 seconds), although by default it is not activated. To activate one minute flow collection amend the **rollup0** frequency from 0 to 60 seconds:

```
[flow]
rollup0=60,1800,1Minute
```

When you run IFA Premium the Entuity web UI includes options for running reports and flow breakdowns using one minute polling. However if you do not configure devices to send flow at one minute intervals, or activate the one minute rollup (**rollup0**), Entuity continues to use the five minute rollup data.

[image]

Parameters in this section apply to images used to represent services.

```
[image]
user_defined_directory=${ENTUITY_HOME}${FPS}etc${FPS}user_images
service_image_size=128
```

where:

- *user_defined_directory*, location of the custom images used in services. You must create this folder on the Entuity server.
- *service_image*, display size in pixels of the service image, by default 64x64.

[install]

Parameters in this section are installation settings for Entuity, for example:

```
[install]
dir=${ENTUITY_HOME}${FPS}install
java=${INSTALL.DIR}${FPS}JRE${FPS}bin${FPS}java
jre=${INSTALL.DIR}${FPS}JRE${FPS}bin${FPS}jre
```

```
platformfile=${ENTUITY_HOME}${FPS}etc${FPS}install.cfg
```

where:

- *dir* is the Entuity installation directory.
- *java* is the Java Runtime Environment.
- *jre* is the Java Runtime Environment used for the server installation.

In this example the *java* and *jre* paths are built using *dir* (i.e. `INSTALL.DIR`), where `INSTALL` refers to the section and `DIR` the variable name.

- *platformfile* is the installation configuration file for the current installation.

[ipman]

These parameters control configuration of ipman.

```
[ipman]
devicefile=D:\Entuity\etc\arp_cache_devices.cfg
```

where:

- *devicefile*, defines the location and name of a user defined file containing router hostname or IP address, and community string details for ipman to use to pull for ARP cache information.

  A device file is only required when an Entuity server does not manage a router containing ARP cache information it requires to populate connected end host IP addresses. (See *ipman*.)

[ipsla]

Parameters in this section are applicable to the Entuity Cisco IP SLA module:

```
[ipsla]
MinDiscoverableIndex=10000
MaxDiscoverableIndex=15000
```

where:

- *MinDiscoverableIndex*, defines the start of the range that Entuity checks for operator indices.
- *MaxDiscoverableIndex*, defines the end of the range that Entuity checks for operator indices

  Entuity checks for operations, with an owner of Entuity, on a device. If it finds an operation that it does not manage then it deletes the operation. When you have more than one Entuity server managing your network you should define a different range of operator indices for each server. This prevents two servers managing the same device destroying each others IP SLA operations, i.e. each server only checks operations that have an owner of Entuity and have an index within their discoverable range.

[Jasper]

Parameters in this section are applicable to the main Entuity reporting function. For example:

```
[Jasper]
maxCachedReportsPerSession=10
```

where:

- *maxCachedReportsPerSession*, sets the number of reports Entuity caches for each user's session. By default Entuity maintains 10 reports.

  You may want to increase this value, for example when using custom dashboards that include more than one report. However the greater the value the potentially greater increase in Apache Tomcat's memory footprint, and this greater load can slow performance.

[lcm]

Parameters in this section are applicable to the Entuity Configuration Monitor module. For example:

```
[lcm]
scriptDir=ENTUITY_HOME\integ\SCRAPE
expectProg=ENTUITY_HOME\integ\SCRAPE\expect
tftpServerIp=10.44.1.109
tftpPUsername=anonymous
tftpPassword=EYE
scpServerIp=10.44.1.109
scpUsername=anonymous
scpPassword=EYE
ftpServerIp=10.44.1.109
ftpUsername=anonymous
ftpPassword=EYE
diffDir=ENTUITY_HOME\integ\etc
tftpHome=c:\tftphome
archivedir=c:\tftpArchive
SNMPTriggerHoldOffTime=300
```

where:

- *scriptDir*, is the location of script files, by default *entuity_home*`/integ/SCRAPE`. Entuity Configuration Monitor includes these example scripts:
  - `start_run_cisco.expect`, for Cisco devices
  - `start_run_hp.expect`, for HP devices
  - `start_run_juniper.expect`, for Juniper devices.

- *expectProg*, full path to the Expect program, by default *entuity_home*/`integ/SCRAPE/expect`.
- *tftpServerIp* is the Entuity server IP address provided to the device. It is used by all Entuity Configuration Management transfer servers, i.e.TFTP, FTP, RCP and SCP and is set during `configure`.
- *FTPUsername*, username for access to devices where the FTP credentials are set on the command line, by default **anonymous**.
- *FTPPassword*, password for access to devices where the FTP credentials are set on the command line, by default **EYE**.

FTPUsername and *FTPPassword* are not used with Cisco devices.

- *TFTPUsername*, username for access to devices where the TFTP credentials are set on the command line, by default **anonymous**.
- *TFTPPassword*, password for access to devices where the TFTP credentials are set on the command line, by default **EYE**.
- *SCPUsername*, username for access to devices where the SCP credentials are set on the command line, by default **anonymous**.
- *SCPPassword*, password for access to devices where the SCP credentials are set on the command line, by default **EYE**.
- *diffDir*, location of differencing exclusions file by default *entuity_home*/`integ/etc`.
- *tftpHome* is the directory where retrieved configurations are first stored by the transfer server (TFTP or FTP). It is set during `configure`.
- *archivedir* is the location of the retrieved configuration archive. It is set during `configure`.
- *SNMPTriggerHoldOffTime* applies to change-based configuration retrieval, and is the period of time Entuity Configuration Monitor waits until making its final timestamp check before stating device configuration retrieval.

  When Entuity Configuration Monitor identifies a timestamp change, it does not immediately initiate device configuration retrieval but continues to monitor configuration timestamps on the device. When it identifies two consecutive polls with unchanged timestamps Entuity Configuration Monitor then waits the set hold time, by default 5 minutes. After the hold time elapses Entuity again checks that the timestamp remains unchanged, and if it remains unchanged initiates configuration retrieval.

[macman]

Parameters in this section are applicable to `macman`, for example:

```
[macman]
excludedMacs=00:00:0D:89:8D:AC-00:00:0D:89:8D:GG:FF,08:00:69:02:01:FC
trunkdevicecount=5
recallqueuetime=300
machistorylimit=50
```

```
queuemaxitems=128
queuemaxthreads=1
```

Where:

- *excludedMacs*, defines MAC addresses for `ipman` and `macman` to ignore, in addition to the default range of 00:00:0C:07:AC:00 to 00:00:0C:07:AC:FF, which are reserved for ethernet and FFDI HSRP group virtual mac addresses.
- *trunkdevicecount* is the threshold level of MAC addresses associated with a port, above which Entuity considers it a trunk port. When absent, or set to 0, the default value 10 is used.

  When trunk ports do not have encapsulation, or it has not been detected in the MIB, the MAC address count could become very large. This could lead to the database running slowly or memory exceptions. Limiting the MAC count using *trunkdevicecount* prevents this.
- *recallqueuetime*, the interval between the reading of requests to run `macman` against devices. By default it is set to 300 seconds, with a maximum value of 3600 seconds. After this delay, `macScheduler` may run all pending requests.
- *machistorylimit*, sets the limit on the retained history of MAC addresses. Entuity maintains two histories, for each:
  - Port Entuity retains, by default, the last fifty MAC addresses discovered on that port, when this threshold is passed Entuity discards the oldest MAC address.
  - MAC address Entuity retains, by default, the last fifty ports discovered for the MAC address. When this threshold is passed Entuity discards the oldest port.

  You should take care when amending *machistorylimit:*
  - Setting very large values increases the amount of data stored and can impact database performance.
  - The MAC Address New event is triggered when a MAC address is not listed in the retained history of MAC addresses for that port. Amending this variable changes when the event is triggered.
  - The MAC Address Port Change event is raised when a port is not listed in the history of that MAC address..
- *queuemaxitems*, maximum number of items in the `macScheduler` queue. By default the queue size is 64, with a maximum of 512.
- *queuemaxthreads* determines the maximum number of `macman`'s that `macScheduler` can run at any one time. For example when set to 1, only one `macman` can run. By default set to 1, with a maximum of 16. `macman` run by `provost` is not included in these restrictions.

[mibs]

Parameters in this section are applicable to how Entuity manages the MIB files it uses when managing the network.

```
[mibs]
```

```
MibDir = =${ENTUITY_HOME}${FPS}lib${FPS}mibs
parsedlimit = 50
parseMibDir = =${ENTUITY_HOME}${FPS}lib${FPS}mibs
```

where

- *MibDir* is the location of the directory holding the MIBs.
- *parsedlimit* sets the maximum number of MIB files that can be included to a batch. When you open the MIB Manager it loads the MIB file in batches.
- *parseMibDir* is the location of the directory holding the parsed MIBs.

When deploying traps and rules across multiple Entuity servers the events project file, together with the MIBs and parsed mibs directories must be identical across the servers.

[MibServ]

Parameters in this section are applicable to `StormWorks` SNMP collection.

The setting of MibServ parameters requires an understanding of the Entuity SNMP request architecture, therefore you should only amend the default settings with the guidance of Entuity Support. Incorrect configuration of these parameters can seriously impact Entuity performance.

- *backlog* controls how readily `StormWorks` increases the number of concurrent SNMP operations (but the concurrency will never go beyond the level set by *workers*). The higher the value the longer `StormWorks` will delay before increasing the concurrency level.

  The default value is 2, minimum value 0, maximum value 100.

It can be very hard to predict long term effects of changes here. Effects may only come to light long after the value was last changed.

- *SNMPagentPort* is the default port used by `StormWorks` for SNMP access to devices. The default is 161.
- *SNMPbadGraceCount* is the number of consecutive failures to communicate with a given device which `StormWorks` will tolerate before marking the device in question as bad. (A failed operation is counted once only, regardless of the number of retries involved.) While a device is marked as bad, all further requests to that device will be treated as though they had failed, without even attempting communication with the device. A device remains marked as bad for *SNMPbadHoldSecs*(qv), after which time `StormWorks` will try to resume normal communication with the device.
  - *SNMPbadGraceCount*=0 means `StormWorks` will consider a device to be bad after first error
  - *SNMPbadGraceCount*=1 means `StormWorks` will consider a device to be bad after two consecutive errors

The default value is 1, minimum 0, maximum 10. It is a single setting applied to all devices accessed via `StormWorks`.

Adjusting this value may degrade performance, but effects may only come to light long after the value was last changed.

- *SNMPbadHoldSecs* is the time `StormWorks` keeps a device marked as bad. During this period all requests to that device will be treated as though they had failed, without even attempting communication with the device. At the end of that period `StormWorks` will try to resume normal communication with the device.

  The default value is 30, minimum 5, maximum 120. It is a single setting affecting all devices accessed via `StormWorks`.

Adjusting this value may degrade performance, but effects may only come to light long after the value was last changed.

- *SNMPgatherMaxMsecs* as described for *SNMPgatherMinMsecs*, if fresh requests for the same target keep arriving, the hold back time may accumulate. The value here serves as an upper bound on worst case cumulative hold back time for any request. Single setting affecting all devices accessed via `StormWorks`.

  The default value is 5000, minimum *SNMPgatherMinMsecs*, maximum 15000.

Adjusting this value may degrade performance, but effects may only come to light long after the value was last changed.

- *SNMPgatherMinMsecs*

  `StormWorks` normally holds back SNMP requests for a short time to take advantage of the efficiency benefit from combining them with other requests to the same device. If a request is held back and a further request on the same device arrives within the time specified here, those requests will continue to be held. If no further requests for some device arrive within the time specified here, all held back requests for that device will then be actioned. Single setting affecting all devices accessed via `StormWorks`.

  The default value is 500, minimum 100, maximum 5000.

Adjusting this value may degrade performance, but effects may only come to light long after the value was last changed.

- *SNMPoidsPerPdu* is the maximum number of oids which to be passed in a single pdu. The default value is 30, minimum 0, maximum 50. It is a single setting affecting all devices accessed via `StormWorks`.

`StormWorks` recognizes device responses caused by oversize pdus, and transparently re-issues the pdu with successively smaller numbers of oids until it succeeds. This mechanism is independent of the value set in *SNMPretryLimit*.

- *SNMPreadCommunity* is the default community string. The default value is public. It is a single setting affecting all devices accessed via `StormWorks`.
- *SNMPredAlertSecs*

  If an `StormWorks` SNMP operation remains internally queued for longer than this time, a red alert error message will be logged in `DsKernelStatic.log`, and the operation will be treated as though it failed. The default value is 120, minimum *SNMPyellowAlertSec*s, maximum 3600. It is a single setting affecting all devices accessed via `StormWorks`.

Red alert messages indicate serious problems internal to `StormWorks` which may need involvement from Entuity Support. The solution will involve adjustments elsewhere, changing the value here could make it harder to solve any resulting problems.

- *SNMPretryLimit* is the number of retries if an initial attempt fails. It is a single setting affecting all devices accessed via `StormWorks`.

  SNMPretryLimit=0 means that if initial attempt fails, `StormWorks` will not retry. The default value is 3, minimum 0, maximum 20.

Increasing this value may degrade performance, but effects may only come to light long after the value was last changed.

- *SNMPretryMillisecs* is the time allowed before attempting to retry. Values must allow for worst case round trip times, with particular attention to any devices accessed via slow or high-latency links.

  The default value is 3000, minimum 0, maximum 30000. It is a single setting affecting all devices accessed via `StormWorks`.

Increasing this value may degrade performance, but effects may only come to light long after the value was last changed.

- *SNMPversion* is the default SNMP version.

  This does not impact functionality implemented via the `StormWorks` language, as this always requires a parameter explicitly specifying the SNMP version for each operation. The default is V1, alternative V2c.

- *SNMPyellowAlertSecs*

  If an `StormWorks` SNMP operation remains internally queued for longer than this time, a yellow alert warning message will be logged in `DsKernelStatic.log` but without any other effect.

The default value is 30, minimum 10, maximum 3600. It is a single setting affecting all devices accessed via `StormWorks`.

Yellow alert messages indicate problems internal to `StormWorks` which may need involvement from Entuity Support.

- *workers* is the maximum number of SNMP operations that can be concurrently active. When the limit is reached operations are queued until a worker is available. The default value is 15, minimum 1, maximum 500.

Excessive values can cause serious performance degradation, but effects may only come to light long after the value was last changed.

- *udpWorkers* is the maximum number of SNMP operations that can be concurrently active with user defined polling. When the limit is reached operations are queued until a worker is available. The default value is 10, minimum 1, maximum 500.

Excessive values can cause serious performance degradation, but effects may only come to light long after the value was last changed.

[OTR]

Parameters in this section are applicable to Entuity trap management and how `prologV2` handles traps from unmanaged devices and interfaces.

```
[OTR]
suppressUnmanagedDevices=false
suppressUnmanagedInterfaces=false
```

where

- *suppressUnmanagedDevices* controls how Entuity handles unmanaged devices. When set to:
  - **false** (default) Entuity handles traps from unmanaged devices.
  - **true**, Entuity suppresses traps from unmanaged devices.
- *suppressUnmanagedInterfaces* controls how Entuity handles unmanaged interfaces. When set to:
  - **false** (default) Entuity handles traps from unmanaged interfaces.
  - **true**, Entuity suppresses traps from unmanaged interfaces.

[prole]

Entuity constructs port descriptions by placing within square brackets the port's index value, using either its enterprise MIB index (entIndex) when available, or interface index (ifIndex). The index value is followed by the port description. Parameters in this section allow you to control exactly how Entuity constructs the displayed interface descriptions.

There will be a delay between activating these settings and the changes becoming apparent in Entuity. One cause of delay is `prole`, or on a large site a number of instances of `prole`, only running every twenty minutes. Another is the Entuity UI refresh rate. There may also be occasions when some ports show the description in the old format, and some in the new format, this is because `prole` cannot read all ports at exactly the same time.

You can configure interface descriptions through this section:

```
[prole]
PollIfName=1
ifDescrUseAlias=true
ifDescrAppendAlias=false
ifDescrSortableIndex=false
ifDescrLabelIfIndex=false
```

Where:

- *PollIfName*, controls the port description square bracket population. When set to:
  - **1** (default), Entuity populates the interface name from the ifName mib variable
  - **0**, Entuity derives the interface name entIndex or ifIndex.
- *ifDescrUseAlias*, sets the port description used by Entuity when set to:
  - **true** (default) Entuity uses the port's interface alias
  - **false** Entuity uses the port's MIB2 description.
- *ifDescrAppendAlias*, customises the port description used by Entuity when set to:
  - **true** Entuity appends to the ifDescr the port's alias within round brackets, e.g. ATM0/IMA0 (*** IMU to Chandler via ATT ***). This setting can only be used when *ifDescrUseAlias* is set to **true**.
  - **false** (default) Entuity replaces the port's ifDescr with its alias, when an alias is set.
- *ifDescrSortableIndex*, sets the format of the port index within square brackets, when set to:
  - **true** presents a ports index in a format suitable for an alphanumeric sort. For example using these formats [99/999], [99/999/999] and [9999] for two part entIndex, 3 part entIndex and If Index respectively.
  - **false** (default) Entuity displays port indices as discovered without adding leading zeroes to improve the sort order. For example [#9/##9], [#9/##9/##9] and [###9] for two part entIndex, 3 part entIndex and If Index respectively, where # indicates low values will not be right side zero filled, e.g. [1] rather than the zero filled [0001].

ifDescrSortableIndex= true	*ifDescrSortableIndex*=false
[0001]	[1]
[0002]	[108]
[0108]	[109]

Table 4 Different Sort Orders of the Interface Description Formats

ifDescrSortableIndex= true	*ifDescrSortableIndex*=false
[0109]	[110]
[0110]	[2]
[02/001]	[2/1]
[02/010]	[2/10]
[02/020]	[2/1]

Table 4 Different Sort Orders of the Interface Description Formats

- *ifDescrLabelIfIndex*, prefixes the interface description with if:, to indicate the value is an interface index (and so should only be used when port data is accessed using its interface index). When set to:
  - **true** Entuity applies the if: prefix, e.g. [if:0001]
  - **false** (default) Entuity does not apply the if: prefix, e.g. [0001].

[proliferate]

Parameters in this section are used with `proliferate`:

```
[proliferate]
maxpolltime=300
```

Where:

- *maxpolltime*, sets the maximum time, in seconds, for a device to respond to an SNMP poll after which Entuity considers it a slow device.

[reporting]

Parameters in this section are used when generating Flex Reports and standard reports:

```
[reporting]
spareporttime=40
ViewReportsDefault:showFlex=1
generateReportUseRedirect=0
generateReportRPCTimeout=60
regenerateReportRPCTimeout=600
deleteFlexReportRPCTimeout=1
generateReportcommand lineRPCTimeout=86400
javaMemory=128000000
viewReportOnScreenMessageSize=200
foCommand=java -cp
${XML.XERCES};${XML.XALAN};org.apache.xalan.xslt.Process
```

```
pdfCommand=java -cp
${XML.FOP};${XML.BATIK};${XML.FOPXALAN};${XML.FOPXERCES};${XML.JIMI};
org.apache.fop.apps.Fop

JasperChangeDataKeepTime=63072000
```

Where:

- *spareporttime*, sets the threshold number of days a port is unused, used in spare port calculations. The default is 40.
- *ViewReportsDefault:showFlex* when set to:
  - 1 automatically displays Flex Reports in the Report Center's View Reports page.
  - 0 (default) does not automatically display Flex Reports in the Report Center's View Reports page.
- *generateReportUseRedirect*, when set to:
  - 1 (default), a redirect page is used with reports generated from Report Center.
  - 0, a redirect page is not used.
- *regenerateReportUseRedirect*, when set to:
  - 1 (default), a redirect page is used with reports regenerated from Report Center.
  - 0, a redirect page is not used.
- *generateReportRPCTimeout*, period of time, in seconds, before Flex Report generation times out (by default one minute). In the browser Entuity displays an information message that the report is still being generated.

  When running Flex Reports using URLs and setting *noCreate=1* increase the timeout parameter. This prevents the CGI timing out during report generation and the report object not being deleted. This can be set as a runtime parameter in the URL.
- *regenerateReportRPCTimeout*, period of time, in seconds, before Flex Report regeneration is considered to have timed out.
- *deleteFlexReportRPCTimeout*, queues Flex Reports for deletion. When **dsKernelStatic** is very busy and not responding this default may be increased.
- *generateReportcommand lineRPCTimeout*, sets the RPC timeout for **GenerateReport** when Flex Reports are run from the command line, by default one day.
- *javaMemory*, size of memory available to the java process when running Flex Reports. When reports fail due to java memory problems the reason is detailed in Report Manager, if extra memory is available increase this setting. The default is 128000000.
- *viewReportOnScreenMessageSize*, sets number of characters available to display Flex Report error messages. The default is 200 characters.
- *foCommand* generates the report object file (FO file) which is used to generate the end report. The parameter are built from values defined through the xml section of `entuity.cfg`.
- *pdfCommand* generates the pdf format report in report from the object file (FO file). The parameter are built from values defined through the xml section of `entuity.cfg`.
- *JasperChangeDataKeepTime*,the data keep time for reports that include a compare function, i.e. the Data Integrity report. The default is two years, entered as seconds, i.e.

63072000.

[search]

This section configures the Search tool.

```
[search]
rpcServerPort=5469
maxResultSize=50,100,500,1000,2000
```

Where:

- *rpcServerPort* specifies the RPC port used by Tomcat to communicate with the Search tool.
- *maxResultSize*, sets the options available to the user when selecting how many search results to return from Entuity servers.

[Server]

Parameters in this section are set during configure and relate to the Entuity server:

```
proxy_timeout=300
ssl_enabled=false
single_socket_enabled=true
id=9c3d450f-a80e-42cc-864a-b9dec8b49549
```

Where:

- *proxy_timeout*, overrides the ProxyTimeout directive set in `httpd_eye.conf`. This directive allows you to set a timeout on proxy requests, which is useful when you have a slow server response. By default set to 300 seconds.
- *ssl_enabled*, when set to true the server uses SSL, when set to false it doe not.
- *id*, is the unique Entuity server identifier. It is used internally by different components of the server and externally by other Entuity servers.

  When using Entuity to send SNMPv3 traps Entuity derives the *engineID* from the Entuity server *id* but also removes the underscores, for example:

  ```
  id=9c3d450f-a80e-42cc-864a-b9dec8b49549
  ```

  becomes the *engineID*:

  ```
  9c3d450fa80e42cc864ab9dec8b49549
  ```

  Through the `entuity.cfg` setting *events.engineidoverwrite* you can override this default and enter your own value, It must be a hex string including only the characters 0-9 and A-F, at least 5 bytes and no more than 32 bytes long.

[Services]

Parameters in this section set the monitoring of remote Entuity servers by the central server:

```
contentchangeregistrymaxstaletime=600
```

```
remotecomponentlistupdatefrequency=600
remotecomponentregistrymaxstaletime=600
staleremoteserverkeeptime=3600
```

Where:

- *contentchangeregistrymaxstaletime* specifies .
- *remotecomponentlistupdatefrequency* specifies .
- *remotecomponentregistrymaxstaletime* specifies .
- *staleremoteserverkeeptime* specifies .

[Sla]

Parameters in this section configure `slalogger` and the roll up of its data. This example uses the default values**:**

```
[sla]
Rollup=192@1h;100@1d;13@1w;24@1m;5@1y
startdayofweek=0
```

Where:

- *Rollup* specifies how the data collected by Availability Monitor is retained. This has the format:

  `<no of retained samples>@<interval length><unit of time>`

  where:

  - <no of retained samples> is how many samples to keep
  - <interval length> is how the original data should be rolled up.
  - <unit of time> is the original sample interval, i.e. **h** for hour, **d** for day, **w** for week, **m** for month and **y** for year.

Hourly samples start on the hour, daily at midnight, weekly start time is set through *startdayofweek*, monthly at midnight on the first day of the month and yearly samples start at midnight on the first of January.

For example:

```
Rollup=192@1h;100@1d;13@1w;24@1m;5@1y
```

This example uses the default values and means:

- Polled hourly data is kept for one hundred and ninety-two hours, equivalent to eight days.
- Rolled-up daily data is kept for one hundred days.
- Rolled-up weekly data is kept for thirteen weeks.

- Rolled-up monthly data is kept for twenty-four months.
- Rolled-up yearly data is kept for 5 years.

If you amend these defaults you must ensure you enter valid values, i.e. do not define strings that request too little data. For example:

```
RollUp=12@1h;31@1d
```

This requests twelve one hour data samples, which is less than the twenty four one hour samples required to make one day. Therefore, Entuity overrides the entered value and takes twenty four one hour samples.

If you amend *Rollup* you must stop and then restart the Entuity server for the changes to take effect.

- *startdayofweek* is used in SLA reports to specify the first day of the reporting week. 0 = Sunday, 1 = Monday, through to 6 = Saturday. The default is 0.

[snews]

Parameters in this section are applicable to Device News module:

- *maxSamples* sets the number of days Device News configuration details should be held (i.e. device and VLAN switch details). The default value is 7, for 7 days.

[SNMPserv]

The parameters within this section are used by the SNMP Server:

- *checkWalkOrder* is used when determining whether Entuity performs lexicographic checking on data returned by a MIB agent.

  This is useful when an agent returns data out of sequence as part of a SNMP GetNext request. For example, with many lower end Cisco devices (e.g. 1900, 2820 and 2900XL's) the section that contains mac address information is unordered. Without the lexicographic checking this data can cause the GetNext request to form a recursive loop, with checking this can be avoided.

  When *checkWalkOrder* is set to:

  - 0, Entuity does not check that the returned data is in the correct order. This is the default state.
  - 1, Entuity performs lexicographic checking. If returned data fails the checking Entuity writes an error message to the calling process' log file and discards the data. For example, if `macman` is run and the data fails the checking, error messages are written to `macman.log` and in the web UI you would notice mac addresses are missing.

If a process inexplicably locks up, e.g. `macman`, `prole`, it may be due to a GetNext request loop and setting *checkWalkOrder* to 1 may solve the problem.

SNMP operations controlled through `StormWorks` are separate from SNMP Server. Lexicographic setting is always enabled.

[syslogger]

Configuration section for the System Logger process. It determines the port the System Logger process listens for syslog messages on and the level of urgency and facility (message type) that then lead to alarms being generated in Event Viewer.

This example section configures `syslogger` to only accept messages that are: from Entuity managed devices; received on port 514; either of message type mail with a log level of debug or higher, or kern with a log level of notice or higher.

```
[syslogger]
loglevel=notice
portnum=514
openReceiver=0
acceptfacs=mail.debug,kern.
replaceEventDetailsAction=s/\n/ /g s/^//g
```

Where:

- *loglevel* is the message urgency level. It sets the urgency level of syslog messages for which Entuity generates events. This level can used/overridden through *acceptfacs*.
- *portnum* is the port System Logger process listens on, the default is 514.
- *openReceiver* when set to:
  - 0, limits System Logger process so it only handles messages from devices managed by Entuity.
  - 1, the default, System Logger process handles messages from all devices.
- *acceptfacs* allows you to specify which facilities are accepted by Entuity and at what urgency level. These are the acceptable formats:
  - *facilityname.loglevel*, for example **mail.debug**. `syslogger` accepts mail syslog messages of debug level and above.
  - *facilityname.*, for example **kern**. Only messages that are both kern message type and have an urgency level of *loglevel* or above are accepted.
  - **All**, the default, accepts all message types. The urgency level is taken from *loglevel.*

  When *acceptfacs* is:
  - Not specified all messages that meet the log level result in Entuity events.
  - Specified only messages of that type and log level result in Entuity events.
- *replaceEventDetailsAction* takes regular expressions through which you can define replacement of characters before information is displayed in Event Details, for example you can replace each line break with a space, remove a carat `^`:

  ```
  replaceEventDetailsAction=s/\n/ /g s/^//g
  ```

replaceEventDetailsAction has the format:

`/s/`*searchString*`/`*replacementString*`/g`

Where:

- /s identifies a substitution command.
- /*searchString* is the string in the trap text to be replaced.
- /*replacementString* is the replacement string which can include a space, or nothing.
- /g identifies it as a global command.

When the syslogger section is not included in `entuity.cfg`, then System Logger process is set to its default state. It accepts messages from the notice urgency level, listens on port 514 and accepts all facilities from all devices.

[system_control]

Parameters in this section define Entuity system control. This example section starts Entuity in maintenance mode:

```
[system_control]
defaultState=maint
```

Available parameters are:

- *config* holds the path and name of the Entuity startup file, *entuity_home*`/etc/startup_`*O*/*S*`.cfg`, where *O*/*S* is an abbreviation that identifies the operating system.
- *defaultState* sets the type functionality when Entuity is started. The default is **normal**. This starts every module in Entuity that has **normal** associated with it in the startup configuration file (see *startup_o*/*s.cfg*).
- *delay* sets the time between each failed start attempt. The default is 5 seconds.
- *retry* sets the number of attempts at starting Entuity. The default is 3.

Entuity recommend you do not adjust the default system control settings.

[ticker]

Parameters in this section are applicable to Ticker:

- *maxClients* is the maximum number of Ticker clients the server can monitor. The default is 256.
- *port* is the port the Ticker server monitors its client ports' activity. The default is 20202, set during Entuity configuration.

[tomcat]

Parameters in this section configure Apache Tomcat application server:

- *adminport*, is the Tomcat administration port, by default 8005.

- *port*, is the Tomcat port, by default 8080.
- *javaMemory*, is the amount of memory assigned to the tomcat java process, by default 512M.

[Topology]

Parameters in this section control the display of topology information in maps.

```
[Topology]
PingStateIncludedDeviceTypes=168,1049,1058,1077,1128,1200
PingStateExcludedInterfaceTypes=24, 28, 33, 34, 48
EnableSpanningTree=1
EnableUplinkDetection=1
```

Where:

- *PingStateIncludedDeviceTypes* allows you to override the default device types included to Trace Route - Ping State maps. This list replaces the default list so you must include all device types you want included to the map, not only the additional device types.
- *PingStateExcludedInterfaceTypes* sets the port types Entuity excludes from the displayed Trace Route - Ping State in maps. This list replaces the default list so you must include all interface types you want excluded from the map, not only the additional interface types.

Device and interface types are listed in *Appendix B - Entuity Internal Identifiers*.

- *EnableSpanningTree* sets whether spanning tree is enabled. When set to:
  - 1 (default), maps can display spanning tree information.
  - 0, maps cannot display spanning tree information.
- *EnableUplinkDetection* sets whether uplink detection is enabled. When set to:
  - 1 (default), maps can display uplink details.
  - 0, maps cannot display spanning uplink details.

[Traps]

Parameters in this section are applicable to `prologV2`**.** You should only adjust these settings when you suspect that the rate at which traps are being received is faster than they can be handled by `prologV2` and traps are being lost. This example section details the default configuration:

```
[Traps]
usetrapqueue=F
queuemaxitems=512
queueresumethreshold=480
```

Where:

- *usetrapqueue* when set to

- F, a queue is not used.
- T, a queue is created by `prologV2` to supplement the system cache.

- *queuemaxitems* is the maximum number of items in the queue, by default 512 traps.
- *queueresumethreshold* causes the queue to stop receiving traps.

[viewServer]

By default the event engine process uses the internal Entuity mechanism, `viewserver` for view membership checks.

```
[viewServer]
RefreshInterval=1200
```

where:

- *RefreshInterval*, `viewserver` checks object-view and content filter settings, by default every twenty minutes (1200 seconds). This coincides, but is not synchronised, with the default interval for the running of `prole`. Valid values are in the range of 60 to 86400 seconds, i.e. one minute to one day.

[updateNames]

`updateNames` compares the device *Display Name* in Entuity against the value on the device. If there is a difference `updateNames` updates the *Display Name.* `updateNames` is scheduled, by default, to run at 03:00 every day. It can be disabled through a setting in `entuity.cfg`:

```
[updateNames]
disabled=true
```

[webUI]

Parameters in this section configure Event Viewer. This is an example configuration:

```
[webUI]
EventViewerMaxEvents=1000
EventViewer.BatchSize=1000
EventViewerShowServerColumn=1
EventViewerSeveritySound=info:chimes.wav, minor:chord.wav,
major:ding.wav, severe:notify.wav, critical:ringin.wav
customDashboardMaxCount=20
customDashboardMaxUrlCount=20
ActiveChartDefaultGroupApproximation=average
```

where:

- *EventViewerMaxEvents*, sets the maximum number of events that can be held by Event Viewer, by default 1000.
- *EventViewer.BatchSize*, sets the maximum number of events that can be displayed by Event Viewer, by default 1000.

- *EventViewerShowServerColumn*, when set to:
  - **0** (default) the server column is hidden in Event Viewer.
  - **1**, Event Viewer displays the server column which identifies the Entuity server that raised the event, which you may require in multi Entuity server environments.

  Changes to this setting are only applied after a restart of tomcat. The setting is only retrieved from the server you are logged into. Setting this option on a remote server has no effect unless you directly login to the remote server.
- *EventViewerSeveritySound*, allows you to set a sound for each event severity level. You must install your own sound files (WAV or MIDI) to *entuity_home*/lib/TomCat/webapps/webUI/sounds. For changes to this setting to be applied you must restart Apache Tomcat.
- *customDashboardMaxCount* sets the upper limit to the number of custom dashboards a user can potentially have available from the **Dashboards > Custom Dashboards** menu. The user configures the maximum custom dashboards available to them through the Preferences page and the Dashboard Count, which by default has an upper limit of 20.

  From the Preferences page the user can amend the Dashboard Count from 5 up to 20.

  By adjusting the value of *customDashboardMaxCount*, up to a maximum value of 50, you can allow the user to potentially set a higher number of custom dashboard menu items. If you set a value greater than 50 Entuity sets the number of dashboards to 50.
- *customDashboardMaxUrlCount* sets the upper limit to the number of URLs in a custom dashboard. By default the maximum number of URLs per dashboard is 20, you can amend this to an upper limit of 50. If you set a value greater than 50 Entuity sets the number of URLs per dashboard to 50.
- *ActiveChartDefaultGroupApproximation* sets how Entuity displays a large amount of data on a chart. When set to:
  - **average** (default), Entuity uses a grouping algorithm to prevent the chart from becoming crowded with overlapping data points. This algorithm can lead to the loss of peak information.
  - **High**, Entuity retains peak data points where high resolution data is available.

  You can modify this setting for individual charts through the Customize Chart dialog and setting Group Approximation to Preserve Peak (High) or Average (average).

[xml]

Parameters in this section are used by the reporting section in `entuity.cfg` when generating Flex Reports. They must not be amended from the default settings:

```
[xml]
xmlDir=${ENTUITY_HOME}${FPS}lib${FPS}xml${FPS}
xerces=${XML.XMLDIR}xalan${FPS}xerces.jar
xalan=${XML.XMLDIR}xalan${FPS}xalan.jar
transformCommand=java -cp
${XML.XERCES};${XML.XALAN};org.apache.xalan.xslt.Process
fop=${XML.XMLDIR}fop${FPS}fop.jar
```

```
batik=${XML.XMLDIR}fop${FPS}batik.jar
fopXalan=${XML.XMLDIR}fop${FPS}xalan-2.0.0.jar
fopXerces=${XML.XMLDIR}fop${FPS}xerces-1.2.3.jar
jimi=${XML.XMLDIR}fop${FPS}jimi-1.0.jar
IllegalCharacters=27,146,147,148
```

Where:

- *xmlDir* is the folder under which are the folders holding the xml library files.
- *xerces* references the java XML parser Xerces.
- *xalan* references the java XSLT stylesheet processor Xalan.
- *transformCommand* generates the report object file (FO file) which is used when displaying report data to screen.
- *fop* references the java XSL Formatting Object processor FOP.
- *batik* references the java based toolkit for Scalable Vector Graphics.
- *fopXalan* references a version of Xalan compatible with FOP.
- *fopXerces* references a version of Xerces compatible with FOP.
- *jimi* references a version of jimi compatible with FOP.
- *IllegalCharacters* identifies unprintable control characters that when encountered when generating the XML would otherwise cause the report to fail. Each unprintable character is replaced with a question mark. Characters are referenced using ISO-8859-1 encoding, but by default are not specified in the configuration.

eventEngine.bat

Location

entuity_home/`bin`

Format

Maintained by Entuity.

Description

A Windows batch file (Linux shell script is `eventEngine`) which when run configures the eventEngine according to settings in `event-engine-cfg.properties`. The eventEngine does not require restarting for the configuration changes to be applied, for example when run from the *entuity_home*/`etc` directory enter:

```
bin/eventEngine.bat -reloadCfg
```

Status

Read-only.

event-engine-cfg-template.properties

Location

entuity_home/etc

Format

Maintained by Entuity.

Description

This is a template file and may be overridden. To make persistent changes copy this file to the event-engine-cfg.properties file and edit it. You can apply changes by running the batch file eventEngine.bat (in Linux the shell script eventEngine).

You should contact your Entuity representative before amending these configurations.

```
# Indicates if tracing is switched on for every incoming event: useful
for debugging rules
traceAllEvents = false
# Queue sizes for the events originating from external systems:
# - initial: the initial size of the queue per worker
# - max: the maximum size of the queue per worker
# - total: total size of queues across all workers
initialRawEventQueueSize = 100
maxRawEventQueueSize = 10000
totalMaxRawEventQueue = 50000
# Queue sizes for the events originating from the event engine itself
initialDerivedEventQueueSize = 10
maxDerivedEventQueueSize = 1000
totalMaxDerivedEventQueue = 5000
# Maximum number of states available to rules
maxRuleStates = 50000
# The duration since the last update to the NofM rule state after
which the state can be discarded
nmRuleStateTimeoutSec = 172800
# Number of events stored in the event cache
maxEventCacheSize = 20000
# The time period for flushing events from the event cache to the
database
eventFlusherFlushPeriodMs = 1000
# The time between archive cleanup jobs
archiveClenupPeriodSec = 1700
```

```
# The number of records to delete in a single batch
archiveDeleteBatchSize = 20000
# The number of events which can be stored in the archive per
situation
archiveMaxSituationEvents = 100
# Maximum number of incidents: including open, closed and expired
maxSituationCount = 50000
# The maximum number of events returned per incident
maxReturnedEventsPerSituation = 100
# The duration for which expired incidents should be kept
situationEvictionPeriodSec = 604800
# The duration for which deleted incidents should remain in memory
situationExtraEvictionPeriodSec = 600
# The name for the default incident
defaultSituationName = Unclassified
# Age out for the default incident
defaultSituationAgeOutSec = 3600
# Expiry window for the default incident
defaultSituationReopenWindowSec = 10800
# Opening window for the default incident
defaultSituationOpeningWindowSec = 300
# Indicates if incident needs to be created for the event with
severity = info
informationalEventCausesDefaultSituation = false
# The minimum duration, which may pass before system event's cache can
be reloaded
minSystemEventReloadPeriodSec = 300
# The View event/incident filter reload period
viewEFilterRefreshPeriodSec = 300
# Positive and negative caching durations for compId to swId
keepTimeForCompIdToSwIdSec = 7200
keepTimeForCompIdToSwIdNegSec = 5
# Positive and negative caching durations for swId to object
description
keepTimeForSwIdToObjectDescriptorSec = 300
keepTimeForSwIdToObjectDescriptorNegSec = 5
# Positive and negative caching durations for swId to object details
```

```
keepTimeForSwIdToObjectDetailsSec = 20
keepTimeForSwIdToObjectDetailsNegSec = 20
# Positive and negative caching durations for swId reference to swId[]
keepTimeForSwIdRefToObjectIdsSec = 20
keepTimeForSwIdRefToObjectIdsNegSec = 20
# Positive and negative caching durations for serverId to deviceId
keepTimeForServerIdToDeviceIdSec = 3600
keepTimeForServerIdToDeviceIdNegSec = 5
```

Status

Read-only.

Changes to

eventProject.xml

Location

entuity_home/etc

Format

Maintained by Entuity.

Description

This file configures the event system, for example its incidents, rules, actions. Entuity is shipped with a default project file. When you save and deploy a project Entuity updates the XML file.

Status

Read-only.

eyepoller_overrides.cfg

Location

entuity_home/etc

Format

Text file.

Description

Entuity's default behavior is to poll a device using a port with MIB2 support. When a device does not include a port with MIB2 support and uses its own enterprise MIB to collect device data Entuity's default behavior would not return data. Through `eyepoller_overrides.cfg` you can configure Entuity to poll the enterprise MIB. The

polling definitions are held in separate configuration files which would be developed by Entuity Professional Services.

On Entuity startup `eyepoller` checks for `eyepoller_overrides.cfg` and when it is available reads its configuration. `eyepoller` only checks `eyepoller_overrides.cfg` when it starts, it does not reread the file again until it is restarted.

`eyepoller` configuration has the format:

```
sysoid> status <admin-status-oid:indexing> <oper-status-oid:indexing>
<time-of-last-change-oid:indexing> {<sysuptime-oid>}

<sysoid> util64 <in-octets-64:indexing> <out-octets-64:indexing>
```

where:

- Indexing should be either **M2** or **ES** to indicate use of ifIndex or entIndex respectively.
- SNMPv1 polling is used for status.
- SNMPv2 for util64, SNMPv3 for SNMPv3 devices.
- Status sysuptime-oid is optional, and if not present the default of 1.3.6.1.2.1.1.3 is used.

If there is an error in the formatting of any line, the line's instructions are ignored and a warning of the failure is entered in `eyepoller.log`. An information message is also added to `eyepoller.log` for each successful override read from the file. Comment lines starting with # and blank lines are silently ignored.

Status

Maintained by Entuity and used with configuration produced by Professional Services. Changes to this file are maintained during Entuity upgrades.

eyepoller_overrides_system.cfg

Location

entuity_home/`etc`

Format

Text file.

Description

Entuity's default behavior is to poll a device using a port with MIB2 support. When a device does not include a port with MIB2 support and uses its own enterprise MIB to collect device data Entuity's default behavior would not return data. Through `eyepoller_overrides.cfg` you can configure Entuity to poll the enterprise MIB. The polling definitions are held in separate configuration files which would be developed by Entuity Professional Services.

On Entuity startup `eyepoller` checks for `eyepoller_overrides.cfg` and when it is available reads its configuration. `eyepoller` only checks `eyepoller_overrides.cfg` when it starts, it does not reread the file again until it is restarted.

`eyepoller` configuration has the format:

```
sysoid> status <admin-status-oid:indexing> <oper-status-oid:indexing>
<time-of-last-change-oid:indexing> {<sysuptime-oid>}
<sysoid> util64 <in-octets-64:indexing> <out-octets-64:indexing>
```

where:

- Indexing should be either **M2** or **ES** to indicate use of ifIndex or entIndex respectively.
- SNMPv1 polling is used for status.
- SNMPv2 for util64, SNMPv3 for SNMPv3 devices.
- Status sysuptime-oid is optional, and if not present the default of 1.3.6.1.2.1.1.3 is used.

If there is an error in the formatting of any line, the line's instructions are ignored and a warning of the failure is entered in `eyepoller.log`. An information message is also added to `eyepoller.log` for each successful override read from the file. Comment lines starting with # and blank lines are silently ignored.

Status

Maintained by Entuity and used with configuration produced by Professional Services. Changes to this file are maintained during Entuity upgrades.

flowcfg-template.properties

Location

entuity_home/`etc`

Format

Text file containing commented out examples of how to customize the configuration of Entuity IFA flow collectors.

Description

Entuity IFA flow collectors are shipped with a factory configuration suitable for most network environments. You can amend this configuration, for example specify more than one port for Entuity to accept flow data, increase the size of the buffer handling incoming flow packets.

When set, values in `flowcfg-template.properties` take precedence over those values entered during `configure` and stored in `entuity.cfg`. If you create `flowcfg.properties` its settings take the highest precedence.

Default configuration:

```
receiver1_port = 9996
receive_buffer_size = 0
jmxserver_port = 12121
jmxFile = C:/Entuity/log/flowJmxUrl.jmx
packet_queue_limit = 10000
```

```
packet_sequence_check = 0
perform_inventory_filtering = 0
permanent_flows_capacity=10000000
dbDriver = com.mysql.jdbc.Driver
dbUrl = jdbc:mysql://127.0.0.1:3306/flowdb
dbUser = root
dbPwd =
partition1_maxCount = 1000000
ageOutFlows1 = 65
ageOutRuns = 1500
ageOutStats = 1500
trace=0
packetLogging=off
```

where:

- *receiver1_port*, by default there is only one receiver, but multiple can be specified, for example:

  ```
  receiver1_port = 9996
  receiver2_port = 9998
  ```

The receiver port setting only applies to the receiving of NetFlow data, IFA only receives IPFIX data on port 2055 and sFlow data on port 6343 of the Entuity server.

- *receive_buffer_siz,e*, the size of the datagram socket receive buffer size in bytes. This is a suggested value and does not reflect actual buffer size. If there are a lot of missed packets observed then this value should be increased. Set it to zero to use OS default settings.
- *jmxserver_port*, the port Entuity uses to manage, e.g. stop, the flow collector process. You can also set *Flow Management Port* during `configure`, by default to 12121.
- *jmxFile*, the URL to the JMX agent
- *packet_queue_limit*, the limit of the packet queue, by default 10000. Receivers write to the queue and packet processor reads from that. If packet queue becomes full then packets get dropped.
- *packet_sequence_check*, indicates whether to check packet sequence numbers. When set to:
  - **1** packet processor calculates the number of missed packets and rejects out-of-sequence packets.
  - **0** (default) is off.
- *perform_inventory_filtering*, indicates whether to filter out the flows. When set to:
  - **1**, IFA only accepts flow from known interfaces, i.e. interfaces under Entuity management

- **0** (default), IFA accepts flows from all interfaces on known devices, i.e. devices under Entuity management.

- *permanent_flows_capacity* sets the size of the cache that retains the current and previous values of the inbound and outbound counters. By default it is set to one million entries (each entry/record has at least 50 - 100 bytes).

  Some devices, for example Cisco ASA firewalls, send absolute transfer values (bytes/packets) instead of relative values. The NetFlow template contains IN_PERMANENT_BYTES(85) instead of IN_BYTES(1) and IN_PERMANENT_PKTS(86) instead of IN_PKTS(2). In these cases Entuity IFA compares the current absolute value with the previous value and calculates the difference to return the relative value. Therefore the first sample is always set to 0 and discounted.

- *dbDriver*, identifies the database driver.
- *dbUrl*, specifies the flow database.
- *dbUser*, name of the root database account.
- *dbPwd*, password for the root database account.
- *partition1_maxCount*, maximum number of flows allowed in the buffer before they get written to the disk, if partition gets full, then flows get dropped. Set by default to 1000000.
- *ageOutFlows1*, the number of minutes to keep flows in the database, by default 65.
- *ageOutRuns*, he number of minutes to keep flow collector operational times in the database, set by default to 1500.
- *ageOutStats*, number of minutes to keep flow collector statistics in the database, set by default to 1500.
- *trace*, indicates whether to log the details of flow records as they are parsed and distributed. When set to:
  - **0** (default), disable tracing
  - **1**, enable tracing.
- *packetLogging*, indicates of whether to dump binary flow packets to file. This file can later be used to replay the packets back to the flow collector, replay packets are never logged. When set to:
  - **off** (the default), packets are not logged
  - **all**, all incoming packets are logged
  - **unknown**, log only packets which are not recognized by the flow collector.

Status

Changes made to this file are included to the server configuration, however changes to this file are not maintained during Entuity upgrades. You should specify your flow configuration customizations in *entuity_home*\`etc\flowcfg.properties`.

Maintained by Entuity.

flowcfg.properties

Location

entuity_home/etc

Format

Text file containing customizations to the configuration of Entuity IFA flow collectors.

Description

Entuity IFA flow collectors are shipped with a factory configuration suitable for most network environments. You can amend this configuration, for example specify more than one port for Entuity to accept flow data, increase the size of the buffer handling incoming flow packets.

You should create `flowcfg.properties` by making a copy of the template file `flowcfg-template.properties`. The template file contains descriptions and examples of configuration options which you can edit.

When set, port values in `flowcfg.properties` take precedence over the values set in `flowcfg-template.properties` and those entered during `configure` and stored in `entuity.cfg`.

Status

Changes made to this file are included to the server configuration, and are maintained during Entuity upgrades. Entuity automatically discovers changes in `flowcfg.properties`, you do not have to run `configure` to apply updates.

flow-applications-template.txt

Location

entuity_home/etc

Format

Text file derived from a version of the application to port mapping file retrieved from http://www.iana.org/assignments/port-numbers.

Description

This file maps port numbers and network protocol to application names and descriptions. These port to application mappings are only used by the Entuity Integrated Flow Analyzer (IFA). When a connection is made from a client to a server the TCP/UDP port on the server end of the connection determines the application in use. The port number allocated to the client end of the connection is referred to as an ephemeral port and has no meaning. Entuity determines which end of a connection is the server end so that its port number can be used to identify the application, by:

1) Considering ports < 1024 as having the highest priority, regardless of whether the other port is in the mapping file or not.

Ports below 1024 are reserved port numbers, and so only one port (either the source or the destination port) should be in the range.

2) Where both ports are greater than 1023, or, more unlikely, both are below 1024 Entuity uses the first port mapping in `flow-applications-template.txt`.

System Administrators can amend and add new mappings to this file, and then include them to the Entuity database using `flowCollector.bat`.

If a port is mapped to two applications, Entuity resolves this conflict by using the last mapping for that port-protocol combination in the file.

This extract shows the port to application mapping for port 80:

```
ttp             80/tcp    World Wide Web HTTP
http            80/udp    World Wide Web HTTP
www             80/tcp    World Wide Web HTTP
www             80/udp    World Wide Web HTTP
www-http        80/tcp    World Wide Web HTTP
www-http        80/udp    World Wide Web HTTP
```

where:

- *www-http*, is the last entry for the port 80 and TCP combination, and is therefore the name Entuity uses for the application.
- *80/tcp*, identifies the port number and protocol. Entuity Integrated Flow Analyzer supports UDP and TCP protocols.
- *World Wide Web HTTP*, is the application description. Entuity Integrated Flow Analyzer does not use the application description.

Status

Maintained by the System Administrator.

flow-exclusions.properties

Location

entuity_home/`etc`

Format

Text file containing configurations to exclude flow data from Entuity IFA flow collectors.

Description

Exclusion filters allow you to exclude data based on source and destination IP addresses and/or source and destination ports. You can enter exact values, or use wild cards to create more extensive filters.

You should specify your exclusion filters in *entuity_home*\etc\flow-exclusions.properties, on each server acting as a flow collector.

You specify exclusion filters:

- On the endpoint, so flows outgoing from or incoming to the specified endpoint are filtered out.

  ```
  IPAddressPattern : PortPattern
  ```

- that are unidirectional, so flows which originate from the specified source endpoint and end at the specified destination endpoint are filtered out.

  ```
  SrcIPAddressPattern : SrcPortPattern > DstIPAddressPattern : DstPort-Pattern
  ```

- that are bidirectional, so flows in both directions between two endpoints are filtered out:

  ```
  IPAddressPattern1 : PortPattern1 = IPAddressPattern2 : PortPattern2
  ```

An *IPAddressPattern* can be one or more IP address or range of IP addresses. These are examples of valid patterns:

- matches a single IP address:

  ```
  10.44.1.101
  ```

- matches all IP addresses within the range:

  ```
  10.44.1/24
  ```

- an asterisk matches all IP addresses:

  ```
  *
  ```

A *PortPattern* can be one or more port numbers, or range of port numbers. These are examples of valid patterns:

- matches a single port:

  ```
  3066
  ```

- matches all ports within the range:

  ```
  2048-2099
  ```

- an asterisk matches all ports, equivalent to 0 to 65535:

  ```
  *
  ```

These are example exclusion filters:

- Filter all flows going from or to applications on port 3306 on 10.44.1.101 host

  ```
  10.44.1.101:3306
  ```

- Filter all flows going from or to applications (ports 3306, 1433) on any of listed hosts

  ```
  10.44.1.101, 10.44.1.102 : 1433, 3306
  ```

- Filter all flows going from host 10.44.1.101 to host 10.44.1.10

  ```
  10.44.1.101:* > 10.44.1.10:*
  ```

- Filter all flows between host 10.44.1.101 and host 10.44.1.10

  ```
  10.44.1.101:* = 10.44.1.10:*
  ```

Status

Created and maintained by System Administrator.

flow-exclusions-template.properties

Location

entuity_home/etc

Format

Text file containing commented out examples of how to exclude flow data from Entuity Integrated Flow Analyzer collectors.

Description

Exclusion filters allow you to exclude data based on source and destination IP addresses and/or source and destination ports. You can enter exact values, or use wild cards to create more extensive filters.

Status

Changes made to this file are included to the server configuration, however changes to this file are not maintained during Entuity upgrades. You should specify your exclusion filters in *entuity_home*\etc\flow-exclusions.properties.

Maintained by Entuity.

flowUserDefGroups.xml

Location

entuity_home/etc

Format

Text file containing a commented out example of how to define user defined groups for flows managed by IFA.

Description

This file contains an example of how you can define user defined groups for flows managed by IFA. Definition of custom data types, for example Location, Department, Customer, whose members, for example UK, US, Dev, Sales, Customer A, Customer B are defined in terms of the available raw data types. This example is synonymous with custom groups and group based analysis.

Each user defined group is structured as a bean definition, with these properties:

- *name*, a unique name for each group definition. Duplicate names will result in an error.
- *displayName*, the textual description shown to user for the group.
- *unmatchedName*, an optional set name where it will be mapped to this name if any of the filter criteria is not met.

- *unmatchedDisplayName*, an optional set display name which is shown to the user for an unsatisfied match.
- *userSets*, a list of set definitions where matching need to be done. Each set in the list is structured as bean definition. The set has these properties:
  - *name*, a unique name for each set that is defined in a group. Duplicate names will result in an error.
  - *displayName*, a textual description shown to user for the set.
  - *expression*, an SQL type expression which flows must meet to be included in the set.

This sample configuration includes custom group definitions:

```
<beans xmlns="http://www.springframework.org/schema/beans"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:util="http://www.springframework.org/schema/util"
   xsi:schemaLocation="http://www.springframework.org/schema/beans
   http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
   http://www.springframework.org/schema/util http://www.spring-
   framework.org/schema/util/spring-util-2.0.xsd">
<bean class="com.entuity.flows.UserDefinedGroup">
   <property name="name" value="My_Apps" />
   <property name="displayName" value="My Applications" />
   <property name="unmatchedName" value="Not_Web" />
   <property name="unmatchedDisplayName" value="Not Web" />
   <property name="userSets">
   <list>
      <bean class="com.entuity.flows.UserDefinedSet">
       <property name="name" value="Web_Requests" />
       <property name="displayName" value="Web Requests" />
       <property name="expression" value="dstPort in (80,443,8080)" />
      </bean>
      <bean class="com.entuity.flows.UserDefinedSet">
       <property name="name" value="Web_Responses" />
       <property name="displayName" value="Web Responses" />
       <property name="expression" value="srcPort in (80,443,8080)" />
      </bean>
      <bean class="com.entuity.flows.UserDefinedSet">
       <property name="name" value="Web_Traffic" />
       <property name="displayName" value="Web Traffic" />
```

```
        <property name="expression" value="dstPort in (80,443,8080) or
    srcPort in (80,443,8080)" />
        </bean>
      </list>
    </property>
    </bean>
    <bean class="com.entuity.flows.UserDefinedGroup">
      <property name="name" value="IFS" />
      <property name="displayName" value="All Interfaces" />
      <property name="userSets">
        <list>
          <bean class="com.entuity.flows.UserDefinedSet">
           <property name="name" value="IF_3" />
           <property name="displayName" value="if 3" />
           <property name="expression" value="ifIn eq 3 or ifOut eq 3" />
          </bean>
          <bean class="com.entuity.flows.UserDefinedSet">
           <property name="name" value="IF_4" />
           <property name="displayName" value="if 4" />
           <property name="expression" value="ifIn eq 4 or ifOut eq 4" />
          </bean>
          <bean class="com.entuity.flows.UserDefinedSet">
           <property name="name" value="IFS_OTHER" />
           <property name="displayName" value="if not 3 or 4" />
            <property name="expression" value="not (ifIn in (3, 4) or
    ifOut in (3, 4))" />
        </bean>
       </list>
      </property>
      </bean>
    </beans>
```

Status

Maintained by the System Administrator, only an example is shipped with Entuity.

forkevent.cfg

Location

entuity_home/etc

Format

Text file containing configuration information for Entuity Event Forwarding.

Description

`forkevent.cfg` is the event forwarding configuration file. It includes sample configurations and instructions for their activation.

Event Forwarding functionality is included with the standard Entuity installation, and is installed but not activated, on the Entuity server. Event Forwarding requires Entuity and the receiving third party software are installed and running, with permitted communication between the two.

`ForkEvent` is an Event Forwarding executable, and is installed to:

entuity_home`/integ/ForkEvent/`

[connection]

```
[connection]
username=admin
view=All Objects
eventsPerBatch=100
extendedEvents=0
```

where:

- *[connection]* is the name of the section that contains the details required to access Entuity event data.
- *username* is the Entuity login name.
- *view* is the Entuity view from which events are collected. Only when an event occurs on a device within the defined view is it forwarded by ForkEvent.
- *extendedEvents* sets the maximum number of characters that `forkevent` forwards for the event description. Event descriptions greater than this setting are truncated. When set to:
  - 0 (default), forwards event descriptions to a maximum of 127 characters.
  - 1, forwards event descriptions to a maximum of 4095 characters. Extended event descriptions are not currently stored in the Entuity database.

Status

Maintained by the System Administrator. Entuity does not maintain user changes to this file during Entuity upgrades.

hostFiles

Attempt to use a hosts file. these live in ENTUITY_HOME\etc\hostFiles directory.

httpd_eye.conf

Location

entuity_home/lib/apache/conf/

entuity_home/install/template/lib/apache/conf

Format

Text file containing configuration information for the Apache web server. For further information on this type of configuration file (default name `httpd.conf`) refer to the Apache documentation which is available from the Apache website:

http://www.apache.org/

Description

`httpd_eye.conf` includes all of the information required by the Apache web server to deliver the Entuity web UI and the RESTful API.

entuity_home/lib/apache/conf/httpd_eye.conf is generated when you run `configure` from the template file, *entuity_home*/install/template/lib/apache/conf/httpd_eye.conf.

If you want to amend the HTTPD configuration of the Apache webserver, for example to reconfigure port numbers or amend log file settings, you should:

- Not amend *entuity_home*/lib/apache/conf/httpd_eye.conf as the next time you run `configure` a new version of this file is generated from the template file and all of your changes would be lost.
- Amend *entuity_home*/install/template/lib/apache/conf/httpd_eye.conf. You will then have to stop Entuity and run `configure` to generate a new version of *entuity_home*/lib/apache/conf/httpd_eye.conf. When you start Entuity then Apache will run using the updated version of `httpd_eye.conf`.

Status

Maintained by the System Administrator. Entuity does not maintain user changes to either versions of this file during Entuity upgrades.

installed_modules.cfg

Text file containing a record of installed modules and their current enabled and visibility states. The default states for each module is initially inherited from `module_definitions.cfg` via `configure`, however when you amend these default states the changes are held here. Where there is a conflict between settings in

`installed_modules.cfg` and `module_definitions.cfg`, `installed_modules.cfg` settings take precedence.

Entuity `configure` references this file when reconfiguring an existing Entuity installation, so the Modules Configuration page displays the current active/inactive status for each module. `configure` also updates `installed_modules.cfg`.

Each Entuity module installed status is defined through its own section:

```
[moduleDefinition autonomous_WAP]
isEnabled=1
isHidden=0
[moduleDefinition Global_Search]
isEnabled=0
isHidden=1
[moduleDefinition Auto_Device_Renaming]
isEnabled=0
isHidden=1
```

where:

- *[moduleDefinition autonomous_WAP]*, is the internal module section name. This section name must match that of the module defined in `module_definitions.cfg`.
- *IsEnabled* when set to:
  - **0**, indicates the module is not enabled
  - **1**, is enabled.

  A module can be enabled but Entuity only activates that module when its license supports that module. During `configure` the Module Configuration page indicates the license status for each module, you can also check license status through checkLicense.
- *IsHidden* when set to:
  - **0**, indicates the module is listed in the Modules Configuration page during Entuity `configure`.
  - **1**, indicates the module is not listed in the Modules Configuration page during Entuity `configure`.

Status

Maintained by `configure` and the Entuity System Administrator.

known_hosts.txt

Location

entuity_home/`etc`

Format

Setup for SSH use with Entuity Configuration Management.

Description

Contains all of the host public keys. Each line comprises of a:

- *host*, resolved name or IP address Entuity uses to access the device.
- *algorithm* used to derive the signing and verification encryption key, i.e. DSS, RSA.
- *host fingerprint*, host fingerprint.

For example:

```
10.44.5.157 ssh-rsa 0e:18:a1:03:53:ed:22:e3:b7:ce:2c:bf:3a:49:c9:7a
10.44.5.156 ssh-dss 10:cb:0f:2b:f5:ce:3d:78:da:92:f1:3a:63:ce:5e:56
```

How Entuity Configuration Management enforces SSH security is dependent upon the policy applied to the Entuity Script Engine. (See *scriptEngine-template.properties*.)

Status

Maintained by the System Administrator.

license.dat (license file)

Location

entuity_home/etc

Format

Internal use only.

Description

Contains coded information about the Entuity managed object credits and expiry dates. The license file, by default `license.dat`, is used by `licenseSrvr` and can be checked using `checkLicense`.

Entuity is shipped with an evaluation license which can only be used for a limited period and should only be used in a test environment.

Status

Provided by an Entuity representative.

Maintained by the System Administrator.

mib.txt

Location

entuity_home/etc

Format

Internal use only.

Description

Contains information relating to ASN (Abstract Syntax Notation). The file is used by various SNMP polling processes, including `prole`, as well as by `snmpget` and `snmpwalk`.

Status

Read only.

module_definitions.cfg

Location

entuity_home/`etc`

Format

Internal use only.

Description

Module definition file provides a central location for the definition of modules available with Entuity. Entuity `configure` references this file when listing available modules and during configuration.

Where there is a conflict between settings in `installed_modules.cfg` and `module_definitions.cfg`, `installed_modules.cfg` settings take precedence. During an upgrade `module_definitions.cfg` is overwritten.

```
[moduleDefinition autonomous_WAP]
displayname=Autonomous WAP
typeLicenses=
 =AutonomousWap,
 =AwapHostCountHiThreshold,
 =AwapHostCountLoThreshold,
 =WirelessPort,
 =Wlan,
productLicenses=
configFiles=
 =sw_device_awap.cfg,
 =sw_port_wireless.cfg,
reportSystemConfigFiles=
 =sw_report_system_wireless_access_point.cfg
menuDefConfigFiles=
exoticaFiles=
 =Cisco-c1130+AWAP.vendor,
 =c1200+AWAP.vendor,
```

```
deprecatedConfig=
dataLossWhenDisabled=1
isEnabledByDefault=0
isHidden=0
```

where

- *[moduleDefinition autonomous_WAP]*, is the internal module section name
- *displayname*, module name used within Entuity, e.g. on the Module Configuration panel during configuration.
- *typeLicenses*, the StormWorks types that must be licensed for successful module activation.
- *productLicenses*, the product license required to run the module.
- *configFiles*, the StormWorks configuration files through which module functionality is defined.
- *reportSystemConfigFiles*, the StormWorks configuration files through which any module Flex reports are defined.
- *menuDefConfigFiles*, the StormWorks configuration file(s) through which any module user actions are defined.
- *exoticaFiles*, the vendor device definition files associated with the module.

  When a module is enabled `configure` copies these files from their reference folder, *entuity_home*\`etc\exotica` to *entuity_home*\`etc`, if subsequently disabled `configure` deletes these files from e*ntuity_home*\`etc`.

  You can use exotica files without activating the module, although you must rename them to prevent `configure` automatically deleting them the next time it is run. Entuity recommend replacing the plus sign (**+**) in exotica file names with an underscore (_), for example `Cisco-c1130+AWAP.vendor`, to `Cisco-c1130_AWAP.vendor`.

- *deprecatedConfig*, references to deprecated files that are still being used to include this module to Entuity. These files should not be included to the configuration. (See the *Entuity Migration Guide*.)
- *dataLossWhenDisabled*, when set to:
  - **0**, prevents `configure` from displaying a warning that disabling of the module will result in loss of data already collected by that module.
  - **1** (default), sets `configure` to display a warning that disabling of the module will result in loss of data already collected by that module.
- *isEnabledByDefault*, when set to:
  - **0**, indicates the module is not enabled
  - **1**, indicates the module is enabled.

  This setting can be overridden by *IsEnabled* in `installed_modules.cfg`.

  A module can be enabled but Entuity only activates that module when its license supports that module. During `configure` the Module Configuration page indicates the license status for each module, you can also check license status through checkLicense.

- *IsHidden* when set to:
  - **0**, indicates the module is listed in the Modules Configuration page during Entuity `configure`.
  - **1**, indicates the module is not listed in the Modules Configuration page during Entuity `configure`.

  This setting can be overridden by *IsEnabled* in `installed_modules.cfg`.

Status

Read only.

newbin.vendor

Location

entuity_home/`etc`

Format

Internal use only.

Description

Deprecated, retained for backward compatibility.

Status

Deprecated. Read-only.

nominal_power.cfg

Location

entuity_home/`etc`

Format

```
[Device Cisco 5505]
SysOID=.1.3.6.1.4.1.9.5.34
NominalPowerWatts=800
Reference=005, 006
```

where:

- *Device* is a unique name identifying the device.
- *SysOID* is the device system OID.
- *NominalPowerWatts* is the estimated power consumption of the object.
- *Reference*, identifies the device. It is also used by modules to make the device - module association.

```
[Module WX-X5530]
```

```
NominalPowerWatts=376
Reference=006
```

- *Module* is a unique name identifying the module.
- *NominalPowerWatts* is the estimated power consumption of the object.
- *Reference*, associated the module with its device.

Description

Identifies a device or module through their system OID, and then maps the object to a nominal power consumption value. Nominal power values are used with the Entuity Green IT Perspective functionality, for example the Green IT Perspective dashboard includes estimates of power consumption in your network and potential for savings.

Status

Read only. When you want add your own power configurations include them to `site_specific_nominal_power.cfg`.

provost.conf

Location

entuity_home/`etc`

Format

Internal use only.

Description

Configuration file for the main scheduling process, `provost` (see *Chapter 2 - Entuity System Processes and Utilities*).

Status

Read only.

scriptEngine-template.properties

Location

entuity_home/`etc`

Format

This is a template file. When changing the default behavior of the Script Engine you should copy this file and rename it to `script_engine.properties`. You can then amend the settings in `script_engine.properties`. You must restart Entuity to apply any changes in `script_engine.properties` to the Script Engine.

This is an example extract:

```
jmxFile=C:/Entuity/log/scriptEngineURL.jmx
```

```
host_verification_policy=RELAXED
known_hosts=C:/Entuity/etc/known_hosts.txt
thread_pool_size=10
output_buffer_size=100000
script_cache_size=400
```

where:

- *jmxFile*, is the file where the URL to the JMX agent can be found.
- *host_verification_policy* which can be:
  - RELAXED, a connection is accepted only if there is no entry in the known hosts file corresponding to the peer or there is an entry and its fingerprint matches the fingerprint sent by the remote host. If there was no entry then a new entry with the received fingerprint will be created. The known hosts file is required and it will be updated with new host entries upon successful termination of the program.
  - ENFORCING, a connection is made only if a corresponding valid entry is found in the known hosts file.
  - PERMISSIVE, any connection is accepted.
- *known_hosts* is the name and location of the known hosts file.
- *thread_pool_size* is the maximum number of script processing threads, i.e. the Script Engine can only concurrently process as many scripts as there are threads. This count is shared by all users on the local server and is by default set to 10. When the limit is reached TomCat queues tasks until a thread becomes available.
- *output_buffer_size* is the maximum length of the script output. If the output is longer than 100000 characters then the initial characters are truncated.
- *script_cache_size* size of the cache used to hold scripts. It should only be amended after consultation with your Entuity representative.

Description

An example configuration file for the Script Engine.

Status

Read only.

security.cfg.xml

Location

entuity_home/etc

Format

Entuity System Administrator can create this file from the supplied template file, `security_template.cfg.xml`. The template file includes extensive notes to aid successful configuration. Entuity recommend updating this internal documentation when implementing authentication.

Description

Main configuration file for Entuity authentication. Each section within the file configures a module.

Authentication module

This section configures the main authentication service behavior.

```
<module name="Authentication">
<authentication internal="true" sso="memory" externalAuth-
Handler="com.entuity.security.external.ldap.LdapLogon" allowSuperUser-
Access="true" />
</module>
```

where:

- *internal*, is a mandatory attribute which specifies that kind of authentication that must be used. When set to:
  - **true**, Entuity uses its internal authentication mechanism (default value)
  - **false**, Entuity uses its external authentication mechanism. When authentication service is configured to use external authentication, then *externalAuthHandler* attribute must also be set and *ExternalAttributesMapping*, *ldap-config* and *ServerAccess* sections must also be configured.
- *externalAuthHandler*, specifies authentication module implementation. It must be present when authentication service is configured to use external authentication, otherwise this value is ignored.

  Default value is **com.entuity.security.external.ldap.LdapLogon**
- *allowSuperUserAccess*, controls whether access to a server should be allowed in an emergency situation. An emergency situation occurs if a security database could not be accessed or if a service is configured to use external authentication and the external authentication server is not accessible.

  When set to:
  - **true**, super users can to access this server in emergency situation (default value)
  - **false**, super user access to this server is disabled.

CentralDB Module

Connection properties for central security database.

```
<module name="CentralDB">
<database host="localhost" port="3306" username="root"
password="5742888A8EBD135553E6001F6442873B" />
</module>
```

where:

- *host*, host name or IP address of the host on which the Entuity database is running. When not specified localhost is used.

- *port*, the port number on which the Entuity database is listening. Optional parameter, with a default value of 3306.
- *username*, name of the user to connect to the Entuity database server. Database server must be configured to accept connections for that user from this host. This is a mandatory parameter.
- *password*, password for the user specified in *username*. If not present, then empty password is used. However, Entuity strongly recommend user accounts are set up with passwords.

If the central database resides on another host (not localhost), Entuity recommend setting up a special user on that database and allow this user to connect from this specific host and/or other hosts that use that central database.

LocalDB Module

Connection properties for a local Entuity database. This connection is used to locate and administer super users.

```
<module name="LocalDB">
<database host="localhost" port="3306" username="root"
password="5742888A8EBD135553E6001F6442873B" />
</module>
```

where:

- *host*, host name or IP address of the host on which the Entuity database is running. When not specified localhost is used.
- *port*, the port number on which Entuity database is listening. Optional parameter, with a default value of 3306.
- *username*, name of the user to connect to the Entuity database server. Database server must be configured to accept connections for that user from this host. This is a mandatory parameter.
- *password*, password for the user specified in *username*. If not present, then empty password is used. However, Entuity strongly recommend user accounts are set up with passwords.

ExternalAttributesMapping

This section specifies how different attributes returned from an external authentication system map to Entuity groups. These groups will be assigned to the authenticated user, with permission being set through grant and revoke rules.

Each rule:

- May have a list of groups to grant or revoke access
- May include conditions, which control when the rule is applied. When a condition is not specified or is empty, then the rule is applied unconditionally.
- Is applied in the order specified in the configuration. You can order grant and revoke rules

as required to suite specific needs.

This example configuration grants members of the network domain user group Technical Support, membership of the Entuity user group Administrators.

```
<module ignorecase="true" name="ExternalAttributesMapping">
<grant name="Admin groups">
   <group name="Administrators" />
   <condition>
      <attr name="groups" contains="Technical Support" />
   </condition>
</grant>
```

where:

- *ignorecase* when set to
  - **true**, external authentication service is case insensitive, and so is recommended fro Windows environments.
  - **false**, external authentication service is case sensitive.

  This flag also affects condition evaluation, as text equality tests are done with reference to this flag. So if you set this flag to **false**, then be careful to enter condition values in exactly the same casing as returned from your external authentication server.
- *grant* is the rule type, it can also be *revoke.*
- *group* name is the an Entuity user group name, e.g. Administrators, that members of the subsequently named network domain groups will be a member of.
- *condition* specifies the rule condition, this can include one or more attributes:
  - *attr name* is the attribute name, e.g. **groups** refers to the network domain user group.
  - *attr contains* specifies the network user group name.

ldap-config Module

This section is only required when configuring Entuity to use Active Directory as an external authentication service.

Entuity include to the template file, `security_template.cfg.xml`, a number of example ldap configurations. Entuity recommend that when you create `security.cfg.xml` you delete from `security.cfg.xml` most of the example configurations and only retain those you want to amend for your installation. This will improve the readability of the file.

The example ldap-config module is for use with Active Directory external authentication that does not require the user to enter a domain name in the logon screen.

```
<module name="ldap-config">
<ldap>
<userBindNameIsDN>false</userBindNameIsDN>
```

```
<userBindName>{1}@ENT</userBindName>
<userSearchBaseCtxDN>ou=Subset, ou=Users, ou=Live, ou=Migration,
dc=entuity, dc=local</userSearchBaseCtxDN>
<userMatchFilter>(sAMAccountName={1})</userMatchFilter>
<property name="java.naming.provider.url" value="ldap://entlondc01" />
</ldap>
</module>
```

where:

- *userBindNameIsDN*, bind name for the user is not distinguished name.
- *userBindName*, bind name for the user will be in format <username>@ENT, where:
  - <username> is entered by user at logon.
  - ENT, must be changed to your domain name.
- *userSearchBaseCtxDN*, specifies location in the directory where to search for the user. User entry must reside below this path.
- *userMatchFilter*, if a user's bind name is not specified as a distinguished name, then this element must be present and with a search criteria to find the user. You may use placeholders in the criteria.
- *property value*, the address of the LDAP server. You can use LDAPs scheme instead of ldap to establish SLL secure connections. You can also specify the port, for example ldap://myserver:1233.
- Placed values, numbers in curly brackets {}, are replaced with values entered by the user. These are valid numbers and corresponding replacement values:
  - {0}, replaced by value user enters in logon screen. It could be just simple name or user name and domain name in UNC (\\domain\username) or UPN (username@domain) format.
  - {1}, replaced by username only without domain.
  - {2}, replaced by domain name - may be empty if not entered by user.
  - {3}, replaced by user's distinguished name and available only in user's group search.

Example Configuration: ldap-config-domain

This example configures Entuity to use Active Directory as an external authentication service, and you require the user to enter the domain name in the logon screen.

```
<module name="ldap-config-domain">
<ldap>
<userBindNameIsDN>false</userBindNameIsDN>
<userBindName>{1}@{2}</userBindName>
<userSearchBaseCtxDN>ou=Subset, ou=Users, ou=Live, ou=Migration,
dc=entuity, dc=local</userSearchBaseCtxDN>
<userMatchFilter>(userPrincipalName={1}@{2})</userMatchFilter>
<property name="java.naming.provider.url" value="ldap://entlondc01" />
```

```
</ldap>
</module>
```

Example Configuration: ldap-config-sun

This example configuration is a minimal configuration for use with Sun ONE Directory Server as an authentication service. Module configuration requires a user to enter a domain name at the logon screen.

```
<module name="ldap-config-sun">
<ldap>
<userBindNameIsDN>true</userBindNameIsDN>
<userBindName>uid={1}, ou=People, dc=example, dc=com</userBindName>
<userRefersToGroup>false</userRefersToGroup>
<groupSearchBaseCtxDN>ou=Groups,dc=example,dc=com</groupSearchBa-
seCtxDN>
<groupMatchFilter>(uniquemember={3})</groupMatchFilter>
<property name="java.naming.provider.url" value="ldap://
localhost:55495" />
</ldap>
</module>
```

where

- *userBindNameIsDN*, bind name for the user is in distinguished name format.
- *userBindName*, bind name for the user, in the format uid=<username>, ou=People, dc=example, dc=com where <username> is value entered by user at logon.
- *userRefersToGroup*, indicates the user entry in the directory does not refer to groups, instead group entries refer to user entries. Therefore, an additional search is required to find groups that refer to our user.
- *<username>* and *<domain>* are entered by user at logon.
- *groupSearchBaseCtxDN*, specifies location in the directory where to search for the group. Group entry must reside below this path.
- *groupMatchFilter*, specifies the search criteria for the groups, when a user entry matches the filter then the user is a member of the group.

Example Configuration: ldap-config-template

This section includes a configuration that includes all of the ldap-config options, one which is not tailored to a particular external authentication solution, unlike the other ldap-config examples.

```
<module name="ldap-config-template">
<ldap>
   <userBindNameIsDN>false</userBindNameIsDN>
   <userBindName>{1}@{2}</userBindName>
```

```
<lookupUserBindDNAsSystemUser>false</lookupUserBindDNAsSystemUser>
<userSearchBaseCtxDN>ou=Users, ou=Live, ou=Migration, dc=entuity,
dc=local</userSearchBaseCtxDN>
<userMatchFilter>(userPrincipalName={1}@{2})</userMatchFilter>
<searchGroupsAsSystemUser>false</searchGroupsAsSystemUser>
<systemUserName>cn=userwithsearchpriveleges, dc=example, dc=com</
systemUserName>
<systemUserPwd>password</systemUserPwd>
<userRefersToGroup>true</userRefersToGroup>
<userMemberOfAttrID>memberOf</userMemberOfAttrID>
<groupSearchBaseCtxDN>OU=Distribution Groups,OU=Company
Data,DC=entuity,DC=local</groupSearchBaseCtxDN>
<groupMatchFilter>(member={3})</groupMatchFilter>
<groupSearchDepth>5</groupSearchDepth>
<groupNameAttrID>cn</groupNameAttrID>
<property name="java.naming.provider.url" value="ldap://entlondc01"
/>
<property name="java.naming.factory.initial"
value="com.sun.jndi.ldap.LdapCtxFactory" />
<property name="java.naming.security.authentication" value="simple"
/>
<attemptAfterAuthError>false</attemptAfterAuthError>
</ldap>
</module>
```

where:

- *userBindNameIsDN*, indicates whether userBindName element is specified as a distinguished name or not. This is not always possible having username and domain name to construct distinguished name of the user's entry. For example, your server may be configured in such a way, that user's DN looks like:

  ```
  CN=FirstName LastName, DC=mydomain
  ```

  Values are false or true.

- *userBindName*, bind name for the user, in the format uid=<username>, ou=People, dc=example, dc=com where <username> is value entered by user at logon.
- *lookupUserBindDNAsSystemUser*, if a user's bind name is not specified as a distinguished name, then the authentication service must lookup the distinguished name. Lookup can be for the authenticating user, or the system user when using a secured directory. When set to:
  - **true**, then you need to specify *systemUserName* and *systemUserPwd* elements.
  - **false**, the default, the authentication service does not lookup the DN.

- *userSearchBaseCtxDN*, if a user's bind name is not specified as a distinguished name, then you must use this element to specify the directory under which search for the user should be done.
- *userMatchFilter*, if a user's bind name is not specified as a distinguished name, then this element must be present and with a search criteria to find the user. You may use placeholders in the criteria
- *searchGroupsAsSystemUser*, during a user's group search you may specify whether the search should be performed on behalf of an authenticated user or where there is a secured directory on behalf of the system user. When set to:
  - **false**, the default, the authentication service does not lookup the DN.
  - **true**, you must also specify *systemUserName* and *systemUserPwd*.
- *systemUserName* and *systemUserPwd*, specify system user name and passwords. These only require setting when *lookupUserBindDNAsSystemUser* and/or *searchGroupsAsSystemUser* are set to true.
- *userRefersToGroup*, indicates the user entry in the directory does not refer to groups, instead group entries refer to user entries. Therefore, an additional search is required to find groups that refer to our user. When set to:
  - true, the default, an additional search is required to find the groups that refer to users.
  - false, indicates user entry refers to groups.
- *userMemberOfAttrID*, if *userRefersToGroup* is true, then this element specifies the name of the attribute in the user or group entry which refers to the group. If this element is absent, than an assumed value of "memberOf" is taken. Defaults are Active Directory friendly.
- *groupSearchBaseCtxDN*, specifies the directory where a search for groups should be performed. This element must be present if *userRefersToGroup* element is **false**.
- *groupMatchFilter*, specifies the search criteria for a group search. This element must be present if *userRefersToGroup* element is **false**. You can use placeholders in this filter.
- *groupSearchDepth*, specifies the recursion level of the group search. This element is used if *userRefersToGroup* is **false**. The default value is 5.
- *groupNameAttrID*, specifies the name of the attribute on the group entry, which has value of the group name. Default value is **cn**, applicable for most LDAP schemas.
- *property*, specifies the address of the LDAP server. The format of the value is:

  ```
  <scheme>://<host>[:<port>]
  ```

  where:

  - <scheme> is ldap or ldaps (for SSL),
  - <host> is name or IP address of the LDAP server host
  - [:<port>] is the IP port for the LDAP server.

  For example:

  ```
  ldaps://myhost
  ```

- *property*, this element is optional and its value should not be changed.

- *attemptAfterAuthError*, this element is for use when multiple LDAP servers are providing authentication services. You should:
  - create an ldap configuration section for each set ldap server. These configurations should be numbered sequentially, i.e. ldap-config-1, ldap-config-2.
  - set *attemptAfterAuthError* from its default value of false to true:

    ```
    <attemptAfterAuthError>true</attemptAfterAuthError>
    ```

  Entuity attempts to connect to the first server using the first configuration, ldap-config-1. When there is an authentication error, not a connection error, Entuity attempts to connect to the next server using the next configuration, ldap-config-2. You can define as many servers as required.

ServerAccess

ServerAccess restricts access to Entuity server. You can deny access through the user's logon name, domain name or by Entuity user group membership. Server access is calculated by applying allow or deny rules, where the order of these rules is important.

By default any authenticated user is allowed.

This example section denies access to the server to members of the Entuity Test Group user group:

```
<module name="ServerAccess">
<serverAccess ignorecase="true">
<denyGroup name="Test Group"/>
</serverAccess>
</module>
```

These rule examples illustrate how you can control server access:

- only accepts users who are members of Administrators group, except user named **oldAdmin**

  ```
  <denyUser name="*" domain="*/># deny all users
  <allowGroup name="Administrators"/> # allow admins
  <denyUser name="oldAdmin"/># deny specific user
  ```

- allows access to all users:

  ```
  <allowUser name="*" domain="*"/>
  ```

- allows access to all users by group:

  ```
  <allowGroup name="*"/>
  ```

- denies access to a specific user from any domain:

  ```
  <denyUser name="test"/>
  ```

- denies access to a specific user from a specific domain:

  ```
  <denyUser name="test" domain="test2"/>
  <denyUser name="test" domain="test2.*"/>
  ```

- denies access to all users from specific domains:

```
<denyUser domain="test2"/>
<denyUser domain="test2.*"/>
```

- denies access to all users who are members of specific group:

```
<denyGroup name="Test Group"/>
```

AuthenticationService

This module defines Entuity's authentication service and must not be amended.

```
<module name="AuthenticationService">
<securedService>
<keyStoreName>auth_cert_store.jks</keyStoreName>
<keyStoreType>jks</keyStoreType>
<keyStorePwd>entuity</keyStorePwd>
<entryAlias>AuthenticationService</entryAlias>
<entryPwd>entuity</entryPwd>
</securedService>
</module>
```

PreferenceService

This module defines Entuity's preference service and must not be amended.

```
<module name="PreferenceService">
<securedService>
<keyStoreName>auth_cert_store.jks</keyStoreName>
<keyStoreType>jks</keyStoreType>
<keyStorePwd>entuity</keyStorePwd>
<entryAlias>PreferenceService</entryAlias>
<entryPwd>entuity</entryPwd>
</securedService>
</module>
```

UserManagementService

This module defines Entuity's user management service and must not be amended.

```
<module name="UserManagementService">
<securedService>
<keyStoreName>auth_cert_store.jks</keyStoreName>
<keyStoreType>jks</keyStoreType>
<keyStorePwd>entuity</keyStorePwd>
<entryAlias>UserManagementService</entryAlias>
```

```
<entryPwd>entuity</entryPwd>
</securedService>
</module>
```

TicketGrantingService

This module defines Entuity's ticket granting service and must not be amended.

```
<module name="TicketGrantingService">
<securedService>
<keyStoreName>auth_cert_store.jks</keyStoreName>
<keyStoreType>jks</keyStoreType>
<keyStorePwd>entuity</keyStorePwd>
<entryAlias>TicketGrantingService</entryAlias>
<entryPwd>entuity</entryPwd>
</securedService>
</module>
```

TGSConfig

This module defines Entuity's TGS configuration and must not be amended.

```
<module name="TGSConfig">
<tgsConfig>
<servicesKeyStoreName>auth_cert_store.jks</servicesKeyStoreName>
<servicesKeyStoreType>jks</servicesKeyStoreType>
<servicesKeyStorePwd>entuity</servicesKeyStorePwd>
<tgsHostAddr>localhost</tgsHostAddr>
</tgsConfig>
</module>
</application>
```

serverid.xml

Location

entuity_home/etc

Format

This file includes details that are used when identifying the Entuity server identity, this is most applicable when distinguishing between multiple Entuity servers.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:serverIdentity xmlns:ns2="http://www.entutity.com/webrpc">
   <id>ce333d40-fc09-42b6-a4dd-a0315ed3da20</id>
```

```
        <version>6.0.0.p0</version>
        <versionDisplay>Entuity 16.0</versionDisplay>
        <hostAddress>COMPRESSOR</hostAddress>
        <webPort>80</webPort>
        <sslAccess>false</sslAccess>

        <certificate>MIIChzCCAfCgAwIBAgIGARUD8xxFMA0GCSqGSIb3DQEBBQUAMIGHMS
        0wKwYDVQQhMDMxNWRhM2VkMjAxFDASBgNVBAsMC0RldmVsb3BtZW50MRAwDgYDVQQKD
        AdFbnR1a0NVoXDTE3MDkxMTEyMTY0NVowgYcxLTArBgNVBAMMJGNlNDg0ZDQwLWZjMD
        gtNDhiNi1hMWRkLWEwMzE1ZGEzZWQyMDEUMBIGA1UECwwLRGV2ZWxvcG1lbnQxEDAOB
        gNVBAoMB0VudHVpdHkxDzANBgNVBAcMBkxvbmRvbjEQMA4GA1UECAwHRW5nbGFuZDEL
        MAkGA1UEBhMCVUswgZ0wDQYJKoZIhvcNAQEBBQADgYsAMIGHAoGBAJCHNZjkkyWKl0H
        sGs72mfU44xoiKiOddCzkSIS2Bj2NL3Qs4tfWslVXaz+Q2PuF4/
        i3i5o8E4jJmZqHqTHaWK8KfGsE6y8eB470oh9ONnMxoFsd4YrUCntrd1X4mbVwvaa6E
        mbQVZgEDZXTZoo2BbfVyhJzA9ey4k2jKSkVLPuTAgEDMA0GCSqGSIb3DQEBBQUAA4GB
        AI2ykCawwzAZ2gfpjPCLymS0DMTDkhXgwc86trG6KnbRdpEpYpApx5I+N5eIaTEVj/
        tH0xBnrKPWnhCMiXqiLgqAsCZ80aPRNc9wPnxIMXdTIwUfeK0wPa+pNe5GyofUYZa-
        la8T4IpBqZy+JhGyLzF+0rSEuwVRoKzLeJQjO87gM</certificate>
    </ns2:serverIdentity>
```

where:

- *serverIdentiity*, web RPC of the Entuity server.
- *id*, unique Entuity server identifier.
- *version*, internal Entuity server version number.
- *versionDisplay*, Entuity server version number displayed through the Help About dialog in the Entuity client.
- *hostAddress*, Entuity server host name.
- *webPort*, Entuity server web port, by default 80.
- *sslAccess*, indicates whether the Entuity server is using SSL, **true**, or not, **false**.
- *certificate*, Entuity server certificate.

Description

This file includes details that are used when identifying the Entuity server identity, this is most applicable when distinguishing between multiple Entuity servers.

Status

Automatically generated by Entuity `install` and `configure`. System administrators can identify and change the *id* used with an Entuity server through `configure serverid`. Entuity maintains changes to this file during Entuity upgrades.

shutdown_policies.cfg

Location

entuity_home/etc

Format

```
[ShutdownPolicyGroup All Hosts]
IPAddressRange=0.0.0.0-255.255.255.255
Description=All Hosts
[ShutdownPolicyGroup London Office]
IPAddressRange=10.44.1.1-10.44.1.50, 10.44.1.60-10.44.1.90,
= 1.2.3.4-1.2.3.5, 10.44.1.98-10.44.1.123, 10.44.1.140-10.44.1.247
Description=Workstations in London Office
[ShutdownPolicyExclusion London Security Cameras]
IPAddressRange=10.44.1.10-10.44.1.12
Description=IP CCTV
```

where

- section header defines the:
  - *type*, **ShutdownPolicyGroup** for a policy group and **ShutdownPolicyExclusion** to specify devices and modules by IP address, that should be excluded from the policy group.
  - Name, name of the policy group, e.g. London Security Cameras.
- *IP Address Range*, specifies the IP addresses to include to, or exclude from, the policy group. For a contiguous IP address range enter the start and end addresses of the range separated by a dash. Where you want the range to be constructed from a number of component IP address ranges, comma separate each component.
- *Description*, name of the policy group that appears in Entuity, e.g. as a group to report on.

Description

Entuity Green IT allows you to group together devices and modules for which you want to apply the same energy policy. Policy group membership is determined by IP addresses, as are the exclusion groups.

Entuity recommend you configure policy groups and their exclusions through this file, where you have full add, amend and delete control.

Status

Maintained by the System Administrator. Entuity maintains changes to this file during Entuity upgrades.

site_specific_nominal_power.cfg

Location

entuity_home/etc

Format

```
[Device Cisco 5505]
SysOID=.1.3.6.1.4.1.9.5.34
NominalPowerWatts=800
Reference=005, 006
```

where:

- *Device* is a unique name identifying the device.
- *SysOID* is the device system OID.
- *NominalPowerWatts* is the estimated power consumption of the object.
- *Reference*, identifies the device. It is also used by modules to make the device - module association.

```
[Module WX-X5530]
NominalPowerWatts=376
Reference=006
```

where:

- *Module* is a unique name identifying the module.
- *NominalPowerWatts* is the estimated power consumption of the object.
- *Reference*, associated the module with its device.

Description

Identifies a device or module through their system OID, and then maps the object to a nominal power consumption value. Nominal power values are used with the Entuity Green IT Perspective functionality, for example the Green IT Perspective dashboard includes estimates of power consumption in your network and potential for savings.

This file is included to `nominal_power.cfg`, and so its configuration is included to Entuity. You can use this file to amend power configurations defined in `nominal_power.cfg`.

Status

Administrator maintained.

sn-example.cfg

Location

entuity_home/etc

Format

Maintained by Entuity.

Description

Example file for making connections to ServiceNow. You can copy this file and rename it to `sn.cfg`, and enter connection details for ServiceNow. The section name is used by the Send to ServiceNow action to call the connection details.

```
[default]
port = 443
host = myhost1.service-now.com
path = /api/now/table/em_event
user = Entuity
pass = ServiceNow
```

Where:

- `[default]` is the name of the connection. When setting up the Send to ServiceNow action you set `cname` to the name of the connection that you want to use.
- *port* is the port used by ServiceNow.
- *host* is the hostname of the ServiceNow instance.
- *path* is the location of the ServiceNow event table.
- *user* is the ServiceNow user name. The account must have the appropriate permission level, i.e. Event Management Administrator (evt_mgmt_admin), Event Management User (evt_mgmt_user), Event Management Integrator (evt_mgmt_integration).
- *pass* is the password to the ServiceNow user account.

snmpMaxPduOverrides.cfg

`snmpMaxPDUoverrides.cfg` includes a set of individual maximum PDU sizes for sysOids identified by Entuity Support as having a problem handling larger PDUs.

Location

entuity_home/`etc`

Format

Text file with each line identifying a sysOid and its maximum PDU size.

Description

Users can also enter individual max PDU size for specified sysOids. The format is:

```
<sysoid>=<Maximum PDU Size>
For example:
.1.3.6.1.4.1.9.1.669=512  # Cisco ASA5510
.1.3.6.1.4.1.9.1.670=512  # Cisco ASA5520
```

```
.1.3.6.1.4.1.9.1.671=512  # Cisco ASA5520sc
.1.3.6.1.4.1.9.1.672=512  # Cisco ASA5540
.1.3.6.1.4.1.9.1.673=512  # Cisco ASA5540sc
.1.3.6.1.4.1.9.1.745=512  # Cisco ASA5505
.1.3.6.1.4.1.9.1.753=512  # Cisco ASA5550
.1.3.6.1.4.1.9.1.763=512  # Cisco ASA5550sc
.1.3.6.1.4.1.9.1.764=512  # Cisco ASA5520sy
```

Status

Maintained by Entuity.

When upgrading Entuity this file is overwritten. When wanting to amend or add to these PDU size definitions you should create your own copy of this file and include it to Entuity through *entuity.cfg* for example:

```
snmpMaxPduSizeOverridesfile=snmpMaxPDUoveride.cfg
```

snmpV3.cfg

For Entuity to handle SNMPv3 traps from devices it must, as a minimum, know device name and user details. For devices Entuity manages, Entuity can retrieve the required information from its database. For devices Entuity does not manage you should enter identifying details in `snmpV3.cfg`.

Devices with duplicate engineIDs are not compliant with the SNMPv3 standard. However, some manufacturers do repeat engineIDs and Entuity supports this behavior. If devices have duplicate engineIDs and are sending DNMPv3 traps with privacy and/or authentication enabled they must use either the same credentials (passwords) or different user names.

Location

entuity_home`/etc`

Format

Text file, with each line defining information required to handle traps from a particular device.

Description

When you require Entuity to handle SNMPv3 traps from devices it does not manage, use this configuration file to specify how Entuity should handle these traps.

Each line details one device, and must include the device name and user and optionally engineID, authentication and privacy password.

For example:

```
-d 10.66.1.13 -u mark
```

```
-d 10.66.1.14 -e 0x80000312010A42010E -u mark -a MD5 -A "Auth
Password"
```

```
-d 10.66.1.15 -e 0x80000312010A42010F -u mark -a MD5 -A "Auth
Password" -x DES -X "Priv Password"
```

where:

- -d specifies the device name, e.g. 10.66.1.15.
- -u specifies the user name, e.g. mark.
- -e specifies the device engine, e.g. 0x80000312010A42010F.
- -a specifies the authentication protocol, i.e. MD5, SHA.
- -A specifies the authentication password, "Auth Password".
- -x specifies the privacy protocol, i.e. AES, DES.
- -X specifies the privacy password, e.g. "Priv Password".

startup_*o*/*s*.cfg

The startup configuration file; for Windows named `startup_WIN32.cfg` and for Linux systems `startup_UNIX.cfg`.

Location

entuity_home/`etc`

Format

Text file containing lines in the format: *systemVariable*=*value*, under headings denoted by square brackets [].

Description

Configuration file used by `starteots` when starting Entuity to determine which processes to start. For Windows implementations Entuity services are also configured here. Each process has its own section. Through the Entuity Health web page you can view the state and criticality of each process, this report identifies each process through its section name.

This is an example section:

```
[syslogger]
state=normal
type=command
start=${ENTUITY_HOME}${FPS}bin${FPS}syslogger
directory=${LOGDIR}
memorylimitmb=4000
is_critical=n
```

These are the available options:

- *[syslogger]*, is the section name enclosed in square brackets. Through the Entuity Health web page you can view the state of each process, this report identifies each process through its section name.
- *state* which sets the state(s) of the module. This label allows you to group modules by associating them with the same state. In the control_system section of `entuity.cfg`

you can set *defaultState*, to your chosen state. When Entuity starts all of those modules start.

For example by default Entuity starts all those sections with state set to normal. However, when reviewing collected data but not wanting to poll a network, e.g. when using Entuity for due diligence, you would use those sections that have state set to **maint**.

A section can have more than one state, each state separated by a comma, e.g.:

```
state=maint,normal
```

state is the only value you can amend. When state is set to **none**, the function always starts.

- *type*, indicates the type of function to start:
  - **command**, indicates *start* includes an instruction to run an executable.
  - **service**, indicates *start* includes an instruction to start a Windows service.
- *servicename*, name of the Windows service to start.
- *start*, includes the instruction used to start the process.
- *directory*, indicates the location of the log file, which when set to ${LOGDIR} is the log directory specified through *logdir* in `entuity.cfg`.
- *memorylimitmb* a Unix and Linux specific configuration setting. By default all processes are set to 4000 (4GB), except `dsKernelStatic` which is set to 8000 (8GB).
- *is_critical*, identifies whether the function is critical to Entuity core functionality, **Y**, yes and **N**, no. *is_critical* is displayed through the Entuity Health page.

The last line of the file must always be a reference to the site specific startup file:

```
!startup_WIN32_site_specific.cfg
```

Status

Maintained by Entuity.

When upgrading Entuity this file is overwritten. You should make any site specific changes to `startup_o/s_site_specifc.cfg`.

startup_o/s_site_specific.cfg

The site specific startup configuration file; for Windows named `startup_WIN32.cfg` and for Linux systems `startup_UNIX.cfg`.

Location

entuity_home/`etc`

Format

Text file containing lines in the format: *systemVariable=value*, under headings denoted by square brackets [].

Description

This file is referenced by `startup_o/s.cfg`. It is where you should enter site specific configuration settings for your installation startup. Values entered here override values for the same settings entered in by `startup_o/s.cfg`.

You can copy an entire section from `startup_o/s.cfg` to this file and amend its settings.

When you only want to amend a small part of an existing startup section, then you can add the section name and just the required attribute(s). This makes it easier to identify your changes. For example when you want to amend the state of Remedy, in `startup_o/s.cfg` the full section is:

```
[remedy]
state=none
type=command
start=${ENTUITY_HOME}\integ\ForkEvent\forkevent
${ENTUITY_HOME}\etc\remedyforkevent.cfg pipe_remedy
directory=${LOGDIR}
is_critical=n
```

In `startup_o/s_site_specific.cfg` you can enter:

```
[remedy]
state=normal
```

Status

Maintained by the System Administrator. When upgrading Entuity this file is preserved.

start_run_*manufacturer*.expect

Location

entuity_home/integ/SCRAPE

Format

A text file containing an Expect script that specifies the transfer of device configurations.

Description

Entuity Configuration Management includes these example scripts:

- `start_run_cisco.expect`
- `start_run_hp.expect`
- `start_run_juniper.expect`.

Scripts can be associated with individual devices through the web UI.

Status

Examples are created and maintained by Entuity. System administrators can develop their own scripts.

sw.cfg

Location

entuity_home/etc

Format

Text file containing lines in the format: *systemVariable*=*value*, under headings denoted by square brackets [].

Description

This is the main StormWorks configuration file and must not be edited. It also contains references to secondary configuration files, particularly sw_common.cfg. sw_common.cfg also contains secondary files all pre-fixed by sw_, that contain details regarding specific Entuity StormWorks services, e.g. events, ip peering. These files also must not be edited.

When Entuity configure is run sw.cfg (and through it the secondary files) is referenced and the StormWorks services are configured.

Status

Created and maintained by Entuity.

sw.cfg, sw_common.cfg and the sw_name.cfg files must only be edited by an Entuity representative, or under guidance of Entuity. Incorrect amendments of these files can seriously impact Entuity's performance.

sw_common.cfg

Location

entuity_home/etc

Format

Text file containing lines in the format: *systemVariable*=*value*, under headings denoted by square brackets [].

Description

This is the main StormWorks configuration file included to sw.cfg and must not be edited. It also contains references to secondary configuration files, all pre-fixed by sw_, that contain details regarding specific Entuity StormWorks services, e.g. events, ip peering. These files also must not be edited.

Status

Created and maintained by Entuity.

`sw.cfg`, `sw_common.cfg` and the `sw_name.cfg` files must only be edited by an Entuity representative, or under guidance of Entuity. Incorrect amendments of these files can seriously impact Entuity's performance.

sw_iptosysname.cfg

Location

entuity_home/`etc`

Format

Text file containing lines in the format: *systemVariable*=*value*, under a heading denoted by square brackets [].

Description

This is the scheduling definition for running `iptosysname`, which changes within Entuity device names to system names.

The default configuration is:

```
[Job jobIpToSysName]
Description=Job to change device names to be sysNames
Interval=86400
Offset=10800
ClientData=
Modes=normal
Method=simple;variable workdir=concat(get_config_var("entuity_home"),
"\\lib\\tools");
      =logMessage(concat(piped_exec("iptosysname",
workdir,0,7200000,""),"\n"))
```

where:

- *Job*, identifies the section as one that defines a job to change device names within Entuity from IP address to sysname.
- *Interval*, time in seconds between running of the job. The default is 86400, one day.
- *Offset*, defines when the job runs as an offset from 00:00. the default is 10800, equivalent to 03:00.
- *Client Data* and *Modes* should not be amended.
- *Method*, defines the job and should not be amended.

Status

Created and maintained by Entuity. This file is only enabled when included to `sw_site_specific.cfg` and `configure` is then run.

sw_menu_def_site_specific.cfg

Location

entuity_home/`etc`

Format

Text file containing references to files that specify Extensible Menus. Files with a hash before their name are not included to the Entuity configuration, for example:

```
#!sw_menu_discover_all.cfg
```

File names that prefixed with an exclamation mark are included to the Entuity configuration:

```
!sw_menu_discover_all.cfg
!sw_menu_example.cfg
```

Currently you can include these menu definitions to Entuity:

- `sw_menu_discover_all.cfg`, should be included to the configuration Entuity options for acting on Discovered Devices.
- `sw_menu_example.cfg`, these are a set of useful example user actions that can also provide the basis for more advanced customizations.

Description

This is the `StormWorks` configuration file to which the configuration files of user specific Extensible Menus are included.

When Entuity `configure` is run `sw.cfg` (and through it the secondary files, including `sw_menu_def.cfg`) is referenced and the `StormWorks` services are configured.

Status

Created and maintained by Entuity. Administrators may be asked to include and exclude references to files when adding and removing modules and other functionality.

When upgrading Entuity this file is overwritten. You should ensure you have taken a backup so that can you refer to it when re-applying your site specific configuration.

sw_module_file_list.cfg

Location

entuity_home/`etc`

Format

Text file containing references to files that specify activated Entuity modules. This file is created and maintained during configure. File names that prefixed with an exclamation mark are included to the Entuity configuration, for example:

```
!sw_green.cfg
!sw_swport_matrix.cfg
!sw_swport.cfg
!sw_swport_status.cfg
```

Description

This is the `StormWorks` configuration file to which the configuration files of Entuity modules are included. When Entuity `configure` is run `sw.cfg` (and through it the secondary files, including `sw_module_file_list.cfg`) is referenced and the `StormWorks` services are configured.

Status

Created and maintained by `configure`. When re-configuring or upgrading Entuity this file is overwritten.

sw_ph.cfg

Location

entuity_home/`etc`

Description

Controls parsing of the `StormWorks` configuration files and must not be edited.

Status

Created and maintained by Entuity.

sw_report_system_site_specific.cfg

Location

entuity_home/`etc`

Format

Text file containing references to files that specify extra reporting functionality, e.g. delivered with Entuity modules, customer specific modifications. Files with a hash before their name are not included to the Entuity configuration, for example:

```
#!sw_report_site_specific.cfg
```

File names that are only prefixed with an exclamation mark are included to the Entuity configuration:

```
!sw_report_site_specific.cfg
```

Description

This is the `StormWorks` configuration file to which extra reports are included, or more specifically their configuration files.

When Entuity `configure` is run `sw.cfg` (and through it the secondary files, including `sw_report_system_site_specific.cfg`) is referenced and the `StormWorks` services are configured.

Status

Created and maintained by Entuity. Administrators may be asked to include and exclude references to files when adding and removing site specific functionality.

When upgrading Entuity this file is overwritten. You should ensure you have taken a backup so that can you refer to it when re-applying your site specific configuration.

sw_site_specific.cfg

Location

entuity_home`/etc`

Format

Text file containing references to files that specify extra functionality, i.e. customer specific modifications. Files with a hash before their name are not included to the Entuity configuration, e.g.

```
#!sw_user_specific_function.cfg
```

File names that are prefixed with an exclamation mark are included to the Entuity configuration:

```
!sw_user_specific_function.cfg
```

Description

This is the `StormWorks` configuration file to which site specific functionality, specifically their configuration files are included.

When Entuity `configure` is run `sw.cfg` (and through it the secondary files, including `sw_site_specific.cfg`) is referenced and the `StormWorks` services are configured.

Status

Created and maintained by Entuity. Administrators may be asked to include and exclude references to files when adding and removing site specific functionality.

When upgrading Entuity this file is not updated, as you would lose your site specific settings. You should check the release notes as to whether the latest version of this file includes new functionality, or examine the file directly.

sw_user_defined_components.cfg

Location

entuity_home/etc

Format

Text file containing the definition of 20 pre-configured object types for use with User Defined Polling. It also includes an object configuration template.

```
[Type UDComponent1]
ClientData+=displayName=UD Component 1\n

[Attribute uDComponents1]
ClientData+=displayName=UDComponents1\n

[Type UDComponent2]
ClientData+=displayName=UD Component 2\n

[Attribute uDComponents2]
ClientData+=displayName=UDComponents2\n

[Type UDComponent3]
ClientData+=displayName=UD Component 3\n

[Attribute uDComponents3]
ClientData+=displayName=UDComponents3\n

[Type UDComponent4]
ClientData+=displayName=UD Component 4\n

[Attribute uDComponents4]
ClientData+=displayName=UDComponents4\n

[Type UDComponent5]
ClientData+=displayName=UD Component 5\n

[Attribute uDComponents5]
ClientData+=displayName=UDComponents5\n
```

Description

User Defined Polling allows you to define your own object types and attributes. This file defines the 20 pre-configured objects together with their attributes shipped with Entuity.

You should not amend this file because any changes to it are overwritten by subsequent Entuity upgrades. Instead create a new configuration file, for example `sw_user_defined_components_site_specific.cfg`, add your configuration to it and

include that configuration file to `sw_site_specific.cfg`. When you next run `configure` Entuity includes your new configuration.

Status

Created and maintained by Entuity. When upgrading Entuity this file is updated and any user customizations are not maintained.

systemcontrol.log

Location

entuity_home/`log`

Description

Log file recording the behavior and state of system processes. If the Process Health page indicates an error in one or more processes you may review this file when troubleshooting the cause.

Status

Created and maintained by Entuity.

system_menus.xml

This file specifies the system menus used in the Entuity web interface. The available web interface menus are a combination of menus defined in this file and in `user_menus.xml`. Menus are added to Entuity during Entuity `configure`.

`system_menus.xml` is managed by Entuity and should only be amended by Entuity.

user_menus.xml

This file specifies all user menus used in the Entuity web interface. The available web interface menus are a combination of menus defined in this file and in `system_menus.xml`. Menus are added to Entuity during Entuity `configure`.

`user_menus.xml` is user maintained. It is not overwritten during Entuity updates.

XMLDataCollector.xml

Specifies how to identify a device, apply the appropriate XML query to the device and interpret its XML reply. For example for Nexus, XML Data Collector identifies a device through its chassis identifier and system version. It can then perform the GET_MAC action with the appropriate XML configuration.

This extract includes the XML for the MAC address implementation. There are 2:

- Match sets with evaluation occurring in the order specified.
- GET_MAC actions called by the version match set. Both actions write to the same table in the XMLAPIDB.

Location

entuity_home/etc

Format

XML text file defining data collection.

```
<?xml version="1.0"?>
<XMLAPIRoot>
<version-match-sets>
  <version-match-set version-match-set-id="Nexus-1000v-001" >
     <version-match-set-test field="chassis_id" value="Nexus 1000V
Chassis" />
     <version-match-set-test field="sys_ver_str" value="4.2\(1\)SV.*"
/>
  </version-match-set>
  <version-match-set version-match-set-id="Nexus-Default">
     <version-match-set-test field="chassis_id" value=".*"    />
  </version-match-set>
</version-match-sets>
<!--   ******************** ACTIONS ******************** -->
<actions>
  <action actionName = "GET_MAC" version-match-set-id="Nexus-1000v-
001"  >
     <command>
        <show>
           <mac>
              <address-table>
                 <static/>
              </address-table>
           </mac>
        </show>
     </command>
     <rowDelimiter delimiter="ROW_mac_address"  />
     <resultTable     databaseAndTable="XMLAPIDB.MacToPort"  />
     <resultFields>
        <resultField field="disp_port"   column="Interface"   />
        <resultField field="disp_mac_addr"    column="MACAddr"  />
     </resultFields>
  </action>
```

```
  <action    actionName = "GET_MAC"    version-match-set-id="Nexus-
Default"  >
    <command>
      <show>
        <mac>
          <address-table>
            <static/>
          </address-table>
        </mac>
      </show>
    </command>
    <rowDelimiter    delimiter="ROW_mac_address"  />
    <resultTable     databaseAndTable="XMLAPIDB.MacToPort"  />
    <resultFields>
      <resultField  field="disp_port"   column="Interface"   />
      <resultField      field="disp_mac_addr"    column="MACAddr"  />
    </resultFields>
  </action>
</actions>
</XMLAPIRoot>
```

XMLDataCollector-log4j.properties

Location

entuity_home/etc/XMLDataCollector-log4j.properties

Description

Sets the level of logging applied to EYEXMLDataCollector.jar.

Status

Created and maintained by Entuity.

Appendix A Generic Trap Definitions

The following table details the OIDs and trap formats of generic standard and standard enterprise traps. Entuity identifies the OID substring and then the trap number, from which it can generate an appropriate event in Event Viewer. The first six traps are the standard generic traps.

For Cisco STP traps Entuity performs extra processing to identify the VLAN affected by the STP change, using the community string that was sent in the trap.

SNMP Trap OID (Trap No.)	Trap Format
0	Cold Start
1	Warm Start
2	Link down ifIndex=$1
3	Link up ifIndex=$1
4	Authentication Failure
5	EGP Neighbor Loss
.1.3.6.1.2.1.17(1)	Spanning tree root change from enterprise $E
.1.3.6.1.2.1.17(2)	Spanning tree topology change from enterprise $E
.1.3.6.1.4.1.9.5(1)	FDDI Link Error Rate Alarm: Trap : fddimibPORTSMTIndex=$1; fddimibPORTIndex=$2
.1.3.6.1.4.1.9.5(2)	FDDI Link Error Rate Alarm reset: Trap : fddimibPORTSMTIndex=$1; fddimibPORTIndex=$2
.1.3.6.1.4.1.9.5(3)	moduleUp trap : moduleIndex=$1
.1.3.6.1.4.1.9.5(4)	moduleDown trap : moduleIndex=$1
.1.3.6.1.4.1.9.5(5)	chassisAlarmOn trap : chassisTempAlarm=$1; chassisMinorAlarm=$2; chassisMajorAlarm=$3
.1.3.6.1.4.1.9.5(6)	chassisAlarmOff trap : chassisTempAlarm=$1; chassisMinorAlarm=$2; chassisMajorAlarm=$3
.1.3.6.1.4.1.9.5(7)	ipPermitDeniedTrap : ipPermitDeniedAddress=$1; ipPermitDeniedAccess=$2
.1.3.6.1.4.1.9.5(9)	Sysconfig changed $2 at time $1
.1.3.6.1.2.1.10.5(1)	X.25 Restart: $# args $*
.1.3.6.1.2.1.10.5(2)	X.25 Reset: $# args $*

Table 5 SNMP Trap OIDs and Formats

SNMP Trap OID (Trap No.)	Trap Format
.1.3.6.1.2.1.10.21.2(1)	dialCtlPeerCallInfo trap : callHistoryPeerId=$1; callHistoryPeerIfIndex=$2;callHistoryLogicalIfIndex=$3; ifOperStatus=$4; callHistoryPeerAddress=$5; callHistoryPeerSubAddress=$6; callHistoryDisconnectCause=$7; callHistoryConnectTime=$8;callHistoryDisconn
.1.3.6.1.2.1.10.21.2(2)	dialCtlPeerCallSetup trap : callActivePeerId=$1; callActivePeerIfIndex=$2;callActiveLogicalIfIndex=$3;ifOperStatus=$4;callActivePeerAddress=$5;callActivePeerSubAddress=$6;callActiveInfoType=$7;callActiveCallOrigin=$8
.1.3.6.1.2.1.10.32(1)	Frame Relay PVC state change: frCircuitIfIndex=$1; frCircuitDlci=$2; frCircuitState=$3
.1.3.6.1.2.1.14.16.2(1)	ospfVirtIfStateChang trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(2)	ospfNbrStateChange trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(3)	ospfVirtNbrStateChange trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(4)	ospfIfConfigError trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(5)	ospfVirtIfConfigError trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(6)	ospfIfAuthFailure trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(7)	ospfVirtIfAuthFailure trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(8)	ospfIfRxBadPacket trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(9)	ospfVirtIfRxBadPacket trap from enterprise $E
.1.3.6.1.2.1.14.16.2(10)	ospfTxRetransmit trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(11)	ospfVirtIfTxRetransmit trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(12)	ospfOriginateLsa trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(13)	ospfMaxAgeLsa trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(14)	ospfLsdbOverflow trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(15)	ospfLsdbApproachingOverflow trap received from enterprise $E
.1.3.6.1.2.1.14.16.2(16)	ospfIfStateChange trap received from enterprise $E
.1.3.6.1.2.1.15.7(1)	bgpEstablished trap received from enterprise $E with $# args: bgpPeerLastError=$1; bgpPeerState=$2
.1.3.6.1.2.1.15.7(2)	bgpBackwardTranstion trap received from enterprise $E with $# args:bgpPeerLastError=$1; bgpPeerState=$2
.1.3.6.1.2.1.16(1)	RMON Rising Alarm from enterprise $E with args:alarmIndex=$1; alarmVariable=$2; alarmSampleType=$3; alarmValue=$4; alarmRisingThreshold=$5

Table 5 SNMP Trap OIDs and Formats

SNMP Trap OID (Trap No.)	Trap Format
.1.3.6.1.2.1.16(2)	RMON Falling Alarm from enterprise $E with $# args: alarmIndex=$1; alarmVariable=$2; alarmSampleType=$3; alarmValue=$4; alarmFallingThreshold=$5
.1.3.6.1.2.1.16(3)	RMON Packet Match trap: Matched channel index #$1 ($3); match count at $2
.1.3.6.1.2.1.22(1)	Repeater health status change from enterprise $E with args: rptrOperStatus=$1
.1.3.6.1.2.1.22(2)	rptrGroupChange trap received from enterprise $E with $# args: rptrGroupIndex=$1
.1.3.6.1.2.1.22(3)	rptrResetEvent trap received from enterprise $E with $# args:rptrOperStatus=$1
.1.3.6.1.2.1.34.1.1.5(2)	snaLuSessnBindFailure trap : snaLuSessnLocalApplName=$1; snaLuSessnRemoteLuName=$2; snaLuSessnOperState=$3; snaLuSessnSenseData=$4
.1.3.6.1.2.1.34.1.1.5(1)	snaLuStateChangeTrap : snaLuOperName=$1; snaLuOperSnaName=$2; snaLuOperState=$3
.1.3.6.1.2.1.34.1.1.10(2)	snaNodeActFailTrap : snaNodeOperName=$1; snaNodeOperState=$2
.1.3.6.1.2.1.34.1.1.10(1)	snaNodeStateChange trap : snaNodeOperName=$1; snaNodeOperState=$2
.1.3.6.1.2.1.41.1.3(1)	sdlcPortStatusChange trap received from enterprise $E with $# args:ifIndex=$1; ifAdminStatus=$2; ifOperStatus=$3; sdlcPortOperLastFailTime=$4; sdlcPortOperLastFailCause=$5
.1.3.6.1.2.1.41.1.3(2)	sdlcLSStatusChange trap received from enterprise $E with $# args:ifIndex=$1; sdlcLSAddress=$2; sdlcLSOperState=$3; sdlcLSAdminState=$4; sdlcLSOperLastFailTime=$5; sdlcLSOperLastFailCause=$6; sdlcLSOperLastFailFRMRInfo=$7; sdlcLSOperLastFailCtrlIn=$8; sdlc
.1.3.6.1.2.1.46.1(1)	dlswTrapCntlTConnPartnerReject trap received from enterprise $E with $# args: dlswTConnOperTDomain=$1;dlswTConnOperRemoteTAddr=$2
.1.3.6.1.2.1.46.1(2)	dlswTrapTConnProtViolation trap received from enterprise $E with $# args: dlswTConnOperTDomain=$1;dlswTConnOperRemoteTAddr=$2

Table 5 SNMP Trap OIDs and Formats

SNMP Trap OID (Trap No.)	Trap Format
.1.3.6.1.2.1.46.1(3)	dlswTrapTConnUp trap received from enterprise $E with $# args: dlswTConnOperTDomain=$1;dlswTConnOperRemoteTAddr=$2
.1.3.6.1.2.1.46.1(4)	dlswTrapTConnDown trap received from enterprise $E with $# args: dlswTConnOperTDomain=$1; dlswTConnOperRemoteTAddr=$2
.1.3.6.1.2.1.46.1(5)	dlswTrapCircuitUp trap received from enterprise $E with $# args: dlswCircuitS1Mac=$1;dlswCircuitS1Sap=$2;dlswCircuitS2Mac=$3;dlswCircuitS2Sap=$4
.1.3.6.1.2.1.46.1(6)	dlswTrapCircuitDown trap received from enterprise $E with $# args: dlswCircuitS1Mac=$1;dlswCircuitS1Sap=$2;dlswCircuitS2Mac=$3;dlswCircuitS2Sap=$4
.1.3.6.1.3.71.2(1)	newFlow trap : rsvpFlowIndex=$1; ifIndex=$2
.1.3.6.1.3.71.2(2)	lostFlow trap : rsvpFlowIndex=$1; ifIndex=$2
.1.3.6.1.4.1(*)	Received event $o (enterprise:$e generic:$G specific:$S), no format in trapd.conf. $# args: $*
.1.3.6.1.4.1.2.6.66.1.2.1.2(1)	ibm8272TsTempThreshold trap received from enterprise $E with $# args: sysName=$1; sysLocation=$2; ibm8272TsSysTemperature=$3
.1.3.6.1.4.1.2.6.66.1.2.1.2(2)	ibm8272TsPwrSupChange trap received from enterprise $E with $# args:sysName=$1; sysLocation=$2; ibm8272TsSysPwrStatus=$3
.1.3.6.1.4.1.2.6.66.1.2.1.2(3)	ibm8272TsFanChange trap received from enterprise $E with $# args:sysName=$1; sysLocation=$2; ibm8272TsSysFanStatus=$3
.1.3.6.1.4.1.2.6.66.1.2.1.2(4)	ibm8272TsVoltageChange trap received from enterprise $E with $# args:sysName=$1; sysLocation=$2; ibm8272TsSysVoltageStatus=$3
.1.3.6.1.4.1.2.6.66.1.2.2(1)	ibm8272TsPortCfgLossTrap received from enterprise $E with $# args:ibm8272TsPortIndex=$1;ibm8272TsPortCfgLoss=$2
.1.3.6.1.4.1.2.6.66.1.2.2(2)	ibm8272TsBeaconStart trap received from enterprise $E with $# args: ibm8272TsPortIndex=$1
.1.3.6.1.4.1.2.6.66.1.2.2(3)	ibm8272TsBeaconEnd trap received from enterprise $E with $# args: ibm8272TsPortIndex=$1
.1.3.6.1.4.1.2.6.66.1.2.2(4)	ibm8272TsMaxFrameSizeExceeded trap received from enterprise $E with $# args: ibm8272TsPortIndex=$1

Table 5 SNMP Trap OIDs and Formats

SNMP Trap OID (Trap No.)	Trap Format
.1.3.6.1.4.1.2.6.66.1.2.2(5)	ibm8272TsPortSwitchModeChangeTrap received from enterprise $E with $# args: ibm8272TsPortIndex=$1; ibm8272TsPortSwitchMode=$2
.1.3.6.1.4.1.2.6.66.1.2.3(1)	ibm8272TsDmnNewRoot trap received from enterprise $E with $# args: ibm8272TsDmnIndex=$1
.1.3.6.1.4.1.2.6.66.1.2.3(2)	ibm8272TsDmnTopologyChange trap received from enterprise $E with $# args: ibm8272TsDmnIndex=$1
.1.3.6.1.4.1.9(*)	Cisco default trap: generic: $G specific: $S args($#): $*
.1.3.6.1.6.3.1.1.5.1.1.3.6.1.4.1(9)	Cold start: Trap : sysUpTime=$1; whyReload=$2
.1.3.6.1.6.3.1.1.5.2.1.3.6.1.4.1(9)	Cisco Agent Up with No Changes (warmStart Trap)
.1.3.6.1.6.3.1.1.5.3.1.3.6.1.4.1(9)	linkDown trap : ifIndex=$1; ifDescr=$2; ifType=$3; locIfReason=$4
.1.3.6.1.6.3.1.1.5.4.1.3.6.1.4.1(9)	linkUp trap : ifIndex=$1; ifDescr=$2; ifType=$3; locIfReason=$4
.1.3.6.1.6.3.1.1.5.5.1.3.6.1.4.1(9)	Authentication Failure trap : authAddr=$1
.1.3.6.1.6.3.1.1.5.6.1.3.6.1.4.1(9)	Cisco EGP Neighbor Down (egpNeighborLoss Trap) egpNeighAddr: $1
.1.3.6.1.4.1.9(0)	Cisco_reload trap : sysUpTime=$1; whyReload=$2
.1.3.6.1.4.1.9(1)	TCP connection terminated. Trap : tslineSesType=$1; tcpConnState=$2; loctcpConnElapsed=$3; loctcpConnInBytes=$4; loctcpConnOutBytes=$5; tsLineUser=$6
.1.3.6.1.4.1.9.1.111.1.2.1.2(1)	TsTempThreshold trap :
.1.3.6.1.4.1.9.1.111.1.2.2(1)	TsPortCfgLossTrap trap :
.1.3.6.1.4.1.9.1.111.1.2.2(2)	TsBeaconStart trap :
.1.3.6.1.4.1.9.1.111.1.2.2(3)	TsBeaconEnd trap :
.1.3.6.1.4.1.9.1.111.1.2.2(4)	TSMaxFrameSizeExceed trap :
.1.3.6.1.4.1.9.1.111.1.2.2(5)	TsPortSwitchModeChge trap :
.1.3.6.1.4.1.9.1.111.1.2.3(1)	2600TsDmnNewRoot trap :
.1.3.6.1.4.1.9.1.111.1.2.3(2)	2600TsDmnTopoChge trap :
.1.3.6.1.6.3.1.1.5.1.1.3.6.1.4.1.9.1(147)	CiscoPro316T on a power-on reset
.1.3.6.1.6.3.1.1.5.2.1.3.6.1.4.1.9.1(147)	CiscoPro316T is reset (warmStart Trap)
.1.3.6.1.6.3.1.1.5.5.1.3.6.1.4.1.9.1(147)	CiscoPro316T Incorrect Community Name (authenticationFailure Trap)

Table 5 SNMP Trap OIDs and Formats

SNMP Trap OID (Trap No.)	Trap Format
.1.3.6.1.6.3.1.1.5.1.1.3.6.1.4.1.9.1(148)	CiscoPro316C on a power-on reset
.1.3.6.1.6.3.1.1.5.2.1.3.6.1.4.1.9.1(148)	CiscoPro316C is reset (warmStart Trap)
.1.3.6.1.6.3.1.1.5.5.1.3.6.1.4.1.9.1(148)	CiscoPro316C Incorrect Community Name (authenticationFailure Trap)
.1.3.6.1.6.3.1.1.5.1.1.3.6.1.4.1.9.1(149)	CiscoPro3116 on a power-on reset
.1.3.6.1.6.3.1.1.5.2.1.3.6.1.4.1.9.1(149)	CiscoPro3116 is reset (warmStart Trap)
.1.3.6.1.6.3.1.1.5.5.1.3.6.1.4.1.9.1(149)	CiscoPro3116 Incorrect Community Name (authenticationFailure Trap)
.1.3.6.1.6.3.1.1.5.1.1.3.6.1.4.1.9.1(150)	Catalyst116T on a power-on reset
.1.3.6.1.6.3.1.1.5.2.1.3.6.1.4.1.9.1(150)	Catalyst116T is reset (warmStart Trap)
.1.3.6.1.6.3.1.1.5.5.1.3.6.1.4.1.9.1(150)	Catalyst116T Incorrect Community Name (authenticationFailure Trap)
.1.3.6.1.6.3.1.1.5.1.1.3.6.1.4.1.9.1(151)	Catalyst116C on a power-on reset
.1.3.6.1.6.3.1.1.5.2.1.3.6.1.4.1.9.1(151)	Catalyst116C is reset (warmStart Trap)
.1.3.6.1.6.3.1.1.5.5.1.3.6.1.4.1.9.1(151)	Catalyst116C Incorrect Community Name (authenticationFailure Trap)
.1.3.6.1.6.3.1.1.5.1.1.3.6.1.4.1.9.1(152)	Catalyst1116 on a power-on reset
.1.3.6.1.6.3.1.1.5.2.1.3.6.1.4.1.9.1(152)	Catalyst1116 is reset (warmStart Trap)
.1.3.6.1.6.3.1.1.5.5.1.3.6.1.4.1.9.1(152)	Catalyst1116 Incorrect Community Name (authenticationFailure Trap)
.1.3.6.1.4.1.9.2.11.1(0)	Possible logon intrusion
.1.3.6.1.4.1.9.2.11.1(1)	Diagnostic failure
.1.3.6.1.4.1.9.2.11.1(2)	Redundant power supply failed
.1.3.6.1.4.1.9.2.11.1(3)	Ip address change
.1.3.6.1.6.3.1.1.5.1.1.3.6.1.4.1.9(5)	Cold Start trap : sysUpTime=$1; whyReload=$2
.1.3.6.1.6.3.1.1.5.2.1.3.6.1.4.1.9(5)	Cisco Agent Up with No Changes (warmStart Trap)
.1.3.6.1.6.3.1.1.5.3.1.3.6.1.4.1.9(5)	linkDown trap received from enterprise $E with ifIndex=$1

Table 5 SNMP Trap OIDs and Formats

SNMP Trap OID (Trap No.)	Trap Format
.1.3.6.1.6.3.1.1.5.4.1.3.6.1.4.1.9 (5)	linkUp trap received from enterprise $E with ifIndex=$1
.1.3.6.1.6.3.1.1.5.5.1.3.6.1.4.1.9 (5)	Authentication Failure trap : authAddr=$1
.1.3.6.1.6.3.1.1.5.5.1.3.6.1.4.1.9 .5(7)	Authentication Failure - Received event $E. $# args: $*
.1.3.6.1.4.1.9.5.11.2(1)	LS1010ChassisFail trap : ciscoLS1010ChassisPs0Status=$1; ciscoLS1010ChassisPs1Status=$2; ciscoLS1010ChassisFanStatus=$3; ciscoLS1010Chassis12VoltStatus=$4; ciscoLS1010ChassisTempStatus=$5
.1.3.6.1.4.1.9.5.11.2(2)	LS1010ChassisChg trap :
.1.3.6.1.4.1.9.5.14.1.1(1)	ciscoEsStackCfgChang trap : sysName=$1; sysLocation=$2; ciscoEsNumSwitches=$3
.1.3.6.1.4.1.9.5.14.1.1(2)	ciscoEsStackProStack trap : sysName=$1; sysLocation=$2; ciscoEsProStackMatrixStatus=$3
.1.3.6.1.4.1.9.5.14.2(1)	ciscoEsStackTempChange trap : sysName=$1; sysLocation=$2; ciscoEsStackSwitchTemperature=$3
.1.3.6.1.4.1.9.5.14.2(3)	Temperature state changed
.1.3.6.1.4.1.9.5.14.4(1)	ciscoEsPortStrNFwdEn trap : sysName=$1; sysLocation=$2; ciscoEsPortActiveMode=$3
.1.3.6.1.4.1.9.5.14.4(4)	Switching mode changed
.1.3.6.1.4.1.9.5.14.6(1)	ciscoEsEtherChannelFail trap : sysName=$1; sysLocation=$2; ciscoEsECPorts=$3
.1.3.6.1.4.1.9.5.14.6(7)	EtherChannel Failure
.1.3.6.1.4.1.9.5.14.8(1)	ciscoEsVLANNewRoot trap : ciscoEsVLANInfoVLANNumber=$1
.1.3.6.1.4.1.9.5.14.8(2)	ciscoEsVLANTopologyChange trap : ciscoEsVLANInfoVLANNumber=$1
.1.3.6.1.4.1.9.5.14.8(5)	Spanning Tree new root
.1.3.6.1.4.1.9.5.14.8(6)	Spanning Tree topology change
.1.3.6.1.4.1.9.9.10.1.3(1)	CopyCompletionTrap : ciscoFlashCopyStatus=$1
.1.3.6.1.4.1.9.9.10.1.3(2)	PartitioningComplete trap : ciscoFlashPartitioningStatus=$1
.1.3.6.1.4.1.9.9.10.1.3(3)	ciscoFlashMiscOpCompletionTrap : ciscoFlashMiscOpStatus=$1
.1.3.6.1.4.1.9.9.10.1.3(4)	ciscoFlashDeviceChangeTrap : ciscoFlashDeviceIndex=$1
.1.3.6.1.4.1.9.9.13.3(1)	Cisco Shutdown Notification from enterprise $E with $# args $*

Table 5 SNMP Trap OIDs and Formats

SNMP Trap OID (Trap No.)	Trap Format
.1.3.6.1.4.1.9.9.13.3(2)	ciscoEnvMonVoltageNotification trap : ciscoEnvMonVoltageStatusDescr=$1; ciscoEnvMonVoltageStatusValue=$2; ciscoEnvMonVoltageState=$3
.1.3.6.1.4.1.9.9.13.3(3)	ciscoEnvMonTemperatureNotification : ciscoEnvMonTemperatureStatusDescr=$1; ciscoEnvMonTemperatureStatusValue=$2; ciscoEnvMonTemperatureState=$3
.1.3.6.1.4.1.9.9.13.3(4)	ciscoEnvMonFanNotification : ciscoEnvMonFanStatusDescr=$1; ciscoEnvMonFanState=$2
.1.3.6.1.4.1.9.9.13.3(5)	ciscoEnvMonRedundantSupplyNotification : ciscoEnvMonSupplyStatusDescr=$1; ciscoEnvMonSupplyState=$2
.1.3.6.1.4.1.9.9.16.2(1)	ciscoPingCompleted trap : ciscoPingCompleted=$1; ciscoPingSentPackets=$2; ciscoPingReceivedPackets=$3
.1.3.6.1.4.1.9.9.18.2(1)	ciuIfLoopStatusNotification : ciuIfLoopStatus=$1
.1.3.6.1.4.1.9.9.20.1.5(1)	cipCardLinkFailure trap : cipCardDtrBrdIndex=$1; cipCardDtrBrdStatus=$2; cipCardDtrBrdSignal=$3; linkIncidentTrapCause=$4; implicitIncidents=$5; codeViolationErrors=$6; linkFailureSignalOrSyncLoss=$7; linkFailureNOSs=$8; linkFailureSequenceTimeouts=$9; li
.1.3.6.1.4.1.9.9.20.1.5(2)	cipCardDtrBrdLinkFailure trap : cipCardDtrBrdStatus=$1; cipCardDtrBrdSignal=$2; linkIncidentTrapCause=$3
.1.3.6.1.4.1.9.9.22.3(1)	IllegalSrcAddrTrap : ciscoRptrPortLastIllegalSrcAddr=$1
.1.3.6.1.4.1.9.9.24.1.4.4(1)	newdspuPuStateChange trap : dspuPuOperName=$1; dspuPuOperState=$2
.1.3.6.1.4.1.9.9.24.1.4.4(2)	newdspuPuActivatFail trap : dspuPuOperName=$1; dspuPuOperState=$2; dspuPuStatsLastActivationFailureReason=$3
.1.3.6.1.4.1.9.9.24.1.5.3(1)	newdspuLuStateChange trap : dspuPuOperName=$1; dspuLuOperState=$2
.1.3.6.1.4.1.9.9.24.1.5.3(2)	dspuLuActivationFail trap : dspuPuOperName=$1; dspuLuOperState=$2; dspuLuOperLastActivationFailureReason=$3
.1.3.6.1.4.1.9.9.24.1.6.2(1)	dspuSapStateChange trap : dspuSapDlcType=$1; dspuSapDlcUnit=$2; dspuSapDlcPort=$3; dspuSapAddress=$4; dspuSapOperState=$5
.1.3.6.1.4.1.9.9.26.2(1)	demandNbrCallInfo trap received from enterprise $E with $# args:\n$*

Table 5 SNMP Trap OIDs and Formats

SNMP Trap OID (Trap No.)	Trap Format
.1.3.6.1.4.1.9.9.26.2(2)	demandNbrCallDetails trap : demandNbrLogIf=$1; demandNbrName=$2; demandNbrAddress=$3; demandNbrLastDuration=$4; demandNbrClearReason=$5; demandNbrClearCode=$6; demandNbrCallOrigin=$7
.1.3.6.1.4.1.9.9.28.2(1)	SdllcPeerStateChange trap : convSdllcAddrState=$1
.1.3.6.1.4.1.9.9.29.2(1)	PeerStateChangeNotification : rsrbRemotePeerState=$1
.1.3.6.1.4.1.9.9.30.2(1)	stunPeerStateChange trap : stunRoutePeerState=$1
.1.3.6.1.4.1.9.9.33.2(1)	snaOpenDuplctSapFail trap : cipCardCsnaSlot=$1; cipCardCsnaPort=$2; cipCardCsnaConnPath=$3; cipCardCsnaConnDevice=$4
.1.3.6.1.4.1.9.9.33.2(2)	Llc2ConnctLimitXceed trap : cipCardAdminMaxLlc2Sessions=$1; cipCardOperMaxLlc2Sessions=$2; cipCardStatsHiWaterLlc2Sessions=$3; cipCardStatsLlc2SessionAllocationErrs=$4
.1.3.6.1.4.1.9.9.35.2(1)	bstunPeerStateChg trap : bstunRoutePeerState=$1
.1.3.6.1.4.1.9.9.41.2(1)	clogMessageGenerated trap : clogHistFacility=$1; clogHistSeverity=$2; clogHistMsgName=$3; clogHistMsgText=$4; clogHistTimestamp=$5
.1.3.6.1.4.1.9.9.42.2(1)	rttMonConnectionChange trap : rttMonCtrlAdminTag=$1; rttMonHistoryCollectionAddress=$2; rttMonCtrlOperConnectionLostOccurred=$3
.1.3.6.1.4.1.9.9.42.2(2)	rttMonTimeoutNotification : rttMonCtrlAdminTag=$1; rttMonHistoryCollectionAddress=$2; rttMonCtrlOperTimeoutOccurred=$3
.1.3.6.1.4.1.9.9.42.2(3)	rttMonThresholdNotification : rttMonCtrlAdminTag=$1; rttMonHistoryCollectionAddress=$2; rttMonCtrlOperOverThresholdOccurred=$3
.1.3.6.1.4.1.9.9.43.2(1)	ciscoConfigManEvent : ccmHistoryEventCommandSource=$1; ccmHistoryEventConfigSource=$2; ccmHistoryEventConfigDestination=$3
.1.3.6.1.4.1.9.9.44.3(1)	T1LoopStatusNotification : ciscoICsuDsuT1LoopStatus=$1
.1.3.6.1.4.1.9.9.44.3(2)	Sw56kLoopStatusNotification : ciscoICsuDsuSw56kLoopStatus=$1
.1.3.6.1.4.1.9.9.46.2.0(1)	vtpConfigRevNumberError : managementDomainConfigRevNumber=$1
.1.3.6.1.4.1.9.9.46.2.0(2)	vtpConfigDigestError : managementDomainConfigRevNumber=$1

Table 5 SNMP Trap OIDs and Formats

SNMP Trap OID (Trap No.)	Trap Format
.1.3.6.1.4.1.9.9.46.2.0(3)	vtpServerDisabled trap : managementDomainConfigRevNumber=$1; vtpMaxVlanStorage=$2
.1.3.6.1.4.1.9.9.46.2.0(4)	vtpMtuTooBig trap : vlanTrunkPortManagementDomain=$1; vtpVlanState=$2
.1.3.6.1.4.1.9.9.46.2.0(5)	vtpVlanRingNumberConfigConflict : vtpVlanIndex=$1; vtpVlanRingNumber=$2; ifIndex=$3; vtpVlanPortLocalSegment=$4
.1.3.6.1.4.1.9.9.46.2.0(6)	vtpVersionOneDeviceDetected trap : vlanTrunkPortManagementDomain=$1
.1.3.6.1.4.1.9.9.52.2(1)	cieTestCompletion trap : cieTestConnSessionStatus=$1; cieTestConnProtectedAddr=$2; cieTestConnUnprotectedAddr=$3
.1.3.6.1.4.1.9.9.61.2(1)	caemTemperatureNotification :ciscoEnvMonTemperatureStatusDescr=$1; ciscoEnvMonTemperatureState=$2
.1.3.6.1.4.1.9.9.63(2)	cvdcPoorQoVNotification : cvVoIPCallHistoryConnectionId=$1; cvVoIPCallHistoryIcpif=$2
.1.3.6.1.4.1.9.9.68.2(1)	vmVmpsChange trap : vmVmpsIpAddress=$1
.1.3.6.1.4.1.9.9.74.2(1)	cdeTrapTConnUpDown trap received from enterprise $E with $# arguements: dlswTConnOperState=$1
.1.3.6.1.4.1.9.9.74.2(2)	cdeTrapCircuitUpDown trap : dlswCircuitState =$1
.1.3.6.1.4.1.9.10.8.2(1)	llcCcStatusChange trap : llcCcOperState=$1; llcCcOperLastFailTime=$2; llcCcOperLastFailCause=$3; llcCcOperLastFailFRMRInfo=$4
.1.3.6.1.4.1.9.10.9.1.7(1)	TConnPartnerReject trap : ciscoDlswTConnOperTDomain=$1; ciscoDlswTConnOperRemoteTAddr=$2
.1.3.6.1.4.1.9.10.9.1.7(2)	TConnProtViolation trap : ciscoDlswTConnOperTDomain=$1; ciscoDlswTConnOperRemoteTAddr=$2
.1.3.6.1.4.1.9.10.9.1.7(3)	TConnUp trap : ciscoDlswTConnOperTDomain=$1; ciscoDlswTConnOperRemoteTAddr=$2
.1.3.6.1.4.1.9.10.9.1.7(4)	TconnDown trap : ciscoDlswTConnOperTDomain=$1; ciscoDlswTConnOperRemoteTAddr=$2
.1.3.6.1.4.1.9.10.9.1.7(5)	DlswTrapCircuitUp trap : ciscoDlswCircuitS1Mac=$1; ciscoDlswCircuitS1Sap=$2; ciscoDlswCircuitS2Mac=$3; ciscoDlswCircuitS2Sap=$4
.1.3.6.1.4.1.9.10.9.1.7(6)	DlswtrapCircuitDown trap : ciscoDlswCircuitS1Mac=$1; ciscoDlswCircuitS1Sap=$2; ciscoDlswCircuitS2Mac=$3; ciscoDlswCircuitS2Sap=$4

Table 5 SNMP Trap OIDs and Formats

SNMP Trap OID (Trap No.)	Trap Format
.1.3.6.1.4.1.9.10.15.2(1)	oamLoopbackPingCompleted : oamLoopbackPingCompleted=$1
.1.3.6.1.4.1.9.10.17.3(1)	acctngFileNearlyFull trap : acctngFileName=$1; acctngFileMaximumSize=$2; acctngControlTrapThreshold=$3
.1.3.6.1.4.1.9.10.17.3(2)	acctngFileFull trap : acctngFileName=$1; acctngFileMaximumSize=$2
.1.3.6.1.4.1.9.14(1)	$1: $2
.1.3.6.1.4.1.9.14(2)	$1: $2
.1.3.6.1.4.1.23.2.5.5(1)	ipxTrapCircuitDown trap : ipxCircSysInstance=$1; ipxCircIndex=$2
.1.3.6.1.4.1.23.2.5.5(2)	ipxTrapCircuitUp trap : ipxCircSysInstance=$1; ipxCircIndex=$2
.1.3.6.1.4.1.141.1.1.3(1)	RMON: Rising (high) threshold exceeded
.1.3.6.1.4.1.141.1.1.3(2)	RMON: Falling (low) threshold crossed
.1.3.6.1.4.1.197.3.1.1(2)	kalEps3StackProStackMatrixChange trap : sysName=$1; sysLocation=$2; kalEps3ProStackMatrixStatus=$3
.1.3.6.1.4.1.197.3.2(1)	kalEps3StackTempChange trap : sysName=$1; sysLocation=$2; kalEps3StackSwitchTemperature=$3
.1.3.6.1.4.1.197.3.4(1)	kalEps3PortStrNFwdEntry trap : sysName=$1; sysLocation=$2; kalEps3PortActiveMode=$3
.1.3.6.1.4.1.197.3.5(1)	kalEps3DmnNewRoot trap : kalEps3DmnInfoDmnNumber=$1
.1.3.6.1.4.1.197.3.5(2)	kalEps3DmnTopologyChange trap : kalEps3DmnInfoDmnNumber=$1
.1.3.6.1.4.1.197.3.6(2)	kalEps3EtherChannelFailed trap : sysName=$1
.1.3.6.1.4.1.353(1)	atmfVpcChange : atmfVpcPortIndex=$1; atmfVpcVpi=$2; atmfVpcOperStatus=$3
.1.3.6.1.4.1.353(2)	atmfVccChange : atmfVccPortIndex=$1; atmfVccVci=$2; atmfVccVpi=$3; atmfVccOperStatus=$4
.1.3.6.1.4.1.437.1.1.1.1(0)	Possible logon intrusion.
.1.3.6.1.4.1.437.1.1.1.1(1)	Diagnostic failure
.1.3.6.1.4.1.437.1.1.3(0)	Possible logon intrusion. Trap : sysName=$1
.1.3.6.1.4.1.437.1.1.3(1)	Diagnostic failure. Trap : sysName=$1
.1.3.6.1.4.1.437.1.1.3(3)	addressViolation trap : ifIndex=$1
.1.3.6.1.4.1.437.1.1.3(4)	Broadcast threshold exceeded. Trap : ifIndex=$1
.1.3.6.1.4.1.437.1.1.3(5)	Redundant power supply failed. Trap : sysName=$1
.1.3.6.1.4.1.494.4(1)	fanPSSpeedFailed trap : ringswitchBasePSFanSpeed=$1

Table 5 SNMP Trap OIDs and Formats

SNMP Trap OID (Trap No.)	Trap Format
.1.3.6.1.4.1.494.4(2)	fanExtSpeedFailed trap : ringswitchBaseExtFanSpeed=$1
.1.3.6.1.4.1.494.4(3)	portFailed trap : ringswitchPortAdapterStatus=$1
.1.3.6.1.4.1.494.4(4)	brTestFailed trap : ringswitchPortTestError=$1
.1.3.6.1.6.3.1.1.5(1)	Agent Up with Possible Changes (coldStart Trap) enterprise:$E ($e) args($#):$*
.1.3.6.1.6.3.1.1.5(2)	Agent Up with No Changes (warmStart Trap) enterprise:$E ($e) args($#):$*
.1.3.6.1.6.3.1.1.5(3)	Agent Interface Down (linkDown Trap) enterprise:$E ($e) on interface $1
.1.3.6.1.6.3.1.1.5(4)	Agent Interface Up (linkUp Trap) enterprise:$E ($e) on interface $1
.1.3.6.1.6.3.1.1.5(5)	Incorrect Community Name (authenticationFailure Trap) enterprise:$E ($e) args($#):$*
.1.3.6.1.6.3.1.1.5(6)	EGP Neighbor Down (egpNeighborLoss Trap) enterprise:$E ($e) neighbor $1

Table 5 SNMP Trap OIDs and Formats

Appendix B Entuity Internal Identifiers

Entuity uses a series of codes to identify the types of objects it manages. These internal codes are sometimes useful when troubleshooting. This section details two types of codes and how they are used in a third, *eosObjectID*, to uniquely identify a managed object.

Entuity Object Types

Entuity identifies different managed object types by assigning each their own unique identifier. This number is only used within Entuity, but occasionally you may find it useful to use some of them. For example, when decoding an *eosObjectID*.

Object Type	Object Identifier
Port	1
Device	4
VLAN	8
VLAN Inherit	16
Marker	32
Device Inheritance	64
Domain	128
Reference	256
Server	512
Application	1024
IP Address	2048
System	4096
StormWorks	2147483648

Table 6 Entuity Object Types

Entuity Device Types

Entuity identifies different device types by assigning each their own unique identifier. This number is only used within Entuity, but occasionally you may need to use it. For example the discovery ping technology used for Entuity for its maps can be configured through `entuity.cfg` to exclude certain device types. These device types are specified through their device identifiers.

Device Type	Entuity Identifier
Aruba Mobility Controller	1102

Table 7 Entuity Device Types

Device Type	Entuity Identifier
Autonomous WAPs	1046
BladeCenters	1001
CUCMs	1002
Firewalls	1049
Hubs	10
Load Balancer	1077
Managed Hosts	1059
Matrix Switch	1124
Multiplexer	1200
Non-SNMP Device	1062
PoE Midspan Injector	1070
Root	11
Routers	168
SSL Proxy	1079
Switches	148
System	0
Unclassified	1088
Unclassified (Full)	1069
Uninterruptible Power Supply	1104
VM Platform	1144
VPNs	1058
Wide Area Application Service	1128
Wireless Controllers	1073 / 1102

Table 7 Entuity Device Types

eosObjectID

eosObjectID is an internal Entuity identifier that uniquely identifies Entuity managed objects. *eosObjectID* is a bit mask with the format:

```
objectType.objectID.portID.StormWorksID
```

where:

- *objectType* is the internal Entuity object type, for example 1 for port, 4 for device.
- *objectID* is the unique object, e.g. device identifier.
- *portID* is the unique port identifier, when used in the context of *objectID*.
- *StormWorksID* is the unique StormWorks identifier. It is set to 0 when the object does not have a StormWorks number.

For example these are valid *eosObjectID* identifiers:

- 1.131.17.36477, represents port 17 on device 131. It also has a StormWorks identifier, 36477.
- 4.131.1.0, represents device 131, and does not include a StormWorks identifier, 0.

eosObjectID is normally only used by internal Entuity processes, however they can be viewed:

- ForkEvent forwards *objectID* and *objectType* as part of *eosObjectID*.
- Entuity Remedy AR System integration module uses ForkEvent to forward *objectID*, *objectType* and *StormWorksID* as part of *eosObjectID*.
- Flex Reports allow you to report on StormWorks identifiers when you select *Show Hidden Data*.

Appendix C Port Interface Types

Entuity distinguishes between WAN and LAN ports by comparing the port's interface type against a list of types held in the Entuity database.

Entuity has determined the most probable use for each interface type and marked it as either a LAN or WAN port interface. Entuity then uses this association to ensure it is reporting on the correct ports when running Routing Summary report.

Entuity identifies a leased line by the interface type, by default either IANAifType 22 (propPointToPointSerial) or 23 (PPP). Entuity recognizes and discounts FrameRelay, ISDN and ATM ports.

Unknown Port Interfaces

Port interfaces that do not belong to either of the WAN or LAN groups listed in the Entuity database are listed as other.

IANAifTypeDesc_32_1	IANAifType	Group Name
unknown	1	other

Table 8 Unknown Port Interfaces

LAN Port Interfaces

The interface types defined through the Module IANAifType_MIB are in the Entuity database marked as either WAN or LAN ports. *Table LAN Port Interfaces* lists the LAN ports by interface description.

IANAifTypeDesc_32_1	IANAifType	Group Name
aflane8023	59	LAN
aflane8025	60	LAN
channel	70	LAN
escon	73	LAN
ethernet3Mbit	26	LAN
ethernetCsmacd	6	LAN
fastEther	62	LAN
fastEtherFX	69	LAN
fddi	15	LAN
fibreChannel	56	LAN

Table 9 LAN Port Interfaces

IANAifTypeDesc_32_1	IANAifType	Group Name
gigabitEthernet	117	LAN
hyperchannel	14	LAN
ibm370parChan	72	LAN
ieee80212	55	LAN
ieee8023adLag	161	LAN
iso88022llc	41	LAN
iso88023Csmacd	7	LAN
iso88024TokenBus	8	LAN
iso88025CRFPInt	98	LAN
iso88025Dtr	86	LAN
iso88025Fiber	115	LAN
iso88025TokenRing	9	LAN
iso88026Man	10	LAN
mpc	113	LAN
opticalChannel	195	LAN
opticalTransport	196	LAN
proteon10Mbit1	12	LAN
proteon80Mbit	13	LAN
starLan	11	LAN

Table 9 LAN Port Interfaces

WAN Port Interfaces

The interface types defined through the Module IANAifType_MIB are in the Entuity database marked as either WAN or LAN ports. *Table WAN Ports* lists the WAN ports by interface description.

IANAifTypeDesc_32_1	IANAifType	Group Name
a12MppSwitch	130	WAN
aal2	187	WAN
aal5	49	WAN
adsl	94	WAN
arap	88	WAN
arcnet	35	WAN
arcnetPlus	36	WAN
async	84	WAN

Table 10 WAN Ports

IANAifTypeDesc_32_1	IANAifType	Group Name
atm	37	WAN
atmDxi	105	WAN
atmFuni	106	WAN
atmIma	107	WAN
atmLogical	80	WAN
atmRadio	189	WAN
atmSubInterface	134	WAN
atmVciEndPt	194	WAN
atmVirtual	149	WAN
basicISDN	20	WAN
bgppolicyaccounting	162	WAN
bsc	83	WAN
cctEmul	61	WAN
ces	133	WAN
cnr	85	WAN
coffee	132	WAN
compositeLink	155	WAN
dcn	141	WAN
ddnX25	4	WAN
digitalPowerline	138	WAN
digitalWrapperOverheadChannel	186	WAN
dlsw	74	WAN
docsCableDownstream	128	WAN
docsCableMaclayer	127	WAN
docsCableUpstream	129	WAN
docsOfdmaUpstream	278	WAN
docsOfdmDownstream	277	WAN
ds0	81	WAN
ds0Bundle	82	WAN
ds1	18	WAN
ds1FDL	170	WAN
ds3	30	WAN
dtm	140	WAN
dvbAsiIn	172	WAN
dvbAsiOut	173	WAN

Table 10 WAN Ports

IANAifTypeDesc_32_1	IANAifType	Group Name
dvbRccDownstream	147	WAN
dvbRccMacLayer	146	WAN
dvbRccUpstream	148	WAN
e1	19	WAN
eon	25	WAN
eplrs	87	WAN
fast	125	WAN
frameRelay	32	WAN
frameRelayInterconnect	58	WAN
frameRelayMPI	92	WAN
frameRelayService	44	WAN
frDlciEndPt	193	WAN
frf16MfrBundle	163	WAN
frForward	158	WAN
g703at2mb	67	WAN
g703at64k	66	WAN
gfast	279	WAN
gr303IDT	178	WAN
gr303RDT	177	WAN
h323Gatekeeper	164	WAN
h323Proxy	165	WAN
hdh1822	3	WAN
hdlc	118	WAN
hdsl2	168	WAN
hiperlan2	183	WAN
hippi	47	WAN
hippiInterface	57	WAN
hostPad	90	WAN
hssi	46	WAN
idsl	154	WAN
ieee1394	144	WAN
ieee80211	71	WAN
if-gsn	145	WAN
imt	190	WAN
interleave	124	WAN

Table 10 WAN Ports

IANAifTypeDesc_32_1	IANAifType	Group Name
ip	126	WAN
ipForward	142	WAN
ipOverAtm	114	WAN
ipOverCdlc	109	WAN
ipOverClaw	110	WAN
ipSwitch	78	WAN
isdn	63	WAN
isdns	75	WAN
isdnu	76	WAN
isup	179	WAN
l2vlan	135	WAN
l3ipvlan	136	WAN
l3ipxvlan	137	WAN
lapb	16	WAN
lapd	77	WAN
lapf	119	WAN
localTalk	42	WAN
mediaMailOverIp	139	WAN
mfSigLink	167	WAN
miox25	38	WAN
modem	48	WAN
mpls	166	WAN
mplsTunnel	150	WAN
msdsl	143	WAN
mvl	191	WAN
myrinet	99	WAN
nfas	175	WAN
nsip	27	WAN
para	34	WAN
plc	174	WAN
pos	171	WAN
ppp	23	WAN
pppMultilinkBundle	108	WAN
primaryISDN	21	WAN
propBWAp2Mp	184	WAN

Table 10 WAN Ports

IANAifTypeDesc_32_1	IANAifType	Group Name
propCnls	89	WAN
propDocsWirelessDownstream	181	WAN
propDocsWirelessMaclayer	180	WAN
propDocsWirelessUpstream	182	WAN
propMultiplexor	54	WAN
propPointToPointSerial	22	WAN
propVirtual	53	WAN
propWirelessP2P	157	WAN
qllc	68	WAN
radioMAC	188	WAN
radsl	95	WAN
reachDSL	192	WAN
regular1822	2	WAN
rfc1483	159	WAN
rfc877x25	5	WAN
rs232	33	WAN
rsrb	79	WAN
sdci	280	WAN
sdlc	17	WAN
sdsl	96	WAN
shdsl	169	WAN
sip	31	WAN
slip	28	WAN
smdsDxi	43	WAN
smdsIcip	52	WAN
softwareLoopback	24	WAN
sonet	39	WAN
sonetOverheadChannel	185	WAN
sonetPath	50	WAN
sonetVT	51	WAN
srp	151	WAN
ss7SigLink	156	WAN
stackToStack	111	WAN
tdlc	116	WAN
termPad	91	WAN

Table 10 WAN Ports

IANAifTypeDesc_32_1	IANAifType	Group Name
tr008	176	WAN
trasnpHdlc	123	WAN
tunnel	131	WAN
ultra	29	WAN
usb	160	WAN
v11	64	WAN
v35	45	WAN
v36	65	WAN
v37	120	WAN
vdsl	97	WAN
virtualIpAddress	112	WAN
voiceEM	100	WAN
voiceEncap	103	WAN
voiceFXO	101	WAN
voiceFXS	102	WAN
voiceOverAtm	152	WAN
voiceOverFrameRelay	153	WAN
voiceOverIp	104	WAN
x213	93	WAN
x25huntGroup	122	WAN
x25mlp	121	WAN
x25ple	40	WAN

Table 10 WAN Ports

Port Interface Types By IANAifType

All of the port interface types loaded into Entuity are listed here, ordered by IANAifType. Those types that are not identified as belonging to either of the WAN or LAN groups, marked as belonging to the other group.

IANAifTypeDesc_32_1	IANAifType	Group Name
unknown	1	other
regular1822	2	WAN
hdh1822	3	WAN
ddnX25	4	WAN
rfc877x25	5	WAN
ethernetCsmacd	6	LAN

IANAifTypeDesc_32_1	IANAifType	Group Name
iso88023Csmacd	7	LAN
iso88024TokenBus	8	LAN
iso88025TokenRing	9	LAN
iso88026Man	10	LAN
starLan	11	LAN
proteon10Mbit1	12	LAN
proteon80Mbit	13	LAN
hyperchannel	14	LAN
fddi	15	LAN
lapb	16	WAN
sdlc	17	WAN
ds1	18	WAN
e1	19	WAN
basicISDN	20	WAN
primaryISDN	21	WAN
propPointToPointSerial	22	WAN
ppp	23	WAN
softwareLoopback	24	WAN
eon	25	WAN
ethernet3Mbit	26	LAN
nsip	27	WAN
slip	28	WAN
ultra	29	WAN
ds3	30	WAN
sip	31	WAN
frameRelay	32	WAN
rs232	33	WAN
para	34	WAN
arcnet	35	WAN
arcnetPlus	36	WAN
atm	37	WAN
miox25	38	WAN
sonet	39	WAN
x25ple	40	WAN
iso88022llc	41	LAN
localTalk	42	WAN

IANAifTypeDesc_32_1	IANAifType	Group Name
smdsDxi	43	WAN
frameRelayService	44	WAN
v35	45	WAN
hssi	46	WAN
hippi	47	WAN
modem	48	WAN
aal5	49	WAN
sonetPath	50	WAN
sonetVT	51	WAN
smdsIcip	52	WAN
propVirtual	53	WAN
propMultiplexor	54	WAN
ieee80212	55	LAN
fibreChannel	56	LAN
hippiInterface	57	WAN
frameRelayInterconnect	58	WAN
aflane8023	59	LAN
aflane8025	60	LAN
cctEmul	61	WAN
fastEther	62	LAN
isdn	63	WAN
v11	64	WAN
v36	65	WAN
g703at64k	66	WAN
g703at2mb	67	WAN
qllc	68	WAN
fastEtherFX	69	LAN
channel	70	LAN
ieee80211	71	WAN
ibm370parChan	72	LAN
escon	73	LAN
dlsw	74	WAN
isdns	75	WAN
isdnu	76	WAN
lapd	77	WAN
ipSwitch	78	WAN

IANAifTypeDesc_32_1	IANAifType	Group Name
rsrb	79	WAN
atmLogical	80	WAN
ds0	81	WAN
ds0Bundle	82	WAN
bsc	83	WAN
async	84	WAN
cnr	85	WAN
iso88025Dtr	86	LAN
eplrs	87	WAN
arap	88	WAN
propCnls	89	WAN
hostPad	90	WAN
termPad	91	WAN
frameRelayMPI	92	WAN
x213	93	WAN
adsl	94	WAN
radsl	95	WAN
sdsl	96	WAN
vdsl	97	WAN
iso88025CRFPInt	98	LAN
myrinet	99	WAN
voiceEM	100	WAN
voiceFXO	101	WAN
voiceFXS	102	WAN
voiceEncap	103	WAN
voiceOverIp	104	WAN
atmDxi	105	WAN
atmFuni	106	WAN
atmIma	107	WAN
pppMultilinkBundle	108	WAN
ipOverCdlc	109	WAN
ipOverClaw	110	WAN
stackToStack	111	WAN
virtualIpAddress	112	WAN
mpc	113	LAN
ipOverAtm	114	WAN

IANAifTypeDesc_32_1	IANAifType	Group Name
iso88025Fiber	115	LAN
tdlc	116	WAN
gigabitEthernet	117	LAN
hdlc	118	WAN
lapf	119	WAN
v37	120	WAN
x25mlp	121	WAN
x25huntGroup	122	WAN
trasnpHdlc	123	WAN
interleave	124	WAN
fast	125	WAN
ip	126	WAN
docsCableMaclayer	127	WAN
docsCableDownstream	128	WAN
docsCableUpstream	129	WAN
a12MppSwitch	130	WAN
tunnel	131	WAN
coffee	132	WAN
ces	133	WAN
atmSubInterface	134	WAN
l2vlan	135	WAN
l3ipvlan	136	WAN
l3ipxvlan	137	WAN
digitalPowerline	138	WAN
mediaMailOverIp	139	WAN
dtm	140	WAN
dcn	141	WAN
ipForward	142	WAN
msdsl	143	WAN
ieee1394	144	WAN
if-gsn	145	WAN
dvbRccMacLayer	146	WAN
dvbRccDownstream	147	WAN
dvbRccUpstream	148	WAN
atmVirtual	149	WAN
mplsTunnel	150	WAN

IANAifTypeDesc_32_1	IANAifType	Group Name
srp	151	WAN
voiceOverAtm	152	WAN
voiceOverFrameRelay	153	WAN
idsl	154	WAN
compositeLink	155	WAN
ss7SigLink	156	WAN
propWirelessP2P	157	WAN
frForward	158	WAN
rfc1483	159	WAN
usb	160	WAN
ieee8023adLag	161	LAN
bgppolicyaccounting	162	WAN
frf16MfrBundle	163	WAN
h323Gatekeeper	164	WAN
h323Proxy	165	WAN
mpls	166	WAN
mfSigLink	167	WAN
hdsl2	168	WAN
shdsl	169	WAN
ds1FDL	170	WAN
pos	171	WAN
dvbAsiIn	172	WAN
dvbAsiOut	173	WAN
plc	174	WAN
nfas	175	WAN
tr008	176	WAN
gr303RDT	177	WAN
gr303IDT	178	WAN
isup	179	WAN
propDocsWirelessMaclayer	180	WAN
propDocsWirelessDownstream	181	WAN
propDocsWirelessUpstream	182	WAN
hiperlan2	183	WAN
propBWAp2Mp	184	WAN
sonetOverheadChannel	185	WAN
digitalWrapperOverheadChannel	186	WAN

IANAifTypeDesc_32_1	IANAifType	Group Name
aal2	187	WAN
radioMAC	188	WAN
atmRadio	189	WAN
imt	190	WAN
mvl	191	WAN
reachDSL	192	WAN
frDlciEndPt	193	WAN
atmVciEndPt	194	WAN
opticalChannel	195	LAN
opticalTransport	196	LAN
propAtm	197	WAN
voiceOverCable	198	WAN
infiniband	199	WAN
teLink	200	WAN
q2931	201	WAN
virtualTg	202	WAN
sipTg	203	WAN
sipSig	204	WAN
docsCableUpstreamChannel	205	WAN
econet	206	WAN
pon155	207	WAN
pon622	208	WAN
bridge	209	WAN
linegroup	210	WAN
voiceEMFGD	211	WAN
voiceFGDEANA	212	WAN
voiceDID	213	WAN
docsOfdmDownstream	277	WAN
docsOfdmaUpstream	278	WAN
gfast	279	WAN
sdci	280	Programmable Controller
mpegTransport	214	
sixToFour	215	
gtp	216	
pdnEtherLoop1	217	

IANAifTypeDesc_32_1	IANAifType	Group Name
pdnEtherLoop2	218	
opticalChannelGroup	219	
homepna	220	
gfp	221	
ciscoISLvlan	222	
actelisMetaLOOP	223	
fcipLink	224	
rpr	225	
qam	226	
lmp	227	
cblVectaStar	228	
docsCableMCmtsDownstream	229	
adsl2	230	DEPRECATED
--		
macSecControlledIF	231	
macSecUncontrolledIF	232	
aviciOpticalEther	233	
atmbond	234	
voiceFGDOS	235	
mocaVersion1	236	
-- as documented in information provided privately to IANA		
ieee80216WMAN	237	
adsl2plus	238	
-- Version 2 Plus and all variants		
dvbRcsMacLayer	239	
dvbTdm	240	
dvbRcsTdma	241	
x86Laps	242	
wwanPP	243	
wwanPP2	244	
voiceEBS	245	
ifPwType	246	
ilan	247	
pip	248	
aluELP	249	

IANAifTypeDesc_32_1	IANAifType	Group Name
gpon	250	
vdsl2	251	
capwapDot11Profile	252	
capwapDot11Bss	253	
capwapWtpVirtualRadio	254	
bits	255	
docsCableUpstreamRfPort	256	
cableDownstreamRfPort	257	
vmwareVirtualNic	258	
ieee802154	259	
otnOdu	260	
otnOtu	261	
ifVfiType	262	
g9981	263	
g9982	264	
g9983	265	
aluEpon	266	
aluEponOnu	267	
aluEponPhysicalUni	268	
aluEponLogicalLink	269	
aluGponOnu	270	
aluGponPhysicalUni	271	
vmwareNicTeam	272	

Appendix D Entuity RESTful API Resources

The Entuity implementation of the RESTful API includes these resources:

- domainFilters
- domainFilters/*filterName*
- eventFilters
- eventFilters/*filterName*
- events
- eventTypes
- incidentFilters
- incidentFilter/*filterName*
- incidents
- incidents/*ID*
- IncidentTypes
- info
- inventory
- inventory/*id*
- Objects/ID/attributes
- objects/*ID*/*attributeName*
- objects/*ID*/*associationName*
- objects/*ID*/configManagement
- portManagement
- servers
- servers/*id*
- tools
- userGroups
- userGroups/*ID*
- usersGroups/*ID*/tools
- users
- users/*ID*
- version
- views
- views/*id*
- views/*id*/objects.

Some resources expect input in the form of:

- Query parameters, for example:

  `http://entuity_server/api/someResource?param=value`

  Where ***value*** must be URL encoded, for example the space in **hello world** must be encoded as **hello%20world**. (For a list of the characters that must be encoded refer to http://www.w3schools.com/tags/ref_urlencode.asp.)

- HTTP content (Entity). The Entity may be represented as one of:
  - XML, with `Content-Type:application/xml`.
  - JSON (JavaScript Object Notation), with `Content-Type:application/json`.

  When sending the entity in the request, you must specify the Content-Type header. If using the curl tool you can use the `-H` argument to specify the header, as in these command extracts:

```
curl -H Content-Type:application/xml ...
curl -H Content-Type:application/json ...
```

The curl examples included with the documentation have been verified using different versions of the generic curl install on both Windows and Linux operating systems.

Most, but not all, resources will return resource representations in either XML or JSON. You can specify the media query parameter using a header field, for example to set the format of the response as in these command extracts:

```
curl -H Accept:application/xml ...
curl -H Accept:application/json ...
```

Alternatively you can specify the representation by supplying a media query parameter with a value of either **xml** or **json**:

`http://entuity_server/api/info?media=xml`

`http://entuity_server/api/info?media=json`

Figure 1 JSON Response

Each response has a response code, indicating the success or failure of the request. These are standard HTTP specification response codes:

```
200-299: indicating success
300-399: indicating redirection: clients should repeat request at
redirected location
400-499: indicating a problem with a client request
500-599: indicating a problem on a server side
```

By default HTTP methods operate on the resources local to the server you are connecting to. However, if the server you are connecting to has remote servers configured you can work with any of them. You can qualify the server you want to be working with by using a query parameter *serverId*, for example:

```
http://entuity_server/api/info?serverId=long-id-of-the-remote-server
```

domainFilters

This resource lists key attributes for identifying available domain filters.

Method	Description
GET	Lists available domain filters.
POST	Creates a new domain filter.

Table 11 domainFilters GET Method

domainFilters GET Method

Response

The GET response includes a list of domain filters and includes identifying information for each domain filter.

Name	Description
id	Domain filter id unique to the server.
Name	Domain filter name.
serverId	Entuity Server Id on which the filter is defined.

Table 12 domainFilters Get Method Response

domainFilters GET Examples

Allows you to retrieve attributes important to domain filters, for example:

- This command retrieves domain filters and requests the response is in XML:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/domainFilters
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<items count="2">
    <item xsi:type="namedItem" name="All Objects" id="1"
serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
    <item xsi:type="namedItem" name="Infrastructure Only" id="2"
serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
</items>
```

- This command retrieves domain filters and requests the response is in JSON:

```
curl -u admin:admin -X GET http://entuity_server/api/
domainFilters?media=json
{
  "items" : [ {
    "serverId" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
    "id" : "1",
    "name" : "All Objects"
  }, {
    "serverId" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
    "id" : "2",
    "name" : "Infrastructure Only"
  } ],
  "count" : 2
}
```

domainFilters POST Method

Creates a new domain filter.

Request Parameters

Name	Description
Name	Name of the domain filter.
rules	The array of rules defining a filter.

Table 13 domainFilters POST Request

Response

Name	Description
Name	Name of the domain filter.

Table 14 domainFilters POST Response

Name	Description
systemFilter	Whether the filter is a system filter
rules	The array of rules defining a filter.

Table 14 domainFilters POST Response

domainFilters POST Examples

```
curl -u admin:admin http://entuity_server/api/domainFilters?media=json
-X POST -H "Content-Type: application/json" -d
'{
  "name" : "Filter A",
  "rules" : [ {
    "SRCTYPE" : "4",
    "DEVNAME" : "Two",
    "ZONENAME" : "Default"
  } ]
}'
{
  "name" : "Filter A",
  "systemFilter" : false,
  "rules" : [ {
    "SRCTYPE" : "4",
    "DEVNAME" : "Two",
    "ZONENAME" : "Default"
  } ]
}
```

domainFilters/*filterName*

This resource represents a set of operations on a particular domain filter.

Method	Description
GET	Shows detailed information about this filter.
PUT	Modifies the parameters of the filter.
DELETE	Deletes selected filter.

Table 15 domainFilters/*filterName* Methods

domainFilters/*filterName* GET Method

Response

Name	Description
Name	Filter name.
systemFilter	Whether the filter is a system filter.
rules	The array of rules defining a filter.

Table 16 domainFilters GET Method

Examples

```
curl -u admin:admin http://entuity_server/api/domainFilters/
Filter%20A?media=json
{
  "name" : "Filter A",
  "systemFilter" : false,
  "rules" : [ {
    "SRCTYPE" : "4",
    "DEVNAME" : "Two",
    "ZONENAME" : "Default"
  } ]
}
```

domainFilters/*filterName* PUT Method

Modifies a filter.

Request

Name	Description
Name	Filter name.
rules	The array of rules defining a filter.

Table 17 domainFilters/*filterName* PUT Request

Response

The filter after update, as detailed GET method would return it.

Name	Description
Name	Filter name.

Table 18 domainFilters/*filterName* PUT Response

Name	Description
systemFilter	Whether the filter is system one.
rules	The list of rules defining a filter.

Table 18 domainFilters/*filterName* PUT Response

Examples

```
curl -u admin:admin http://entuity_server/api/domainFilters/
Filter%20A?media=json
-X PUT -H "Content-Type: application/json" -d
'{
  "name" : "B",
  "rules" : [ {
    "SRCTYPE" : "1024",
    "DEVTYPE" : "158",
    "ZONENAME" : "Default"
  } ]
}'
{
  "name" : "B",
  "rules" : [ {
    "SRCTYPE" : "1024",
    "DEVTYPE" : "158",
    "ZONENAME" : "Default"
  } ]
}
```

domainFilters/*filterName* DELETE Method

Deletes a filter.

Request

No additional parameters needed.

Response

"OK", if operation was successful, and an error description otherwise.

domainFilters DELETE Example

```
curl -u admin:admin http://entuity_server/api/domainFilters/
B?media=json
-X DELETE -H "Content-Type: application/json"
```

```
"OK"
```

eventFilters

This resource lists key attributes for identifying available event filters.

Method	Description
GET	Lists available event filters.
POST	Creates an event filter.

Table 19 eventFilters GET Method

eventFilters GET Method

Response

The response includes a list of event filters and their attributes.

Attribute	Description
id	Event filter id unique to the server.
Name	Event filter name.
serverId	Entuity Server Id on which the resource resides.

Table 20 eventFilters GET Method Response

eventFilters GET Examples

Allows you to retrieve attributes important to event filters, for example:

- This command retrieves event filters and requests the response is in XML:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/eventFilters
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="2">
    <item xsi:type="namedItem" name="All Events" id="1"
serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
    <item xsi:type="namedItem" name="Entuity System Events" id="1001"
serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
</items>
```

- This command retrieves event filters and requests the response is in JSON:

```
curl -u admin:admin -X GET http://entuity_server/api/
eventFilters?media=json
{
```

```
  "items" : [ {
    "serverId" : "d5f11137-be43-4c58-a547-6f4d68cb4e83",
    "id" : "1",
    "name" : "All Events"
  }, {
    "serverId" : "d5f11137-be43-4c58-a547-6f4d68cb4e83",
    "id" : "1001",
    "name" : "Entuity System Events"
  } ],
  "count" : 2
}
```

eventFilters POST Method

Creates new event filter.

Request

Name	Description
Name	Filter name.
selectedNames	The array of filter names.
passIP	When set to: ■ **true** indicates the filter should include events raised against devices that are not managed by Entuity. ■ **false** indicates the filter should only include events raised against devices that are managed by Entuity.

Table 21 eventFilters POST Method Request

Response

The newly created entry, as detailed GET method would return it.

Name	Description
Name	Filter name
selectedNames	The array of filter names.
systemFilter	When set to **true** indicates the filter is a system filter, when set to **false** it is a user defined filter.

Table 22 eventFilters POST Method Response

Name	Description
passIP	When set to: ■ **true** indicates the filter should include events raised against devices that are not managed by Entuity. ■ **false** indicates the filter should only include events raised against devices that are managed by Entuity.

Table 22 eventFilters POST Method Response

Examples

```
curl -u admin:admin http://entuity_server/api/eventFilters?media=json
-X POST -H "Content-Type: application/json" -d
'{
  "name" : "A",
  "selectedNames" : [
    "AvailMonitor Application Unavailable",
    "AvailMonitor High Latency Reaching Application Cleared"
  ],
  "passIP" : true
}'
{
  "name" : "A",
  "selectedNames" : [
    "AvailMonitor Application Unavailable",
    "AvailMonitor High Latency Reaching Application Cleared"
  ],
  "systemFilter" : false,
  "passIP" : true
}
```

eventFilters/*filterName*

This resource represents a set of operations on a particular event filter.

Method	Description
GET	Shows detailed information about this filter
PUT	Modifies the parameters of the filter
DELETE	Deletes selected filter

Table 23 eventFilters/*filterName* Methods

eventFilters/filterName GET Method

Response

Name	Description
Name	Filter name
selectedNames	The array of filter names.
systemFilter	Whether the filter is a system filter.
passIP	When set to: ■ **true** indicates the filter should include events raised against devices that are not managed by Entuity. ■ **false** indicates the filter should only include events raised against devices that are managed by Entuity.

Table 24 eventFilters/*filterName* GET Method Response

Examples

```
curl -u admin:admin http://entuity_server/api/eventFilters/
A?media=json

{
  "name" : "A",
  "selectedNames" : [
    "AvailMonitor Application Unavailable",
    "AvailMonitor High Latency Reaching Application Cleared"
  ],
  "systemFilter" : false,
  "passIP" : true
}
```

eventFilters/*filterName* PUT Method

Modifies a filter.

Request

Name	Description
Name	Filter name.
selectedNames	The array of filter names.
passIP	Whether the filter should include devices not under management.

Table 25 eventFilters/*filterName* PUT Method Request

Response

Name	Description
Name	Filter name.
selectedNames	The array of filter names.
systemFilter	Whether the filter is a system filter.
passIP	When set to: ■ **true** indicates the filter should include events raised against devices that are not managed by Entuity. ■ **false** indicates the filter should only include events raised against devices that are managed by Entuity.

Table 26 eventFilters/*filterName* PUT Method Response

Examples

```
curl -u admin:admin http://entuity_server/api/eventFilters/
A?media=json
-X PUT -H "Content-Type: application/json" -d
'{
  "name" : "B",
  "selectedNames" : [
    "AvailMonitor Application Unavailable",
    "WAN Port Low Outbound Utilization Cleared",
    "AvailMonitor High Latency Reaching Application Cleared"
  ],
  "passIP" : false
}'
{
  "name" : "B",
  "selectedNames" : [
    "WAN Port Low Outbound Utilization Cleared",
    "AvailMonitor Application Unavailable",
    "AvailMonitor High Latency Reaching Application Cleared"
  ],
  "systemFilter" : false,
  "passIP" : false
}
```

eventFilters/*filterName* DELETE Method

Deletes a filter.

Request

No additional parameters needed.

Response

OK, if the operation was successful, and an error description otherwise.

Examples

```
curl -u admin:admin http://entuity_server/api/eventFilters/
B?media=json
-X DELETE -H "Content-Type: application/json"
"OK"
```

events

Method	Description
GET	Lists available events.

Table 27 events Method

events GET Method

Response

Response includes an array of event. Each event has following attributes.

Name	Description
description	Event description.
details	Event details.
objKey	Object key, consisting of StormWorks ID and Classic ID.
severity	Event severity.
sourceDescription	Source description.
impactDescription	Impact description.
timeStamp	Event timestamp.
eventID	ID of the event.
eventNumber	Event number.
eventCount	Event count.

Table 28 events GET Method

Example

```
curl -u admin:admin http://entuity_server/api/events?media=json

[ {
  "description" : "Port Speed Change",
  "details" : "Port speed changed from 10Mbps to 100Mbps",
  "objKey" : {
    "id" : 676,
    "compId" : {
      "ids" : [ 1, 1, 20, 0 ],
      "type" : 1,
      "invalid" : false,
      "networkPath" : false,
      "device" : false,
      "port" : true,
      "root" : false,
      "view" : false,
      "dsObject" : false
    },
    "viewName" : null
  },
  "severity" : 2,
  "sourceDescription" : "top2960  [ Fa0/19 ] FastEthernet0/19",
  "impactDescription" : "HOSTS: 00:15:5d:04:1c:01 a4:ba:db:f0:87:f8 ",
  "timeStamp" : 1457474761,
  "eventID" : 524309,
  "eventNumber" : 66,
  "eventCount" : 0
}, {
  ...removed for brevity...
} ]
```

eventTypes

Returns the list of event types

Method	Description
GET	Lists available event types

Table 29 eventTypes Method

eventTypes GET Method

Response

Response includes an array of event types, each of them having the following format:

Name	Description
Name	Name of this event type
severity	Severity of this event type

Table 30 eventTypes GET Method Response

Example

```
curl -u admin:admin http://entuity_server/api/eventTypes?media=json

[ {
    "name" : "ATM VCC High Inbound Utilization",
    "severity" : 6
  }, {
  ...removed for brevity...
  }, {
    "name" : "WAN Port Low Outbound Utilization Cleared",
    "severity" : 2
} ]
```

incidentFilters

This resource lists key attributes for identifying available incident filters.

Method	Description
GET	Lists available incident filters.
POST	Creates a new incident filter.

Table 31 incidentFilters Methods

incidentFilters GET Method

Response

The response includes a list of incident filters and their attributes.

Attribute	Description
serverId	Entuity Server Id on which resource resides.
id	Incident filter id unique to the server.
Name	Incident filter name.

Table 32 incidentFilters Response

incidentFilters Examples

Allows you to retrieve attributes important to incident filters. This example retrieves incident filters and:

- Uses the Accept header to request the response is in XML:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/incidentFilters
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="2">
    <item xsi:type="namedItem" name="All Incidents" id="1"
serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
    <item xsi:type="namedItem" name="Entuity System Incidents" id="2"
serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
</items>
```

- Requests the response is in JSON:

```
curl -u admin:admin -X GET http://entuity_server/api/
incidentFilters?media=json
{
  "items" : [ {
    "serverId" : "d5f11137-be43-4c58-a547-6f4d68cb4e83",
    "id" : "1",
    "name" : "All Incidents"
  }, {
    "serverId" : "d5f11137-be43-4c58-a547-6f4d68cb4e83",
    "id" : "2",
    "name" : "Entuity System Incidents"
  } ],
```

```
  "count" : 2
}
```

incidentFilters POST Method

Creates new incident filter.

Request

Name	Description
Name	Filter name
selectedNames	The array of filter names
passIP	Whether the filter should include devices not under management

Table 33 incidentFilters POST Method Request

Response

Name	Description
Name	Filter name
selectedNames	The array of filter names
systemFilter	Whether the filter is a system filter
passIP	Whether the filter should include devices not under management

Table 34 incidentFilters POST Method Response

Examples

```
curl -u admin:admin http://entuity_server/api/
incidentFilters?media=json
-X POST -H "Content-Type: application/json" -d
'{
  "name" : "A",
  "selectedNames" : [
    "AP Antenna Host Count Abnormality",
    "AP Not Associated With Controller"
  ],
  "passIP" : false
}'
{
  "name" : "A",
  "selectedNames" : [
    "AP Antenna Host Count Abnormality",
```

```
    "AP Not Associated With Controller"
  ],
  "systemFilter" : false,
  "passIP" : false
}
```

incidentFilter/*filterName*

This resource represents a set of operations on a particular incident filter.

Method	Description
GET	Shows detailed information about this filter.
PUT	Modifies the parameters of the filter.
DELETE	Deletes selected filter.

Table 35 incidentFilter/*filterName* Methods

incidentFilter/*filterName* GET Method

Response

Name	Description
Name	Incident filter name.
systemFilter	Whether this filter is a system one.
rules	List of rules that this filter consists of.

Table 36 incidentFilter/*filterName* GET Method

Examples

```
curl -u admin:admin http://entuity_server/api/domainFilters/
A?media=json

{
  "name" : "A",
  "selectedNames" : [
    "AP Antenna Host Count Abnormality",
    "AP Not Associated With Controller"
  ],
  "systemFilter" : false,
  "passIP" : false
```

```
}

curl -u admin:admin http://localhost/api/domainFilters/2?media=xml

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<eventFilterInfo systemFilter="false" passIP="true" name="A">
  <selectedNames>
    <filterName>AvailMonitor High Latency</filterName>
    <filterName>AvailMonitor Application Problem</filterName>
  </selectedNames>
</eventFilterInfo>
```

incidentFilter/*filterName* PUT Method

Modifies a filter.

Request

Name	Description
Name	Filter name
rules	The list of rules defining a filter

Table 37 incidentFilter/*filterName* PUT Method Request

Response

Name	Description
Name	Filter name
systemFilter	Whether the filter is system one
rules	The list of rules defining a filter

Table 38 incidentFilter/*filterName* PUT Method Response

Examples

```
curl -u admin:admin http://entuity_server/api/incidentFilters/
A?media=json
-X PUT -H "Content-Type: application/json" -d
'{
  "name" : "B",
  "selectedNames" : [
    "AP Antenna Host Count Abnormality",
    "AP Antenna Power Change Frequency High",
    "AP Not Associated With Controller"
```

```
  ],
  "systemFilter" : false,
  "passIP" : true
}'
{
  "name" : "B",
  "selectedNames" : [
    "AP Antenna Host Count Abnormality",
    "AP Antenna Power Change Frequency High",
    "AP Not Associated With Controller"
  ],
  "systemFilter" : false,
  "passIP" : true
}
```

incidentFilter/*filterName* DELETE Method

Deletes a filter.

Request

No additional parameters needed.

Response

"OK", if operation was successful, and an error description otherwise.

Examples

```
curl -u admin:admin http://entuity_server/api/incidentFilters/
B?media=json
-X DELETE -H "Content-Type: application/json"
"OK"
```

incidents

Returns the list of incidents.

Method	Description
GET	Lists available incidents.

Table 39 incidents Method

incidents GET Method

Response

Response includes an array of event. Each event has following attributes:

Name	Description
description	Event description.
details	Event details.
objKey	Object key, consisting of StormWorks ID and Classic ID.
severity	Event severity.
sourceDescription	Source description.
impactDescription	Impact description.
timeStamp	Event timestamp.
eventID	ID of the event.
eventNumber	Event number.
eventCount	Event count.

Table 40 incidents GET Method

Example

```
curl -u admin:admin http://entuity_server/api/incidents?media=json
```

```
[ {
  "description" : "Network Outage",
  "details" : "Entuity Server disconnected from network? ",
  "objKey" : {
    "id" : 0,
    "compId" : {
      "ids" : [ 2048, 2130706433, 0, 0 ],
      "type" : 2048,
      "invalid" : false,
      "networkPath" : false,
      "device" : false,
      "port" : false,
      "root" : false,
      "view" : false,
      "dsObject" : false
    },
```

```
    "viewName" : null
  },
  "severity" : 10,
  "sourceDescription" : "127.0.0.1",
  "impactDescription" : "",
  "timeStamp" : 1457013255,
  "eventID" : 983047,
  "eventNumber" : 2,
  "eventCount" : 1
} ]
```

incidents/*ID*

Represents set of operations applicable to an incident.

Method	Description
PUT	Close the incident

Table 41 incidents/*ID* PUT Method

incidents/*ID* PUT Method

Request

All the parameters are sent in an URL. If the request contains attribute "expire" and its value is "1", "yes" or "true", then the incident is marked as expired.

Response

"OK" if the incident was properly closed, an error message otherwise.

Examples

```
curl -u admin:admin http://entuity_server/api/incidents/
983047?media=json
-X PUT -H "Content-Type: application/json"
"OK"
```

IncidentTypes

Returns the list of incident types

Method	Description
GET	Lists available incident types

Table 42 incidentTypes GET Method

incidentTypes GET Method

Response

Response includes an array of event types, each of them having the following format:

Name	Description
name	Name of this incident type

Table 43 incidentTypes GET Method Response

Example

```
curl -u admin:admin http://entuity_server/api/incidentTypes?media=json

[ {
  "name" : "AP Antenna Channel Change Frequency High"
}, {
  ...removed for brevity...
}, {
  "name" : "Wireless Controller High Number of Connected APs"
} ]
```

info

The resource returns key information about the installed product.

Method	Description
GET	Returns brief information about the installed product.

Table 44 info GET Method

info GET Method

Response

This resource identifies key attributes of the Entuity server.

Attribute	Description
hostAddress	The host name for accessing the product.
id	Server ID.
product	Product edition.
sslAccess	Specifies to use HTTP (false) or HTTPS (true).
version	Version of the product.

Table 45 info Response

Attribute	Description
versionDisplay	User-friendly version string of the product.
webPort	The port number for accessing the product over HTTP(S).

Table 45 info Response

info Examples

Allows you to retrieve attributes important to identifying the Entuity server, for example:

- This command retrieves server information and requests the response is in XML:

```
curl -u admin:admin http://entuity_server/api/info?media=xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<serverInfo sslAccess="false" webPort="80"
hostAddress="entuity_server" product="EYE" versionDisplay="Entuity
15.5" version="15.5.0.p0" id="921c3f82-bcdf-4fef-a5e8-7d2524928d96"/>
```

- This command retrieves server information and requests the response is in JSON:

```
curl -u admin:admin http://entuity_server/api/info?media=json
{
  "id" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
  "version" : "15.5.0.p0",
  "versionDisplay" : "Entuity 15.5",
  "product" : "EYE",
  "hostAddress" : "entuity_server",
  "webPort" : 80,
  "sslAccess" : false
}
```

inventory

Provides access to the device inventory on the identified Entuity server or add a new device to the inventory.

Method	Description	Formats
GET	List the contents of the inventory.	XML, JSON
POST	Queue a device to be added to the inventory of a server.	XML, JSON

Table 46 inventory Methods

inventory GET Method

Request Parameters

None

Response Data Keys

Name	Description
Items	Number of devices in the inventory.
Item	List of device summary details: ■ *name*, device display Name. ■ *id*, unique device identifier. This is the identifier assigned to the object when it is first taken under management by Entuity. It is therefore different from the StormWorks identifier (*dsObjectId*) assigned when the object's properties are discovered. ■ *polledName*, DNS name or IP address. ■ *serverId*, server identifier.

Table 47 inventory Response Data Keys

inventory GET Examples

Allows you to retrieve attributes for each device under management of the Entuity server, for example:

- This command retrieves device inventory information and requests the response is in XML:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/inventory
<items count="2">
    <item xsi:type="device" polledName="10.44.1.116"
name="10.44.1.116" id="5" serverId="4b9d48fa-e72f-4e03-b7c3-
0a853ddef8a3"/>
    <item xsi:type="device" polledName="10.44.1.118"
name="10.44.1.118" id="33" serverId="4b9d48fa-e72f-4e03-b7c3-
0a853ddef8a3"/>
</items>
```

- This command retrieves device inventory information and requests the response is in JSON:

```
curl -u admin:admin -H Accept:application/json -X GET http://
entuity_server/api/inventory
{
  "items" : [ {
    "serverId" : "4b9d48fa-e72f-4e03-b7c3-0a853ddef8a3",
```

```
    "name" : "e2821",
    "id" : 42,
    "polledName" : "e2821"
  }, {
    "serverId" : "4b9d48fa-e72f-4e03-b7c3-0a853ddef8a3",
    "name" : "R10",
    "id" : 8,
    "polledName" : "R10"
  }],
  "count" : 2
}
```


inventory POST Method

Request Parameters

Parameter	Description
authType	SNMP v3 authentication type, i.e. NONE, MD5, SHA.

Table 48 inventory Post Request Parameters

Parameter	Description
deviceType	Device type, one of: ■ Hub ■ Token Ring Switch ■ Ethernet Switch ■ ATM Switch ■ Router ■ Blade Center ■ User Created Node ■ VM Platform ■ Autonomous WAP ■ Firewall ■ VPN ■ Managed Host ■ Non-SNMP Device ■ Unclassified (Full) ■ PoE Midspan Injector ■ Load Balancer ■ SSL Proxy ■ Unclassified ■ Wireless Controller ■ Uninterruptible Power Supply ■ Matrix Switch ■ Wide Area Application Service ■ Multiplexer.
encrType	SNMP v3 encryption type, i.e. NONE, DES, AES.
managementLevel	Management Level: ■ FULL ■ FULL_MGMT_PORT_ONLY ■ FULL_NO_PORTS ■ BASIC ■ PING_ONLY ■ WEB.
name	Device name.

Table 48 inventory Post Request Parameters

Parameter	Description
nameUsing	Device Name is determined using one of: ■ CUSTOM ■ IPADDRESS ■ POLLEDNAME ■ RESOLVABLENAME ■ RESOLVABLENAMEFQ ■ SYSTEMNAME.
polledName	DNS Name or IP Address.
protocol	Transport protocol: ■ IPv4 ■ IPv6.
readCommunity	SNMP v1/v2 read community string.
snmpPDUSize	Maximum size of SNMP PDU, 0 = system default.
snmpRetry	Number of SNMP retries, 0 = system default.
snmpTimeout	SNMP timeout in seconds, 0 = system default.
snmpType	SNMP type: ■ v1 ■ v2c ■ v3 ■ v1/2.
username	SNMP v3 user name.
vmAccessKey	Access key (Amazon platform only).
vmPassword	Virtual platform password (Non Amazon virtual platforms).
vmPlatformType	Virtual Platform Type: ■ VMWARE_ESXi, ■ ORACLE_VM_MANAGER ■ HYPER_V ■ AMAZON_WEB_SERVICE.
vmSecretKey	Secret key (Amazon platform only).
vmURL	Virtual platform URL (Non Amazon virtual platforms).
vmUser	Virtual platform user name (Non Amazon virtual platforms).

Table 48 inventory Post Request Parameters

Response

A message indicating the device is queued for adding to the Entuity server inventory.

inventory POST Examples

Allows you to add devices to the Entuity server. You can only add devices with the correct and complete credentials, for example this command adds the apcr4 device to Entuity and uses the:

- XML notation, and therefore requires that you specify the `inventoryDevice` element. This example includes two header definitions, `Content-Type` identifies the request as using XML and `Accept` requires the response to also use XML:

```
curl -u admin:admin -H Content-Type:application/xml -H
Accept:application/xml -X POST http://entuity_server/api/inventory -d
"<inventoryDevice protocol='IPv4' managementLevel='FULL'
nameUsing='POLLEDNAME' polledName='apcr4' readCommunity='public' />"
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<result message="Queued"/>
```

- JSON notation. This example does not specify the response format and therefore uses the default JSON:

```
curl -u admin:admin -H Content-Type:application/json -X POST http://
entuity_server/api/inventory -d '{"protocol":"IPv4",
"managementLevel":"FULL", "nameUsing":"POLLEDNAME",
"polledName":"apcr4", "readCommunity"="public" }'
```

```
{
  "message" : "Queued"
}
```

inventory/*id*

List, update or delete an inventory device as identified through its device identifier.

Method	Description	Formats
GET	Get inventory device details.	XML, JSON
PUT	Change an inventory device's details.	XML, JSON
DELETE	Remove a device from the inventory.	

Table 49 inventory/*id* Method Summary

Request Parameters

None

Response Data Keys

Parameter	Description
authType	SNMP v3 authentication type, i.e. NONE, MD5, SHA.

Table 50 inventory/*id* Response Data Keys

Parameter	Description
capabilities	Device Capabilities, i.e. Routing, Switching, Switching & Routing.
certified	Device has been certified.
Context	SNMPv3 context.
deviceType	Device type, one of: ■ Hub ■ Token Ring Switch ■ Ethernet Switch ■ ATM Switch ■ Router ■ Blade Center ■ User Created Node ■ VM Platform ■ Autonomous WAP ■ Firewall ■ VPN ■ Managed Host ■ Non-SNMP Device ■ Unclassified (Full) ■ PoE Midspan Injector ■ Load Balancer ■ SSL Proxy ■ Unclassified ■ Wireless Controller ■ Uninterruptible Power Supply ■ Matrix Switch ■ Wide Area Application Service ■ Multiplexer.
dsObjectId	Device's StormWorks identifier (*dsObjectId*) that is assigned when the object's properties are discovered.
encrType	SNMP v3 encryption type, i.e. None, DES or AES.
id	Device's unique identifier assigned to the object when it is first taken under management by Entuity. It is therefore different from the StormWorks identifier (*dsObjectId*) assigned when the object's properties are discovered.
managementIP	Management IP address.

Table 50 inventory/*id* Response Data Keys

Parameter	Description
managementLevel	Management Level: ■ FULL ■ FULL_MGMT_PORT_ONLY ■ FULL_NO_PORTS ■ BASIC ■ PING_ONLY ■ WEB.
name	Device name.
nameUsing	Device Name is determined using either ■ CUSTOM ■ IPADDRESS ■ POLLEDNAME ■ RESOLVABLENAME ■ RESOLVABLENAMEFQ ■ SYSTEMNAME.
polledName	DNS Name or IP Address
protocol	Transport protocol: ■ IPv4, ■ IPv6
readCommunity	SNMP v1/v2 read community string
serverId	Server identifier
snmpPDUSize	Maximum size of SNMP PDU, 0 = system default.
snmpRetry	Number of SNMP retries, 0 = system default.
snmpTimeout	SNMP timeout in seconds, 0 = system default.
snmpType	SNMP type: ■ v1, ■ v2c, ■ v3, ■ v1/2.
sysDescription	SNMP description field
sysLocation	SNMP retrieved system Location field
sysOid	SNMP system identifier field
username	SNMP v3 user name
vmAccessKey	Access key (Amazon platform only)
vmPassword	Virtual platform password (Non Amazon virtual platforms)

Table 50 inventory/*id* Response Data Keys

Parameter	Description
vmPlatformType	Virtual Platform Type: ■ VMWARE_ESXi, ■ ORACLE_VM_MANAGER ■ HYPER_V ■ AMAZON_WEB_SERVICE.
vmSecretKey	Secret key (Amazon platform only)
vmURL	Virtual platform URL (Non Amazon virtual platforms)
vmUser	Virtual platform user name (Non Amazon virtual platforms)
ZoneId	The zone identifier.

Table 50 inventory/*id* Response Data Keys

inventory/*id* GET Examples

The device identifier used with the inventory resource is the identifier assigned to the device when it is first taken under Entuity management. This identifier is not available through the Entuity web UI, however you can retrieve it by making an inventory request. (See *inventory GET Method*.)

The inventory device identifier is different to the device's StormWorks identifier (*dsObjectId*). The device's StormWorks identifier is the device identifier used with the views resource.

Using `inventory/id` GET you can retrieve details of the specified device using the device identifier, for example:

- This command retrieves inventory data on the device with the identifier 3, and specifies the XML response format:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/inventory/3
```

```
<inventoryDevice snmpPDUSize="0" snmpRetry="0" snmpTimeout="0"
encrType="NONE" authType="NONE" userName="" readCommunity="public"
snmpType="v1/v2c" protocol="IPv4" certified="Yes"
deviceType="Uninterruptible Power Supply" managementLevel="FULL"
capabilities="" sysLocation="London" sysDescription="APC Web/SNMP
Management Card (MB:v3.8.6 PF:v5.1.3 PN:apc_hw05_aos_513.bin
AF1:v5.1.3 AN1:apc_hw05_sumx_513.bin MN:AP9630 HR:05"
sysOid=".1.3.6.1.4.1.318.1.3.27" managementIP="10.44.1.65"
polledName="10.44.1.65" nameUsing="POLLEDNAME" dsObjectId="610"
name="10.44.1.65" id="3" serverId="4b9d48fa-e72f-4e03-b7c3-
0a853ddef8a3"/>
```

- This command retrieves inventory data on the device with the identifier 3, and specifies the JSON response format:

```
curl -u admin:admin -H Accept:application/json -X GET http://
entuity_server/api/inventory/3
```

```
{
  "serverId" : "4b9d48fa-e72f-4e03-b7c3-0a853ddef8a3",
  "id" : "3",
  "name" : "10.44.1.65",
  "dsObjectId" : 610,
  "snmpTimeout" : 0,
  "snmpRetry" : 0,
  "snmpPDUSize" : 0,
  "protocol" : "IPv4",
  "snmpType" : "v1/v2c",
  "nameUsing" : "POLLEDNAME",
  "certified" : "Yes",
  "polledName" : "10.44.1.65",
  "managementIP" : "10.44.1.65",
  "sysOid" : ".1.3.6.1.4.1.318.1.3.27",
  "sysDescription" : "APC Web/SNMP Management Card",
  "sysLocation" : "London",
  "deviceType" : "Uninterruptible Power Supply",
  "readCommunity" : "public",
  "userName" : "",
  "authPass" : null,
  "encrPass" : null,
  "vmUser" : null,
  "vmPassword" : null,
  "vmURL" : null,
  "vmAccessKey" : null,
  "vmSecretKey" : null,
  "vmPlatformType" : null,
  "authType" : "NONE",
  "encrType" : "NONE",
  "managementLevel" : "FULL"
}
```

inventory/*id* PUT Method

Request Parameters

Parameter	Description
authPass	SNMP v3 authentication password.
authType	SNMP v3 authentication type, i.e. NONE, MD5, SHA.
cliMethod	Connection method, e.g. SSH, Telnet.
cliPassword1	Password 1.
cliPassword2	Password 2.
cliPort	Connection port.
cliUsername	CLI User name.
context	SNMP v3 context.
encrType	SNMP v3 encryption type, i.e. NONE, DES, AES.
encrPass	SNMP v3 encryption password.
name	Device name.
nameUsing	Device Name is determined using either ■ CUSTOM ■ IPADDRESS ■ POLLEDNAME ■ RESOLVABLENAME ■ RESOLVABLENAMEFQ ■ SYSTEMNAME.
protocol	Transport protocol: ■ IPv4, ■ IPv6
readCommunity	SNMP v1/v2 read community string
snmpPDUSize	Maximum size of SNMP PDU, 0 = system default.
snmpRetry	Number of SNMP retries, 0 = system default.
snmpTimeout	SNMP timeout in seconds, 0 = system default.
snmpType	SNMP type: ■ v1, ■ v2c, ■ v3, ■ v1/2.
username	SNMP v3 user name
vmAccessKey	Access key (Amazon platform only)
vmPassword	Virtual platform password (Non Amazon virtual platforms)

Table 51 inventory/*id* Put Request Parameters

Parameter	Description
vmPlatformType	Virtual Platform Type: ■ VMWARE_ESXi, ■ ORACLE_VM_MANAGER ■ HYPER_V ■ AMAZON_WEB_SERVICE.
vmSecretKey	Secret key (Amazon platform only)
vmURL	Virtual platform URL (Non Amazon virtual platforms)
vmUser	Virtual platform user name (Non Amazon virtual platforms)

Table 51 inventory/*id* Put Request Parameters

Response data

Displays the updated inventory summary for the device.

inventory/*id* PUT Examples

The device identifier used with the inventory resource is the identifier assigned to the device when it is first taken under Entuity management. This identifier is not available through the Entuity web UI, however you can retrieve it by making an inventory request. (See *inventory GET Method*.)

The inventory device identifier is different to the device's StormWorks identifier (*dsObjectId*). The device's StormWorks identifier is the device identifier used with the views resource.

Using `inventory/id` PUT you can amend attributes important to managing the specified device by using the device identifier, for example:

- This command alters the list display name used for the device. The command uses the XML format and requests the response is also in XML:

```
curl -u admin:admin -H Content-Type:application/xml -H
Accept:application/xml -X PUT http://entuity_server/api/inventory/11 -
d "<inventoryDevice nameUsing='POLLEDNAME' />"

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<inventoryDevice snmpPDUSize="0" snmpRetry="0" snmpTimeout="0"
encrType="NONE" authType="NONE" userName="" readCommunity="public"
snmpType="v1/v2c" protocol="IPv4" certified="Yes"
deviceType="Uninterruptible Power Supply" managementLevel="FULL"
capabilities="" sysLocation="London" sysDescription="APC Web/SNMP
Management Card (MB:v3.8.6 PF:v5.1.3 PN:apc_hw05_aos_513.bin
AF1:v5.1.3 AN1:apc_hw05_sumx_513.bin MN:AP9630 HR:05"
sysOid=".1.3.6.1.4.1.318.1.3.27" managementIP="10.44.6.4"
polledName="apcr4" nameUsing="POLLEDNAME" dsObjectId="3707"
name="apcr4" id="11" serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83"/>
```

- This is a Windows specific format of a command that alters the list display name used for the device. The command uses the JSON format and accepts the default response format:

```
echo {"nameUsing":"SYSTEMNAME"} | curl -u admin:admin -H Content-
Type:application/json -X PUT http://entuity_server/api/inventory/11 -d
@-
{
  "serverId" : "d5f11137-be43-4c58-a547-6f4d68cb4e83",
  "id" : "11",
  "name" : "apcr4",
  "dsObjectId" : 3707,
  "snmpTimeout" : 0,
  "snmpRetry" : 0,
  "snmpPDUSize" : 0,
  "protocol" : "IPv4",
  "snmpType" : "v1/v2c",
  "nameUsing" : "SYSTEMNAME",
  "certified" : "Yes",
  "polledName" : "apcr4",
  "managementIP" : "10.44.6.4",
  "sysOid" : ".1.3.6.1.4.1.318.1.3.27",
  "sysDescription" : "APC Web/SNMP Management Card (MB:v3.8.6
PF:v5.1.3 PN:apc_hw05_aos_513.bin AF1:v5.1.3 AN1:apc_hw05_sumx_513.bin
MN:AP9630 HR:05",
  "sysLocation" : "London",
  "deviceType" : "Uninterruptible Power Supply",
  "readCommunity" : "public",
  "userName" : "",
  "authPass" : null,
  "encrPass" : null,
  "vmUser" : null,
  "vmPassword" : null,
  "vmURL" : null,
  "vmAccessKey" : null,
  "vmSecretKey" : null,
  "vmPlatformType" : null,
  "authType" : "NONE",
  "encrType" : "NONE",
```

```
  "managementLevel" : "FULL"
}
```

- This command changes the name of view 23. It also gives edit permission on the view to the user group London. The XML command reflects the structure of the XML definition, with the `accessGroup` element inside the `viewPathEditRequest` element:

```
curl -u admin:admin -H Content-Type:application/xml -H
Accept:application/xml -X PUT http://entuity_server/api/views/23 -d
"<viewPathEditRequest name='My View'>  <accessGroup editable='true'
userGroupName='London'/> </viewPathEditRequest>"
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<viewPathDetails owner="admin" incidentFilterName="All Incidents"
eventFilterName="All Events" manuallyPopulated="true"
domainFilterName="All Objects" baseViewAggregation="NONE"
displayName="My View" path="My View" id="23" serverId="d5f11137-be43-
4c58-a547-6f4d68cb4e83">
    <accessGroup editable="true" userGroupName="Administrators"/>
    <accessGroup editable="true" userGroupName="London"/>
</viewPathDetails>
```

- This command alters the list display name used for the device. The command uses the JSON format and accepts the default response format:

```
curl -u admin:admin -H Content-Type:application/json -X PUT http://
entuity_server/api/inventory/11 -d '{"nameUsing":"SYSTEMNAME"}'
```

```
{
  "serverId" : "d5f11137-be43-4c58-a547-6f4d68cb4e83",
  "id" : "11",
  "name" : "apcr4",
  "dsObjectId" : 3707,
  "snmpTimeout" : 0,
  "snmpRetry" : 0,
  "snmpPDUSize" : 0,
  "protocol" : "IPv4",
  "snmpType" : "v1/v2c",
  "nameUsing" : "SYSTEMNAME",
  "certified" : "Yes",
  "polledName" : "apcr4",
  "managementIP" : "10.44.6.4",
  "sysOid" : ".1.3.6.1.4.1.318.1.3.27",
```

```
  "sysDescription" : "APC Web/SNMP Management Card (MB:v3.8.6
PF:v5.1.3 PN:apc_hw05_aos_513.bin AF1:v5.1.3 AN1:apc_hw05_sumx_513.bin
MN:AP9630 HR:05",
  "sysLocation" : "London",
  "deviceType" : "Uninterruptible Power Supply",
  "readCommunity" : "public",
  "userName" : "",
  "authPass" : null,
  "encrPass" : null,
  "vmUser" : null,
  "vmPassword" : null,
  "vmURL" : null,
  "vmAccessKey" : null,
  "vmSecretKey" : null,
  "vmPlatformType" : null,
  "authType" : "NONE",
  "encrType" : "NONE",
  "managementLevel" : "FULL"
}
```

inventory/*id* DELETE Method

Deletes the specified managed object from the inventory.

Request

No request expected

Response

The command deletes the managed object and does not report on its success.

inventory/*id* Example

The device identifier used with the inventory resource is the identifier assigned to the device when it is first taken under Entuity management. This identifier is not available through the Entuity web UI, however you can retrieve it by making an inventory request. (See *inventory GET Method*.)

The inventory device identifier is different to the device's StormWorks identifier (*dsObjectId*). The device's StormWorks identifier is the device identifier used with the views resource.

Using `inventory/id DELETE` you can delete the nominated device from the nominated server. For example to delete the managed object with the id 48:

```
curl -u admin:admin -X DELETE http://entuity_server/api/inventory/48
```

Objects/ID/attributes

URL: objects/{swID}/attributes?includeDetails=true&name=name1&name=name2

Method	Description
GET	Lists attributes of the object.

Table 52 Objects/*ID*/attributes

Objects/*ID*/attributes GET Method

Gets the list of attributes of the object.

Request

The request may contain a list of attributes name to retrieve in a form of "name=value". By default, all attributes are returned. It is also possible to supply "includeDetails" flag. If it is set to "true", "yes", "t", "y" and "1" than it is assumed to be true. If it is set to any other value, it is false. If it is absent, than it is automatically set to true if there is at least one "name=value" entry, false otherwise.

The "includeDetails" controls the level of details, returned by this method. If false, only the attribute names will be returned. If true - attribute names together with their values will be returned.

Response

Name	Description
Name	The value of "name" attribute of the device. This is the only attribute returned if includeDetails is set to false.
displayName	Value of "displayName" attribute of the device
userEditable	Whether this attribute can be user-modified
userOverriden	Whether this attribute is set to polled or user-defined value
values	The list of values for this attribute

Table 53 Objects/*ID*/attributes GET Method Response

Examples

```
curl -u admin:admin http://entuity_server/api/objects/
611?media=json&includeDetails=y


[ {
  "name" : "certified",
  "displayName" : "Certified",
  "userEditable" : false,
  "userOverriden" : false,
```

```
    "values" : [ "1" ]
  }, {
...removed for brevity...
  }, {
    "name" : "typeName",
    "displayName" : "typeName",
    "userEditable" : false,
    "userOverriden" : false,
    "values" : [ "SwitchDevice" ]
  } ]
```

objects/*ID*/*attributeName*

Method	Description
GET	Lists values of one of attributes of this object
PUT	Modifies an attribute the object

Table 54 objects/ID/*attributeName* Methods

objects/ID/*attributeName* GET Method

Lists values of one of attributes of this object.

Response

The list of attributes, with entries according to the following format:

Name	Description
Name	The value of "name" attribute of the device
displayName	Value of "displayName" attribute of the device
userEditable	Whether this attribute can be user-modified
userOverriden	Whether this attribute is set to polled or user-defined value
values	The list of values for this attribute

Table 55 objects/ID/*attributeName* GET Method

Examples

```
curl -u admin:admin http://entuity_server/api/objects/611/
typeName?media=json

{
  "name" : "typeName",
```

```
  "displayName" : "typeName",
  "userEditable" : false,
  "userOverriden" : false,
  "values" : [ "SwitchDevice" ]
}
```

objects/ID/*attributeName* PUT Method

Modifies single attribute of the device

Request

An attribute definition:

Name	Description
Name	The name of attribute to be changed
userOverriden	Whether this attribute has user-supplied or polled value
values	The list of values for this attribute

Table 56 objects/*ID*/*attributeName* PUT Method

Response

Name	Description
Name	The value of "name" attribute of the device
displayName	Value of "displayName" attribute of the device
userEditable	Whether this attribute can be user-modified
userOverriden	Whether this attribute is set to polled or user-defined value
values	The list of values for this attribute

Table 57 objects/*ID*/*attributeName* PUT Method Response

Examples

```
curl -u admin:admin http://entuity_server/api/objects/611?media=json
-X PUT -H "Content-Type: application/json" -d
'{
  "name" : "TransferServer",
  "displayName" : "Configuration Transfer Server",
  "userOverriden" : true,
  "values" : [ "10.44.2.101" ]
}'
{
```

```
  "name" : "TransferServer",
  "displayName" : "Configuration Transfer Server",
  "userEditable" : true,
  "userOverriden" : true,
  "values" : [ "10.44.2.101" ]
}
```

objects/*ID*/*associationName*

Method	Description
GET	Lists attributes of this device

Table 58 objects/*ID*/*associationName*

objects/*ID*/*associationName* GET Method

Gets association for this object.

```
objects/{swID}/
{associationName}?includeVirtual=BOOL&includeHidden=BOOL
```

This method accepts two boolean parameters, which can be equal to either of "yes", "true", "y", "t" or "1". Any other value or absence of the parameter is treated as "false".

Response

The list of attributes, with entries according to the following format:

Name	Description
objectId	StormWorks ID of the associated object.
typeName	Type of the associated object.
displayName	Display name.
displayType	Display type.
hasServiceStatus	Whether the object has service status.

Table 59 objects/*ID*/*associationName* GET Method

Examples

```
curl -u admin:admin http://entuity_server/api/objects/611/
ports?media=json


[ {
    "objectId" : 658,
    "typeName" : "portEx",
```

```
      "displayName" : " [ Fa0/1 ] FastEthernet0/1",
      "displayType" : "",
      "hasServiceStatus" : true
    }, {
...removed for brevity...
  }, {
      "objectId" : 683,
      "typeName" : "portEx",
      "displayName" : " [ Gi0/2 ] GigabitEthernet0/2",
      "displayType" : "",
      "hasServiceStatus" : true
    } ]
```

objects/*ID*/*associations*

Method	Description
GET	Lists associations of this device

Table 60 objects/*ID*/*associationName*

objects/*ID*/*associations* GET Method

This method accepts a "showEmpty" parameter, which controls whether or not the empty associations should be listed. This parameter may be equal to either of "yes", "true", "y", "t" or "1". Any other value or absence of the parameter is treated as "false".

Response

The list of items, every one of which represents the name of the association.

Name	Description
Name	Association name.

Table 61 objects/*ID*/*associations* GET Method

Examples

```
curl -u admin:admin http://localhost/api/objects/655/
associations?showEmpty=yes&media=json


{
  "items": [
    "AdaptorUnits",
```

```
    "Annotation",
    "ChassisList",
    "ComputeBlades",
    "DeviceMIBPolls",
    "FabricExtenders",
    "FanModules",
    "Fans",
    "HostEthernets",
    "IPSLABaseCreators",
    "IPSLABasePollers",
    "IpAddresses",
    "LocalDisks",
  ...removed for brevity...
    "uDComponents19",
    "uDComponents20"
  ],
  "count": 60
}
```

objects/*ID*/configManagement

URL: objects/{swID}/configManagement

Method	Description
GET	Lists configuration management settings.
PUT	Modifies configuration management attribute of the device.

Table 62 objects/*ID*/configManagement Methods

objects/*ID*/configManagement GET Method

Gets the list of attributes of the object

Response

The list of attributes, with entries according to the following format.

Name	Description
Name	The value of "name" attribute of the device.

Table 63 objects/*ID*/configManagement GET Method Response

Name	Description
displayName	Value of "displayName" attribute of the device.
userEditable	Whether this attribute can be user-modified.
userOverriden	Whether this attribute is set to polled or user-defined value.
values	The list of values for this attribute: ■ NumberOfConfigsToArchive ■ configMonitorRetrievalScript ■ configMonitorPolicyRules ■ ConfigRetrievalEnabled ■ configMonitorTransferMethod ■ SNMPChangeDetectionEnabled ■ configMonitorExcludedDifference ■ cliMethod ■ cliUsername.

Table 63 objects/*ID*/configManagement GET Method Response

Examples

```
curl -u admin:admin
http://localhost/api/objects/605/
ports?includeVirtual=y&includeUnmanaged=y&media=json

{
  "items" : [ {
    "portAttributes" : {
      "compId" : {
        "ids" : [ 1, 2, 1, 0 ],
        "type" : 1,
        "invalid" : false,
        "root" : false,
        "networkPath" : false,
        "device" : false,
        "port" : true,
        "view" : false,
        "dsObject" : false
      },
      "connectedHostIps" : "",
      "connectedHostMacs" : "",
      "connectedHostNames" : "",
```

```
      "connectedHosts" : "",
      "displayName" : " [ Vl1 ] Vlan1",
      "ifDescr" : " [ Vl1 ] Vlan1",
      "ifIndex" : "1",
      "ifType" : "Prop. Virtual/Internal",
      "ipAddresses" : "10.44.1.41",
      "objectId" : 723,
      "portAdminStatus" : "up",
      "portAlias" : "",
      "portDescr" : "Vlan1",
      "portDuplex" : "Unknown",
      "portInSpeed" : 0,
      "portMac" : "00:19:06:d2:1e:c0",
      "portOperationalStatus" : "up",
      "portOutSpeed" : 0,
      "portShortDescr" : "[ Vl1 ]",
      "portSpare" : "No",
      "portVipStatus" : " ",
      "portVirtualIndicator" : "Virtual",
      "statusEventsEnabled" : 0,
      "statusFasterPolling" : 0,
      "timeOfLastChange" : null,
      "topoNodeState" : 292,
      "typeDisplayName" : "Switch Device",
      "typeName" : "portEx",
      "utilFasterPolling" : 0,
      "vlans" : ""
    }
  }, {
  ...removed for brevity...
  }, {
    "portAttributes" : {
      "compId" : {
        "ids" : [ 1, 2, 6, 0 ],
        "type" : 1,
        "invalid" : false,
```

```
        "root" : false,
        "networkPath" : false,
        "device" : false,
        "port" : true,
        "view" : false,
        "dsObject" : false
      },
      "connectedHostIps" : "",
      "connectedHostMacs" : "",
      "connectedHostNames" : "",
      "connectedHosts" : "",
      "displayName" : " [ Fa0/4 ] FastEthernet0/4",
      "ifDescr" : " [ Fa0/4 ] FastEthernet0/4",
    ...removed for brevity...
    }
  }, {
  ...removed for brevity...
  }, {
    "portAttributes" : {
      "compId" : {
        "ids" : [ 1, 2, 18, 0 ],
        "type" : 1,
        "invalid" : false,
        "root" : false,
        "networkPath" : false,
        "device" : false,
        "port" : true,
        "view" : false,
        "dsObject" : false
      },
      "displayName" : " [ Fa0/16 ] FastEthernet0/16",
      "objectId" : 18,
      "typeName" : "UnmanagedPort"
    }
  } ],
  "count" : 29
```

```
}
```

objects/*ID*/configManagement PUT Method

Modifies single attribute of the device

Request

An attribute definition:

Name	Description
name	The name of attribute to be changed: ■ NumberOfConfigsToArchive ■ configMonitorRetrievalScript ■ configMonitorPolicyRules ■ ConfigRetrievalEnabled ■ configMonitorTransferMethod ■ SNMPChangeDetectionEnabled ■ configMonitorExcludedDifference ■ cliMethod ■ cliUsername.
values	The list of values for this attribute.

Table 64 objects/*ID*/configManagement PUT Method Request

Response

An updated attribute:

- OK if the method succeeded
- An error description otherwise.

Examples

```
curl -u admin:admin http://entuity_server/api/objects/611?media=json
-X PUT -H "Content-Type: application/json" -d
'{
  "name" : "cliMethod",
  "values" : [ "ssh" ]
}'
"OK"


curl -u admin:admin http://localhost/api/objects/611/
configManagement?media=xml
```

```
-X PUT -H "content-type: application/xml" -d
'<objectAttribute>
  <name>cliMethod</name>
  <value>ssh</value>
</objectAttribute>'
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<statusInfo>
    <message>OK</message>
</statusInfo>
```

portManagement

Allows you to manage and unmanage ports.

Method	Description
PUT	Controls the managed/unmanaged state of a port.

Table 65 portManagement Method

portManagement PUT Method

This method accepts two boolean parameters, which can be equal to either of **yes**, **true**, **y**, **t** or **1**.

If both parameters are present, "reManage" takes over the "unManage".

If none is present, then this method reports an error.

Response

A message describing the current management state of a port.

Examples

This JSON example unmanages a port, where *Device ID* is 2 and *Interface Index* is 18:

```
curl -u admin:admin http://entuity_server/api/portManagement/2/
18?unManage=y&media=json
-X PUT -H "Content-Type: application/json" -d
{
  "message" : "Successfully marking port as unmanaged"
}
```

This XML example unmanages a port, where *Device ID* is 2 and *Interface Index* is 18:

```
curl -u admin:admin http://entuity_server/api/portManagement/2/
18?unManage=y&media=xml
-X PUT -H "Content-Type: application/xml" -d
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<statusInfo>
    <message>Successfully marking port as unmanaged</message>
</statusInfo
```

servers

Allows you to list Entuity servers or add a new server.

Method	Description	Formats
GET	List of Entuity servers.	XML, JSON
POST	Add a new server.	XML, JSON

Table 66 Servers Method Summary

servers GET Method

Request Parameters

None

Response Data Keys

Name	Description
count	Number of servers.
servers	List of servers.
server	Server summary details, with attributes: ■ *name*, DNS name of the host. ■ *id*, Unique server Identifier.

Table 67 Servers Response Data Keys

servers GET Examples

Provides details of the connected server and its remote servers, for example this command queries the server and requests the:

- XML response format:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/servers
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="2">
    <item xsi:type="namedItem" name="entlonppvm01" id="9e2456cd-1e19-
47d9-860b-509af0e8a11a" serverId="9e2456cd-1e19-47d9-860b-
509af0e8a11a" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
```

```
    <item xsi:type="namedItem" name="century" id="d5f11137-be43-4c58-
a547-6f4d68cb4e83" serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
</items>
```

- JSON response format:

```
curl -u admin:admin -H Accept:application/json -X GET http://
entuity_server/api/servers
{
  "items" : [ {
    "serverId" : "9e2456cd-1e19-47d9-860b-509af0e8a11a",
    "id" : "9e2456cd-1e19-47d9-860b-509af0e8a11a",
    "name" : "entlonppvm01"
  }, {
    "serverId" : "d5f11137-be43-4c58-a547-6f4d68cb4e83",
    "id" : "d5f11137-be43-4c58-a547-6f4d68cb4e83",
    "name" : "century"
  } ],
  "count" : 2
}
```

servers/*id*

Methods allow you to list details of, amend the details of or delete the specified server.

Method	Description	Formats
GET	Get server details.	XML, JSON
PUT	Change a server's details.	XML, JSON
DELETE	Remove the required server.	XML, JSON

Table 68 servers/*id* Methods

servers/*id* GET Method

Request Parameters

None

Response Data Keys

Name	Description
centralServer	Is the server a Central Server? True or False.
Name	Name of host running the Entuity server.
included	Does the server provide results in a multi server system? True or False.
licensed	Is the server licensed? True or False.
local	Is this server the one servicing the request? True or False.
role	Servers role.Polling, FlowCollector or ESPServer.
serverId	Unique server identifier.
ssl	Is the server configured to use Secure Socket Layer? True or False.
webPort	Port number of the web server.

Table 69 servers/*id* Response Data Keys

servers/*id* GET Examples

Every Entuity server has its own identifier that you can retrieve by making a servers request. (See *servers GET Method*.)

`servers/id` GET provides details of the connected server and this command requests the:

- XML response format:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/servers/9e2456cd-1e19-47d9-860b-509af0e8a11a

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<entuityServer licensed="true" role="Polling" local="true"
included="true" centralServer="false" ssl="false" webPort="80"
name="century" serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83"/>
```

- JSON response format:

```
curl -u admin:admin -H Accept:application/json -X GET http://
entuity_server/api/servers/9e2456cd-1e19-47d9-860b-509af0e8a11a

{
  "ssl" : false,
  "serverId" : "d5f11137-be43-4c58-a547-6f4d68cb4e83",
  "local" : true,
  "licensed" : true,
  "webPort" : 80,
  "centralServer" : false,
  "name" : "century",
  "included" : true,
  "role" : "Polling"
```

```
}
```

tools

Returns a list of tools available on this server.

Method	Description
GET	Lists all the tools for this server

Table 70 tools Method

tools GET Method

Response

The list of tools, grouped into subgroups, where with every entry having a following format:

Method	Description
id	Tool ID, unique per server.
Name	The name of the tool.

Table 71 tools GET Method

Examples

```
curl -u admin:admin http://localhost/api/tools?media=json

{
  "items" : [ {
    "subgroup" : "Administrator Tools",
    "tools" : [ {
      "id" : 79,
      "name" : "Event Administration"
    }, {
    ...removed for brevity...
    }, {
      "id" : 111,
      "name" : "UD Polling"
    } ]
  }, {
  ...removed for brevity...
  }, {
    "subgroup" : "Tools",
```

```
    "tools" : [ {
      "id" : 15,
      "name" : "Ticker"
    }, {
    ...removed for brevity...
    }, {
      "id" : 112,
      "name" : "Configuration Management"
    } ]
  ...removed for brevity...
  } ],
  "count" : 6
}
```

```
curl -u admin:admin http://localhost/api/tools?media=xml
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="6">
  <item xsi:type="toolsGroup" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
    <subgroup>Administrator Tools</subgroup>
    <tools>
      <tools name="Event Administration" id="79"/>
    ...removed for brevity...
      <tools name="UD Polling" id="111"/>
    </tools>
  </item>
  ...removed for brevity...
  <item xsi:type="toolsGroup" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
    <subgroup>Tools</subgroup>
    <tools>
      <tools name="Ticker" id="15"/>
      ...removed for brevity...
      <tools name="Configuration Management" id="112"/>
    </tools>
  </item>
  ...removed for brevity...
```

```
</items>
```

userGroups

The resource identifies key attributes of the Entuity user groups.

Method	Description
GET	Lists available user groups.
PUT	Creates a new group of users.

Table 72 userGroups Method Summary

userGroups GET Method

The list of user groups returned is restricted for non-administrators to the groups the user is currently a member of.

Response

Response includes a list of user groups together with key attributes.

Name	Description
id	User group id unique to the server.
name	User group name.
serverId	Entuity Server Id on which resource resides.

Table 73 userGroups Response

userGroups GET Examples

Lists user groups on the specified server, for example in the:

- XML response format:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/userGroups
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="3">
    <item xsi:type="namedItem" name="Administrators" id="1"
serverId="921c3f82-bcdf-4fef-a5e8-7d2524928d96" />
    <item xsi:type="namedItem" name="All Users" id="2"
serverId="921c3f82-bcdf-4fef-a5e8-7d2524928d96" />
    <item xsi:type="namedItem" name="Support" id="6"
serverId="921c3f82-bcdf-4fef-a5e8-7d2524928d96" />
</items>
```

- JSON response format:

```
curl -u admin:admin -H Accept:application/json -X GET http://
entuity_server/api/userGroups
{
  "items" : [ {
    "serverId" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
    "id" : "1",
    "name" : "Administrators"
  }, {
    "serverId" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
    "id" : "2",
    "name" : "All Users"
  }, {
    "serverId" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
    "id" : "6",
    "name" : "Support"
  } ],
  "count" : 3
}
```

userGroups POST Method

Creates a new group of users.

Request

Name	Description
Name	Name of the group

Table 74 userGroups POST Method

Response

The list of user groups after an update, as the GET method would return them.

Examples

```
curl -u admin:admin http://localhost/api/userGroups?media=json

-X POST -H "Content-Type: application/json" -d
'{
  "name" : "Group A"
}'
{
```

```
  "items" : [ {
    "serverId" : "9558b377-67fc-417d-8d8a-87411e50f84c",
    "id" : "1",
    "name" : "Administrators"
  }, {
    "serverId" : "9558b377-67fc-417d-8d8a-87411e50f84c",
    "id" : "2",
    "name" : "All Users"
  }, {
    "serverId" : "9558b377-67fc-417d-8d8a-87411e50f84c",
    "id" : "6",
    "name" : "Group A"
  } ],
  "count" : 3
}
```

```
curl -u admin:admin http://localhost/api/userGroups?media=xml

-X POST -H "Content-Type: application/xml" -d
'<userGroupInfo audit="false">
  <name>Group A</name>
</userGroupInfo>'
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="3">
  <item xsi:type="namedItem" name="Administrators" id="1"
serverId="3deb141f-1f33-43ac-ad59-18ea64a28b2d" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
  <item xsi:type="namedItem" name="All Users" id="2"
serverId="3deb141f-1f33-43ac-ad59-18ea64a28b2d" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
  <item xsi:type="namedItem" name="Group A" id="6" serverId="3deb141f-
1f33-43ac-ad59-18ea64a28b2d" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"/>
</items>
```

userGroups/*ID*

```
URL: usersGroups/{groupID}
```

This resource represents a set of operations on a particular group.

Method	Description
DELETE	Deletes the group

Table 75 userGroups/ID Methods

userGroups/*ID* DELETE Method

Deletes the specified user group.

Request

Name	Description
Name	Name of the group

Table 76 userGroups DELETE Method

Response

The list of users after an update, as GET method would return them.

Examples

```
curl -u admin:admin http://localhost/api/userGroups?media=json -X
DELETE

{
  "items" : [ {
    "serverId" : "9558b377-67fc-417d-8d8a-87411e50f84c",
    "id" : "1",
    "name" : "Administrators"
  }, {
    "serverId" : "9558b377-67fc-417d-8d8a-87411e50f84c",
    "id" : "2",
    "name" : "All Users"
  } ],
  "count" : 2
}

curl -u admin:admin http://localhost/api/userGroups?media=xml -X
DELETE

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="2">
```

```
  <item xsi:type="namedItem" name="Administrators" id="1"
serverId="3deb141f-1f33-43ac-ad59-18ea64a28b2d" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
  <item xsi:type="namedItem" name="All Users" id="2"
serverId="3deb141f-1f33-43ac-ad59-18ea64a28b2d" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
</items>
```

usersGroups/*ID*/tools

URL: usersGroups/{groupID}/tools

This resource views and modifies the set of tools per given group.

Method	Description
GET	Lists tools that this group has permissions to
PUT	Modifies the list of tools that this group has permissions to use.

Table 77 usersGroups/*ID*/tools Methods

usersGroups/*ID*/tools GET Method

The list of tools that this group has a permission to access.

Response

List of items, each one according to the following format:

Name	Description
id	Tool ID, unique per server
name	Tool name
subgroup	The name of subgroup that given tool is a part of

Table 78 usersGroups/*ID*/tools GET Method

Examples

```
curl -u admin:admin http://localhost/api/userGroups/6/tools?media=json

{
  "items" : [ ],
  "count" : 0
}

curl -u admin:admin http://localhost/api/userGroups/6/tools?media=xml

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<items count="0"/>
```

usersGroups/*ID*/tools PUT Method details

Modifies the list of tools that this group has permissions to

Request

The list of tool IDs.

Response

The updated list of group permissions, as if GET method would return them.

Examples

```
curl -u admin:admin http://localhost/api/userGroups/6?media=json

-X PUT -H "Content-Type: application/json" -d

'{
  "tools" : [ {
    "id" : 15
  }, {
    "id" : 89
  } ]
}'
{
  "items" : [ {
    "id" : 15,
    "toolName" : "Ticker",
    "groupName" : "Tools"
  }, {
    "id" : 89,
    "toolName" : "Data Export",
    "groupName" : "Administrator Tools"
  } ],
  "count" : 2
}
```

users

The resource identifies key attributes of the Entuity users.

Method	Description
GET	Lists available users.
POST	Creates a new user.

Table 79 Users Method Summary

Users GET Method

The list of users returned is restricted for non-administrators: only the user object corresponding to the current user is returned.

Response

Response includes a list of users.

Attribute	Description
id	User group id unique to the server.
Name	User group name.
serverId	Entuity Server Id on which resource resides.

Table 80 userGroups Response

Users GET Examples

Provides details of the specified users, for example this command queries the server and requests the:

- XML response format:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/users
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="2">
    <item xsi:type="namedItem" name="admin" id="3" serverId="d5f11137-
be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"/>
    <item xsi:type="namedItem" name="user" id="4" serverId="d5f11137-
be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"/>
</items>
```

- JSON response format:

```
curl -u admin:admin -H Accept:application/json -X GET http://
entuity_server/api/users
{
  "items" : [ {
    "serverId" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
```

```
    "id" : "7",
    "name" : "John"
  }, {
    "serverId" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
    "id" : "3",
    "name" : "admin"
  }, {
    "serverId" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
    "id" : "4",
    "name" : "user"
  } ],
  "count" : 3
}
```

Users POST Method

Creates a new user.

Request

Name	Description
Name	Name of the user
password	User password

Table 81

Response

The list of users after an update, as GET method would return them.

Example

```
curl -u admin:admin http://localhost/api/users?media=json

-X POST -H "Content-Type: application/json" -d
'{
  "name" : "John",
  "password" : "little"
}'
{
  "items" : [ {
    "serverId" : "821c3e87-bcdf-4fef-a5e8-7d2524928d96",
    "id" : "7",
```

```
    "name" : "John"
  }, {
    "serverId" : "821c3e87-bcdf-4fef-a5e8-7d2524928d96",
    "id" : "3",
    "name" : "admin"
  } ],
  "count" : 2
}

curl -u admin:admin http://localhost/api/users?media=xml

-X POST -H "Content-Type: application/xml" -d
'<userPasswordWrapper>
    <name>John</name>
    <password>little</password>
</userPasswordWrapper>'
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="2">
  <item xsi:type="namedItem" name="admin" id="3" serverId="3deb141f-
1f33-43ac-ad59-18ea64a28b2d" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"/>
  <item xsi:type="namedItem" name="John" id="4" serverId="3deb141f-
1f33-43ac-ad59-18ea64a28b2d" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"/>
</items>
```

users/*ID*

This resource implements operations acting on a single user.

Method	Description
GET	Presents detailed user info.
PUT	Modifies parameters of a user.
DELETE	Deletes user.

Table 82 Users/ID Methods

users/*ID* GET Method

Returns a detailed information about the given user.

Response

User info structure with the following format:

Name	Description
id	User ID, unique per server
Name	The name of the user
lockAttempts	The number of failed login attempts before locking the account
expiryDays	The number of days after which the account expires
timeoutMinutes	The number of minutes after which session becomes inactive
pwChangeDays	Days to password change
forcePWChange	Whether the user needs to change their password
groups	The list of groups that this user is a member of
locked	The "locked" status of the user
admin	Whether the user is admin
expired	Has the user account expired

Table 83 users/ID GET Method

Examples

```
curl -u admin:admin http://localhost/api/users/7?media=json
{
  "id" : 7,
  "name" : "John",
  "lockAttempts" : -3,
  "expiryDays" : -14,
  "pwChangeDays" : -14,
  "forcePWChange" : false,
  "groups" : [ "All Users" ],
  "locked" : false,
  "expired" : false,
  "admin" : false
}
```

```
curl -u admin:admin http://localhost/api/users/7?media=xml
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<userAccountInfo expired="false" locked="false" admin="false" id="7">
    <name>John</name>
    <forcePWChange>false</forcePWChange>
```

```
    <lockAttempts>-3</lockAttempts>
    <expiryDays>-14</expiryDays>
    <pwChangeDays>-14</pwChangeDays>
    <groups>All Users</groups>
</userAccountInfo>
```

users/*ID* PUT Method

Modifies parameters of a user.

Request

Name	Description
lockAttempts	The number of failed login attempts before locking the account
expiryDays	The number of days after which the account expires
timeoutMinutes	The number of minutes after which session becomes inactive
pwChangeDays	Days to password change
forcePWChange	Whether the user needs to change their password
groups	The list of groups that this user is a member of

Table 84 users/ID PUT Method

Response

Detailed information about the user after changes, as GET method would return it.

Examples

```
curl -u admin:admin http://localhost/api/users/7?media=json -X PUT -H
"Content-Type: application/json" -d

'{
  "name" : "John",
  "lockAttempts" : -3,
  "expiryDays" : 14,
  "forcePWChange" : true,
  "groups" : [ "Group A", "Group B" ]
}'
{
  "id" : 7,
  "name" : "John",
  "lockAttempts" : -3,
  "expiryDays" : 14,
```

```
  "pwChangeDays" : -14,
  "forcePWChange" : false,
  "groups" : [ "All Users", "Group A", "Group B" ],
  "locked" : false,
  "admin" : false,
  "expired" : true
}
```

users/*ID* DELETE Method

Deletes user account.

Response

The current list of users, as the GET method of URL "users" (without the ID part) would return it.

Examples

```
curl -u admin:admin http://localhost/api/users/7?media=json -X DELETE

{
  "items" : [ {
    "serverId" : "d56f745a-4487-4d40-9fad-9f711bb2cfd4",
    "id" : "3",
    "name" : "admin"
  } ],
  "count" : 1
}

curl -u admin:admin http://localhost/api/users/7?media=xml -X DELETE

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="1">
  <item xsi:type="namedItem" name="admin" id="3" serverId="d56f745a-
4487-4d40-9fad-9f711bb2cfd4" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"/>
</items>
```

version

Allows you to identify the Entuity RESTful API version.

Method	Description
GET	Lists available views.

Table 85 version Method Summary

version GET Method

Response

Response identifies the current version of the Entuity RESTful API.

Attribute	Description
version	Version of the Entuity RESTful API name.

Table 86 version Response

version GET Example

Returns the version number of the Entuity RESTful API, for example this command queries the server and requests the:

- XML response format:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/version
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<versionInfo version="v1"/>
```

- JSON response format:

```
C:\>curl -u admin:admin -H Accept:application/json -X GET http://
entuity_server/api/version
{
  "version" : "v1"
}
```

views

Show views or add a view.

Method	Description
GET	Lists available views.
POST	Create new view.

Table 87 views Method Summary

views GET Method

Response

Response includes a list of views and their key attributes.

Attribute	Description
displayName	View name.
id	View id unique to the server.
path	View path with forward slash as a sub-view separator.
serverId	Entuity Server Id on which resource resides.

Table 88 views Response

views GET Examples

Provides details of the views on the specified server, for example this command queries the server for views and requests the:

- XML response format:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/views
<items count="11">
    <item xsi:type="viewPathItem" path="All Objects" displayName="All
Objects" id="1" serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <item xsi:type="viewPathItem" path="admin::My Network"
displayName="My Network (admin)" id="2" serverId="d5f11137-be43-4c58-
a547-6f4d68cb4e83" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"/>
.
. (This is an edited extract of the returned items)
.
    <item xsi:type="viewPathItem" path="Europe" displayName="Europe"
id="10" serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <item xsi:type="viewPathItem" path="Europe/England"
displayName="Europe/England" id="12" serverId="d5f11137-be43-4c58-
a547-6f4d68cb4e83" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"/>
</items>
```

- JSON response format:

```
curl -u admin:admin -H Accept:application/json -X GET http://
entuity_server/api/views
{
```

```
  "items" : [ {
    "serverId" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
    "id" : "1",
    "displayName" : "All Objects",
    "path" : "All Objects"
  }, {
    "serverId" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
    "id" : "2",
    "displayName" : "My Network (admin)",
    "path" : "admin::My Network"
  } ],
  "count" : 2
}
```


Views POST Method

Creates a new view.

Request

Request is an object which may contain a subset of the following properties.

Name	Description
accessGroups	The array of access objects, specifying view access permissions.
baseViewAggregation	A way to aggregate base views: One of NONE (default), UNION or INTERSECTION.
baseViewPaths	An array of base views.
domainFilterName	The name of the domain filter to use.
eventFilterName	The name of the event filter to use.
incidentFilterName	The name of the incident filter to use.
Name	The name of the view to create.
owner	User name who will be an owner of the view. Defaults to the user making a call. Only administrators may specify a user different to themselves.
parentViewPath	Forward-slash separated path of the parent view. Leave out to create a top-level view.

Table 89 views Request

Response

After creating a view Entuity displays a summary of all views on the server, this includes the *id* of your new view. You can use the *id* to display, and therefore check, your view configuration, for example:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/views/15
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<viewPathDetails owner="admin" incidentFilterName="All Incidents"
eventFilterName="All Events" manuallyPopulated="false"
domainFilterName="All Objects" baseViewAggregation="UNION"
displayName="London Core" path="London Core" id="15" serverId="e003cbb
4-5d98-4711-9bac-1cc8ffbf8cf6">
    <baseViewPath>Europe/England/London Admin</baseViewPath>
    <baseViewPath>Key Devices</baseViewPath>
    <accessGroup editable="true" userGroupName="Administrators"/>
</viewPathDetails>
```

Views POST Examples

You can create a view and sub views, for example:

- This command creates a new view at the server root. It uses the XML request format and therefore you must specify the `viewPathCreateRequest` element:

```
curl -u admin:admin -H Content-Type:application/xml -H
Accept:application/xml -X POST http://entuity_server/api/views -d
"<viewPathCreateRequest name='API View' />"
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="12">
    <item xsi:type="viewPathItem" path="All Objects" displayName="All
Objects" id="1" serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
.
. (This edited extract only includes the first and last items)
.
    <item xsi:type="viewPathItem" path="API View" displayName="API
View" id="16" serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
</items>
```

- This command creates a new view, London Admin, that is the child view of England, which itself is the child view of Europe. It also specifies domain, event and incident filters and uses the XML request format and therefore you must specify the `viewPathCreateRequest` element:

```
curl -u admin:admin -H Content-Type:application/xml -H
Accept:application/xml -X POST http://entuity_server/api/views -d
"<viewPathCreateRequest name='London Admin' parentViewPath='Europe/
England' domainFilterName='Routers' eventFilterName='Entuity System
Events' incidentFilterName='Entuity System Incidents'/>"
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="14">
    <item xsi:type="viewPathItem" path="All Objects" displayName="All
Objects" id="1" serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
.
. (This edited extract only includes the first and last items)
.
    <item xsi:type="viewPathItem" path="Europe/England/London Admin"
displayName="Europe/England/London Admin" id="18" serverId="d5f11137-
be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"/>
</items>
```

- This command creates a new view, London Core, that contains the intersection of two views. It uses the XML request format and therefore you must specify the `viewPathCreateRequest` and `baseViewPath` elements:

```
curl -u admin:admin -H Content-Type:application/xml -H
Accept:application/xml -X POST http://entuity_server/api/views -d
"<viewPathCreateRequest name='London Core' parentViewPath='Europe/
England' baseViewAggregation='INTERSECTION'><accessGroup
editable='true' userGroupName='Europe'/> <baseViewPath>Key Devices</
baseViewPath><baseViewPath>Europe/England/London Admin</
baseViewPath></viewPathCreateRequest>"
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="10">
    <item xsi:type="viewPathItem" path="All Objects" displayName="All
Objects" id="1" serverId="e003cbb4-5d98-4711-9bac-1cc8ffbf8cf6"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
.
. (This is an edited extract of the returned items)
.
    <item xsi:type="viewPathItem" path="Europe" displayName="Europe"
id="4" serverId="e003cbb4-5d98-4711-9bac-1cc8ffbf8cf6"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <item xsi:type="viewPathItem" path="Europe/England"
displayName="Europe/England" id="5" serverId="e003cbb4-5d98-4711-9bac-
1cc8ffbf8cf6" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <item xsi:type="viewPathItem" path="Europe/England/London Admin"
displayName="Europe/England/London Admin" id="6" serverId="e003cbb4-
5d98-4711-9bac-1cc8ffbf8cf6" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"/>
```

```
    <item xsi:type="viewPathItem" path="Key Devices" displayName="Key
Devices" id="7" serverId="e003cbb4-5d98-4711-9bac-1cc8ffbf8cf6"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
    <item xsi:type="viewPathItem" path="Europe/England/London Core"
displayName="Europe/England/London Core" id="13" serverId="e003cbb4-
5d98-4711-9bac-1cc8ffbf8cf6" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"/>
</items>
```

You can also create views by specifying their configuration through a data file, for example this calls a JSON data file:

```
curl -u admin:admin -H "Content-Type: application/json" -X POST --data
@createView.json http://entuity_server/api/views
```

These are JSON format examples which could be included to the data file (`createView.json`):

- Creates a top-level view, called Simple View with all other view attributes set to their default:

```
{
  "name" : "Simple View"
}
```

- Creates a top-level view with the filter information also specified:

```
{
  "name" : "MyView",
  "domainFilterName" : "All Objects",
  "eventFilterName" : "All Events",
  "incidentFilterName" : "All Incidents"
}
```

- Creates a new view, London, that is the child view of England, which itself is the child view of Europe:

```
{
  "name" : "London",
  "parentViewPath" : "Europe/England"
}
```

- Creates a view which unions contents from the two base views:

```
{
  "name" : "Eastern",
  "baseViewAggregation" : "UNION",
  "baseViewPaths" : [ "NewYork", "Boston" ]
```

```
}
```

- Creates a view which contains the key devices in London by intersecting the content from its two base views, Key_Devices and London:

```
{
  "name" : "Intersection View",
  "baseViewAggregation" : "INTERSECTION",
  "baseViewPaths" : [ "Key_Devices", "Europe/England/London" ]
}
```

- Creates a view with a specific owner and access rights:

```
{
  "name" : "John's View",
  "owner" : "John",
  "accessGroups" : [ {
    "userGroupName" : "Support",
    "editable" : true
  }, {
    "userGroupName" : "All Users",
    "editable" : false
  } ]
}
```

views/*id*

With this resource you can inspect, update or delete a specified view.

Method	Description
GET	Inspect a view
PUT	Update a view
DELETE	Delete a view

Table 90 views/*id* Method

views/*id* GET Method

Response

Request is an object which may contain a subset of the following properties.

Name	Description
accessGroups	Group access permissions.
baseViewAggregation	One of NONE, UNION, INTERSECTION.
baseViewPaths	The array of base view paths.
displayName	User-friendly view name.
domainFilterName	Domain filter name.
eventFilterName	Event filter name.
id	view id unique to the server.
implicitAccessGroups	Groups having implicit access.
implicitAccessUsers	Users having implicit access.
incidentFilterName	Incident filter name.
manuallyPopulated	Specifies if contents of the view is manually populated (true) or automatically (false).
owner	User owning a view.
path	View path.
serverId	Entuity Server Id on which resource resides.

Table 91 views/*id* Response

views/*id* GET Examples

You use the view identifier to apply your GET request to the required view. You can retrieve the identifier by making a view request. (See *views GET Method*.)

`views/id` GET provides details of the specified view, for example this command queries view:

- 7 which is manually populated and requests the XML response format:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/views/7
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<viewPathDetails owner="admin" incidentFilterName="All Incidents"
eventFilterName="All Events" manuallyPopulated="true"
domainFilterName="All Objects" baseViewAggregation="NONE"
displayName="Business/QA" path="Business/QA" id="7"
serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83">
   <accessGroup editable="true" userGroupName="Administrators"/>
</viewPathDetails>
```

- 15 which is:
  - Based on the intersection of two base views. Views are identified through their full view path, for example **Europe**/**England**/**London Admin** indicates **London Admin** is the child view of **England** which is a sub-view of **Europe**.

- Editable by the user group Europe user group with edit permission (as well as the Administrators user group).

The command also requests the XML response format:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/views/15
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<viewPathDetails owner="admin" incidentFilterName="All Incidents"
eventFilterName="All Events" manuallyPopulated="false"
domainFilterName="All Objects" baseViewAggregation="INTERSECTION"
displayName="London Core" path="London Core" id="15"
serverId="e003cbb4-5d98-4711-9bac-1cc8ffbf8cf6">
   <baseViewPath>Europe/England/London Admin</baseViewPath>
   <baseViewPath>Key Devices</baseViewPath>
   <accessGroup editable="true" userGroupName="Administrators"/>
   <accessGroup editable="true" userGroupName="Europe"/>
</viewPathDetails>
```

- 48 and requests the JSON response format:

```
curl -u admin:admin -H Accept:application/json -X GET http://
entuity_server/api/views/48
{
  "serverId" : "921c3f82-bcdf-4fef-a5e8-7d2524928d96",
  "id" : "48",
  "path" : "Simple View",
  "displayName" : "Simple View",
  "baseViewAggregation" : "NONE",
  "baseViewPaths" : [ ],
  "domainFilterName" : "All Objects",
  "manuallyPopulated" : true,
  "eventFilterName" : "All Events",
  "incidentFilterName" : "All Incidents",
  "owner" : "admin",
  "accessGroups" : [ {
    "userGroupName" : "Administrators",
    "editable" : true
  } ],
  "implicitAccessGroups" : [ ],
  "implicitAccessUsers" : [ ]
}
```

views/*id* PUT Method

This resource updates the specified view.

Request

Request has the same structure as the POST request of views resource for adding a new view, except the `parentViewPath` property must be absent:. You need to specify only properties you want to be changed.

Response

Response has the same structure as the GET response: shows view details after the update.

views/*id* Examples

You use the view identifier to apply your PUT request to the required view. You can retrieve the identifier by making a view request. (See *views GET Method*.)

`views/id` PUT allows you to update the specified view, for example this command instructs Entuity to update view:

- 15 with the new name London Key, using the XML format:

```
curl -u admin:admin -H Content-Type:application/xml -H
Accept:application/xml -X PUT http://entuity_server/api/views/15 -d
"<viewPathEditRequest name='London Key'> </viewPathEditRequest>"
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<viewPathDetails owner="admin" incidentFilterName="All Incidents"
eventFilterName="All Events" manuallyPopulated="false"
domainFilterName="All Objects" baseViewAggregation="UNION"
displayName="London Key" path="London Key" id="15" serverId="e003cbb4-
5d98-4711-9bac-1cc8ffbf8cf6">
    <baseViewPath>Europe/England/London Admin</baseViewPath>
    <baseViewPath>Key Devices</baseViewPath>
    <accessGroup editable="true" userGroupName="Administrators"/>
    <accessGroup editable="true" userGroupName="Europe"/>
</viewPathDetails>
```

- 48 according to the configuration in the JSON data file:

```
curl -u admin:admin -H "Content-Type:application/json" -X PUT --data
@updateView.json http://entuity_server/api/views/48
```

views/*id* DELETE Method

This resource deletes the specified view.

Request

No request expected.

Response

Entuity does not prompt you to confirm the deletion, the view is immediately deleted. Entuity also does not confirm successful view deletion.

views/*id* DELETE Examples

You use the view identifier to apply your DELETE request to the required view. You can retrieve the identifier by making a view request. (See *views GET Method*.)

`views/id DELETE` allows you to delete the specified view and any child views, for example this command instructs Entuity to delete view 13:

```
curl -u admin:admin -X DELETE http://entuity_server/api/views/13
```

views/*id*/objects

Inspect, add or delete managed objects within the specified view.

Method	Description
GET	List view items.
PUT	Add items to a view.
DELETE	Remove items from a view.

Table 92 views/*id*/objects Method Summary

views/*id*/objects GET Method

Method returns objects contained in a view.

Request

This resource identifies managed objects within the specified view.

Name	Description
indirect	Set to **include** to return objects from the view and its sub-views. By default, objects from sub-views are not included.

Table 93 views/*id*/objects Request

Response

List each item within the specified view.

Name	Description
displayName	User-friendly name of the item.
id	Item id unique to the server.
serverId	Entuity Server Id on which resource resides.

Table 94 views/*id*/objects Response

Name	Description
typeDisplayName	User-friendly name of the item type.
typeName	Name of the item type.

Table 94 views/*id*/objects Response

views/*id*/objects GET Examples

You use the view identifier to apply your GET request to the required view. You can retrieve the identifier by making a view request. (See *views GET Method*.)

`views/id/objects GET` provides details of the managed objects in the specified view, for example this command queries view:

- 23 and requests the XML response format:

```
curl -u admin:admin -H Accept:application/xml -X GET http://
entuity_server/api/views/23/objects
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="2">
    <item xsi:type="viewContentItem" displayName="1"
typeDisplayName="Vlan" typeName="Vlan" id="1070" serverId="d5f11137-
be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"/>
    <item xsi:type="viewContentItem" displayName="bsw1"
typeDisplayName="Switch Device" typeName="SwitchDevice" id="3515"
serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
</items>
```

- 23 and its sub-views and requests the XML response format:

```
curl -u admin:admin -H Accept:application/xml -X GET "http://
entuity_server/api/views/23/objects?indirect=include"
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="4">
    <item xsi:type="viewContentItem" displayName="1"
typeDisplayName="Vlan" typeName="Vlan" id="1070" serverId="d5f11137-
be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance"/>
    <item xsi:type="viewContentItem" displayName="bsw1"
typeDisplayName="Switch Device" typeName="SwitchDevice" id="3515"
serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
    <item xsi:type="viewContentItem" displayName="APCR1"
typeDisplayName="Device" typeName="DeviceEx" id="3603"
serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
```

```
    <item xsi:type="viewContentItem" displayName="apcr4"
typeDisplayName="Device" typeName="DeviceEx" id="3707"
serverId="d5f11137-be43-4c58-a547-6f4d68cb4e83" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
</items>
```

- 36 and requests the JSON response format:

```
curl -u admin:admin -H Accept:application/json -X GET http://
entuity_server/api/views/36/objects
{
  "items" : [ {
    "serverId" : "9e2456cd-1e19-47d9-860b-509af0e8a11a",
    "id" : 815,
    "typeName" : "DeviceEx",
    "typeDisplayName" : "Device",
    "displayName" : "apcr1"
  } ],
  "count" : 1
}
```

views/*id*/objects PUT Method

Request

Following query parameters can be specified

Name	Description
id	Item id. This parameter can appear multiple times.

Table 95 views/*id*/objects

Response

Response is the same as from a GET method after items have been added.

views/*id*/objects PUT Examples

You use the view identifier to apply your PUT request to the required view. You can retrieve the identifier by making a view request. (See *views GET Method*.)

When putting a device into a view you should use the device's StormWorks identifier (*dsObjectId*). When you make a general GET request (see *inventory GET Method*) the returned inventory includes the device identifier which you can then use to retrieve the StormWorks identifier. (See *inventory/id GET Examples*.)

The inventory device identifier is different to the device's StormWorks identifier (*dsObjectId*). The device's StormWorks identifier is the device identifier used with the views resource.

`views/`*`id`*`/objects` `PUT` adds the managed objects to the specified view, for example this command:

- Adds two items (1030 and 3482) to a previously empty view (211) using the XML format:

```
curl -u admin:admin -H Accept:application/xml -X PUT "http://
entuity_server/api/views/211/objects?id=1030&id=3482"

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="2">
    <item xsi:type="viewContentItem" displayName="buildervm"
typeDisplayName="Managed Host" typeName="ManagedHost" id="1030"
serverId="9e2456cd-1e19-47d9-860b-509af0e8a11a" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
    <item xsi:type="viewContentItem" displayName="jupiter"
typeDisplayName="Managed Host" typeName="ManagedHost" id="3482"
serverId="9e2456cd-1e19-47d9-860b-509af0e8a11a" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
</items>
```

- Adds two items (1030 and 3482) to a previously empty view (211) using the JSON format:

```
curl -u admin:admin -H Accept:application/json -X PUT "http://
entuity_server/api/views/211/objects?id=1030&id=3482"

{
  "items" : [ {
    "serverId" : "9e2456cd-1e19-47d9-860b-509af0e8a11a",
    "id" : 1030,
    "typeName" : "ManagedHost",
    "typeDisplayName" : "Managed Host",
    "displayName" : "buildervm"
  }, {
    "serverId" : "9e2456cd-1e19-47d9-860b-509af0e8a11a",
    "id" : 3482,
    "typeName" : "ManagedHost",
    "typeDisplayName" : "Managed Host",
    "displayName" : "jupiter"
  } ],
  "count" : 2
```

```
}
```

views/*id*/objects DELETE Method

Request

Query parameters specify objects to delete from the specified view.

Name	Description
id	Item id. This parameter can appear multiple times.
include	Can have a special value **all** to empty a view completely. If specified any *id* parameters are ignored.

Table 96 views/*id*/objects Delete Request

Response

Entuity does not prompt you to confirm the deletion, the objects are immediately removed from the view. Entuity does not confirm successful deletion but it does display the contents of the view for you to verify the success of the command. This response is the same as from the GET method.

views/*id*/objects DELETE Examples

You use the view identifier to apply your DELETE request to the required view. You can retrieve the identifier by making a view request. (See *views GET Method*.)

When deleting a device from a view you should use the device's StormWorks identifier (*dsObjectId*). When you make an inventory GET request (see *inventory GET Method*) the returned inventory includes the device identifier which you can then use to retrieve the StormWorks identifier. (See *inventory/id GET Examples*.)

The inventory device identifier is different to the device's StormWorks identifier (*dsObjectId*). The device's StormWorks identifier is the device identifier used with the views resource.

`views/id/objects DELETE` removes the managed objects from the specified view, for example this command:

- Removes two devices (1030 and 3482) from a view (211), leaving the view empty requesting the XML response format:

  ```
  curl -u admin:admin -H Accept:application/xml -X DELETE "http://
  entuity_server/api/views/211/objects?id=1030&id=3482"

  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <items count="0"/>
  ```

- Removes two devices (1030 and 3482) from a view (211), leaving the view empty using the JSON format:

  ```
  curl -u admin:admin -H Accept:application/json -X DELETE "http://
  entuity_server/api/views/211/objects?id=1030&id=3482"
  ```

```
{
  "items" : [ ],
  "count" : 0
}
```

- Removes all items from the current view (23) (it does not remove the content from child views) using the XML format:

```
curl -u admin:admin -H Accept:application/xml -X DELETE "http://
entuity_server/api/views/23/objects?include=all"

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="0"/>
```

- Remove all items from a view (211) using the JSON format:

```
curl -u admin:admin -H Accept:application/json -X DELETE "http://
entuity_server/api/views/211/objects?include=all"

{
  "items" : [ ],
  "count" : 0
}
```

Zones

Resource for displaying a configured zones summary and for defining new zones.

Zones Methods

Method	Description
GET	Lists defined zones
POST	Creates a new zone

Table 97 Zones Methods

Zones GET Method

The list of users returned is restricted for non-administrators: only the user object corresponding to the current user is returned.

Response

Response includes a list of users. Each user has following attributes.

Name	Description
id	Zone ID, unique to the server
name	Zone name
serverId	Entuity Server ID on which resource resides

Table 98 Zones Methods Get

Examples

```
curl -u admin:admin http://localhost/api/zones?media=json


{
  "items" : [ {
    "serverId" : "9558b377-67fc-417d-8d8a-87411e50f84c",
    "id" : "3",
    "name" : "Zone A"
  } ],
  "count" : 1
}

curl -u admin:admin http://localhost/api/zones?media=xml


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="1">
    <item xsi:type="namedItem" name="Zone A" id="3"
serverId="9558b377-67fc-417d-8d8a-87411e50f84c" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
</items>
```

Zones POST Method

Creates a new zone.

Request

Name	Description
id	Zone ID, unique per server.
name	The name of the zone.
flags	Zone flags.

Table 99 Zones POST Method Request

Name	Description
description	The description of the zone.
v4interface	IPv4 interface for this zone.
v6interface	IPv6 interface for this zone.
domainSuffx	Domain suffix for this zone.
proxy	
devicePrefix	
hostFile	The host file.
dnsServer	A list of DNS servers.

Table 99 Zones POST Method Request

Response

The list of users after an update, as GET method would return them.

Zones Example

```
curl -u admin:admin http://localhost/api/zones?media=json -X POST -H
"Content-Type: application/json" -d

'{
  "name" : "Zone B",
  "flags" : 0,
  "description" : "",
  "v4Interface" : "10.44.2.103",
  "v6Interface" : "",
  "domainSuffix" : "",
  "proxy" : "",
  "devicePrefix" : "",
  "hostFile" : "",
  "dnsServers" : [ "" ]
}'
{
  "items" : [ {
    "serverId" : "9558b377-67fc-417d-8d8a-87411e50f84c",
    "id" : "3",
    "name" : "Zone A"
  }, {
    "serverId" : "9558b377-67fc-417d-8d8a-87411e50f84c",
    "id" : "4",
```

```
    "name" : "Zone B"
  } ],
  "count" : 2
}

curl -u admin:admin http://localhost/api/zones?media=xml
-X POST -H "Content-Type: application/xml" -d
'<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<zoneParametersInfo>
    <name>Zone B</name>
    <flags>0</flags>
    <description></description>
    <v4Interface>10.44.2.103</v4Interface>
    <v6Interface></v6Interface>
    <domainSuffix></domainSuffix>
    <proxy></proxy>
    <devicePrefix></devicePrefix>
    <hostFile></hostFile>
    <dnsServers>
        <dnsServers></dnsServers>
    </dnsServers>
</zoneParametersInfo>'
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<items count="2">
    <item xsi:type="namedItem" name="Zone A" id="3"
serverId="9558b377-67fc-417d-8d8a-87411e50f84c" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
    <item xsi:type="namedItem" name="Zone B" id="4"
serverId="9558b377-67fc-417d-8d8a-87411e50f84c" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance"/>
</items>
```

Zone/*zoneID*

This resource implements operations acting on a single user.

Method	Description
GET	Presents detailed zone information.
PUT	Modifies parameters of a zone.
DELETE	Deletes a zone.

Table 100 Zone/zoneID Methods

GET Method details

Returns a detailed information about the given user.

Response

User info structure with the following format:

Name	Description
id	Zone ID, unique per server
name	The name of the zone
flags	Zone flags
description	The description of the zone
v4interface	IPv4 interface for this zone
v6interface	IPv6 interface for this zone
domainSuffx	Domain suffix for this zone
proxy	
devicePrefix	
hostFile	The host file
dnsServer	A list of DNS servers

Table 101 Zone/zoneID Get Method

Examples

```
curl -u admin:admin http://localhost/api/zones/3?media=json

{
  "id" : 3,
  "name" : "Zone A",
  "flags" : 0,
  "description" : "",
  "v4Interface" : "10.44.2.102",
  "v6Interface" : "",
  "domainSuffix" : "",
  "proxy" : "",
```

```
  "devicePrefix" : "",
  "hostFile" : "",
  "dnsServers" : [ "" ]
}
```

```
curl -u admin:admin http://localhost/api/zones/3?media=xml
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<zoneParametersInfo id="3">
    <name>Zone A</name>
    <flags>0</flags>
    <description></description>
    <v4Interface>10.44.2.102</v4Interface>
    <v6Interface></v6Interface>
    <domainSuffix></domainSuffix>
    <proxy></proxy>
    <devicePrefix></devicePrefix>
    <hostFile></hostFile>
    <dnsServers>
        <dnsServers></dnsServers>
    </dnsServers>
</zoneParametersInfo>
```

PUT Method

Modifies parameters of a zone.

Request

Name	Description
id	Zone ID, unique per server
name	The name of the zone
flags	Zone flags
description	The description of the zone
v4interface	IPv4 interface for this zone
v6interface	IPv6 interface for this zone
domainSuffx	Domain suffix for this zone
proxy	
devicePrefix	

Table 102 Zones Put Method

Name	Description
hostFile	The host file
dnsServer	A list of DNS servers

Table 102 Zones Put Method

Response

Detailed information about the user after changes, as GET method would return it.

Examples

```
curl -u admin:admin http://localhost/api/zones/3?media=json

-X PUT -H "Content-Type: application/json" -d
'{
  "name" : "Zone B",
  "flags" : 0,
  "description" : "",
  "v4Interface" : "10.44.2.103",
  "v6Interface" : "fe80:0:0:0:0:5efe:a2c:266",
  "domainSuffix" : "",
  "proxy" : "",
  "devicePrefix" : "",
  "hostFile" : "",
  "dnsServers" : [ "" ]
}'
{
  "id" : 3,
  "name" : "Zone B",
  "flags" : 0,
  "description" : "",
  "v4Interface" : "10.44.2.103",
  "v6Interface" : "fe80:0:0:0:0:5efe:a2c:266",
  "domainSuffix" : "",
  "proxy" : "",
  "devicePrefix" : "",
  "hostFile" : "",
  "dnsServers" : [ "" ]
}
```

```
curl -u admin:admin http://localhost/api/zones/3?media=xml

-X PUT -H "Content-Type: application/xml" -d
'<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<zoneParametersInfo id="3">
    <name>Zone B</name>
    <flags>0</flags>
    <description></description>
    <v4Interface>10.44.2.103</v4Interface>
    <v6Interface>fe80:0:0:0:0:5efe:a2c:266</v6Interface>
    <domainSuffix></domainSuffix>
    <proxy></proxy>
    <devicePrefix></devicePrefix>
    <hostFile></hostFile>
    <dnsServers>
        <server></server>
    </dnsServers>
</zoneParametersInfo>'
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<zoneParametersInfo id="3">
    <name>Zone B</name>
    <flags>0</flags>
    <description></description>
    <v4Interface>10.44.2.103</v4Interface>
    <v6Interface>fe80:0:0:0:0:5efe:a2c:266</v6Interface>
    <domainSuffix></domainSuffix>
    <proxy></proxy>
    <devicePrefix></devicePrefix>
    <hostFile></hostFile>
    <dnsServers>
        <server></server>
    </dnsServers>
</zoneParametersInfo>
```

Zones DELETE Method

Deletes a zone.

Response

The current list of zones, as the GET method would return it.

Examples

```
curl -u admin:admin http://localhost/api/zones/3?media=json -X DELETE
```

```
{
  "message" : "Zone deleted successfully"
}
```

```
curl -u admin:admin http://localhost/api/zones/3?media=xml -X DELETE
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<statusInfo>
    <message>Zone deleted successfully</message>
</statusInfo>
```

Appendix E StormWorks Data Model

Entuity data model is implemented through StormWorks:

- StormWorks Data Dictionary, allows you to interrogate the StormWorks data model.
- StormWorks Data Structures provides an overview of the concepts behind StormWorks.

StormWorks Data Dictionary

The Data Dictionary tool provides an interface to the Entuity data structure, a knowledge of which you will require, for example to correctly configure data export jobs, writing configuration management scripts, variables for reports and when configuring User Defined Polling.

The Data Dictionary reporting tool interrogates the Entuity data model, allowing:

- Viewing of data types, streams and attributes through an index.
- Display of all data entities on one page, enabling search of the HTML.
- Discovery of object associations.

To open the Data Dictionary and interrogate the Entuity data model:

1) Click **Help > Contents**.

   The home page of the help system includes a hyperlink to the Entuity Data Dictionary.

2) In the Get Started column is an Additional Documentation section. Click **Entuity Data Dictionary** hyperlink.

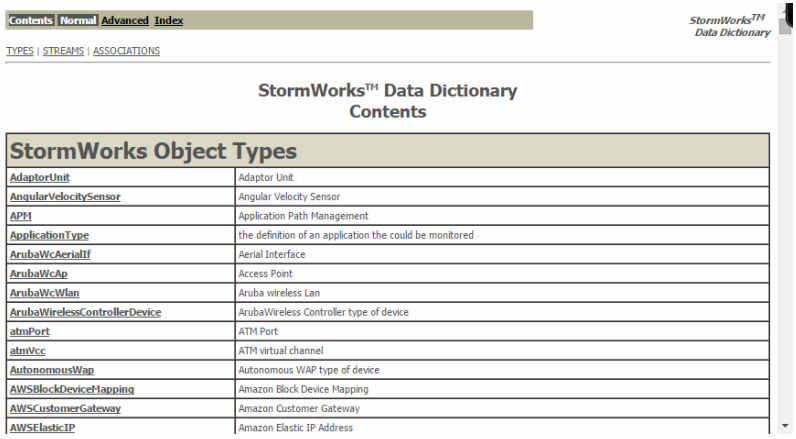

Figure 2 StormWorks Data Dictionary Contents

Changing the Data Access Level

StormWorks data is tagged with a reportable level. In Data Export, and Flex Reports, by default Entuity displays the most useful set of data structures, hiding data not usually useful for users. Entuity Data Dictionary follows the same approach however when required you can remove this filter to display all available data types, streams and attributes.

To change the data access level:

1) From the Entuity Data Dictionary page click **Advanced**.

When you view details of an object, the metadata includes its Reportable Level.

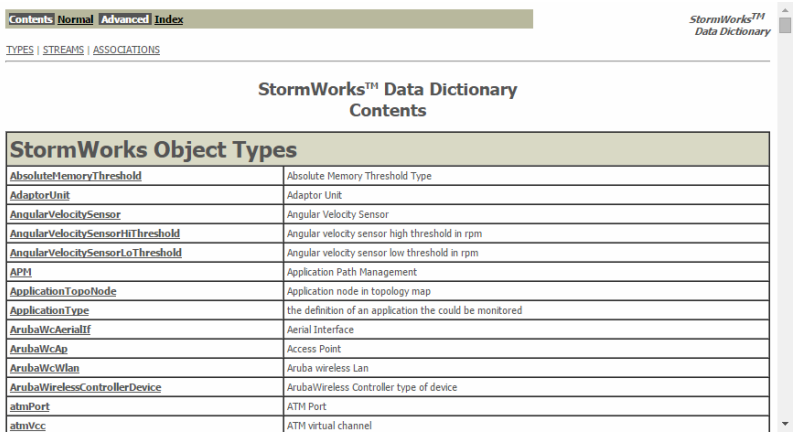

Figure 3 StormWorks Data Dictionary Advanced Contents

Navigating the Data Dictionary

The menu bar includes two navigation lines:

- Content, links to the contents page where data is grouped by streams, types and associations.
- Index, groups data alphabetically, with a sub menu providing alphabet oriented links. There is also an All link, making it possible to view all the data on a page, useful for a page search.

Hyperlinks within the displayed data allow you to move through the data structure.

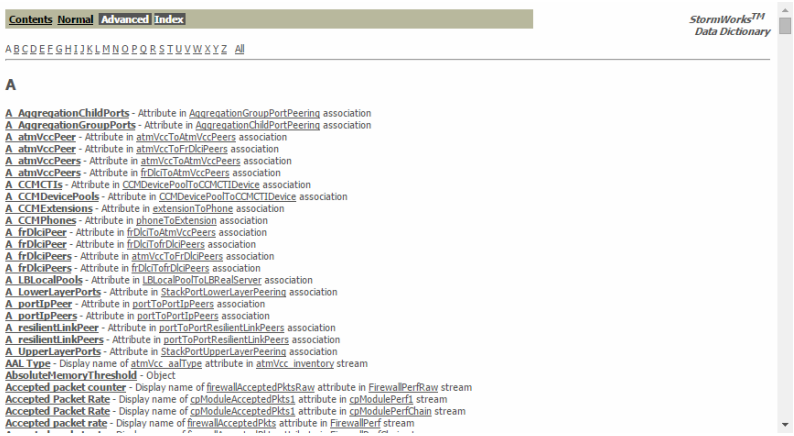

Figure 4 StormWorks Index

Following Data Types

When identifying information to export, a starting point is to find the required StormWorks data type. From there you can find that type's attributes, its streams and its associated types.

To view a data type:

1) From the Entuity Data Dictionary page click **Types**. Data Dictionary displays the list of data types, together with their description in alphabetical order.

2) Locate and select the required data type. The Data Dictionary displays a detailed breakdown of the data type.

   You can navigate through this data type's section, and have the option of using hyperlinks to drill further down to view attribute and association details.

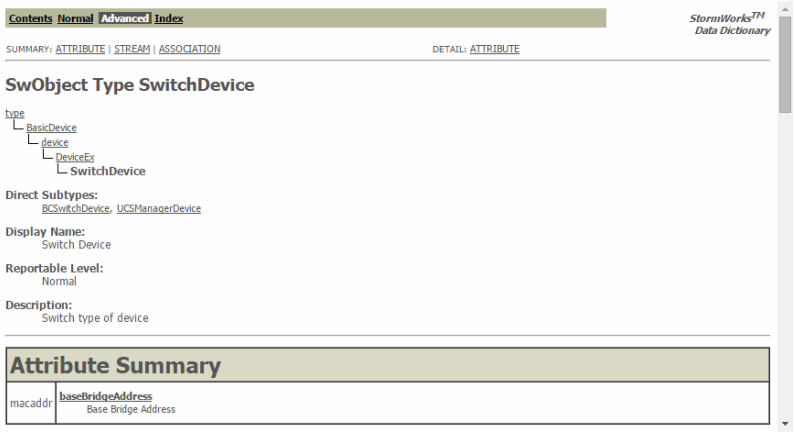

Figure 5 StormWorks Types

Following Associations

Associations define relationships between object types, for example there is an association between a device and its ports. Associations are also used when identifying the topology of your network.

To view a StormWorks association:

1) From the Entuity Data Dictionary page click **Associations**. Data Dictionary displays the list of associations, together with their description in alphabetical order.

2) Locate and select the required association. The Data Dictionary displays a detailed breakdown of the association.

   You can navigate through this association's section, and have the option of using hyperlinks to drill further down to view attribute details.

Figure 6 StormWorks Associations

Following Streams

Streams are the mechanism through which StormWorks maintains historic data, or rather the time series attributes belonging to a stream.

To view a StormWorks stream:

1) From the Entuity Data Dictionary page click **Streams**. Data Dictionary displays the list of streams, together with their description, in alphabetical order.

2) Locate and select the required stream. The Data Dictionary displays a detailed breakdown of the stream.

   You can navigate through this stream's section, and have the option of using hyperlinks to drill further down to view attribute details.

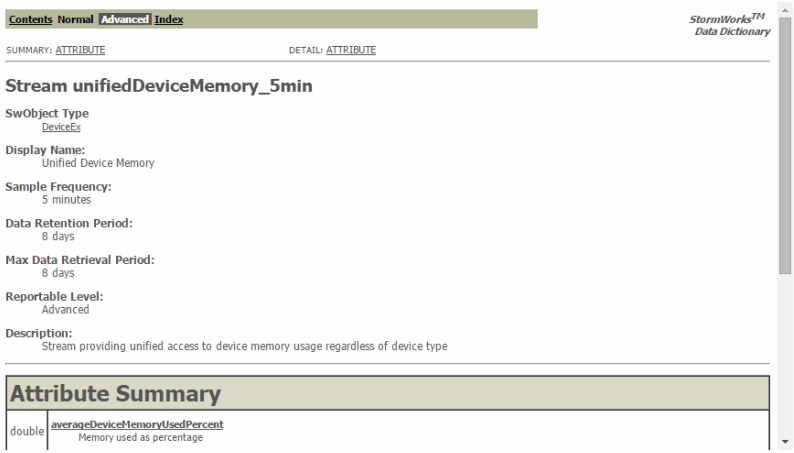

Figure 7 StormWorks Streams

StormWorks Data Structures

The StormWorks Data Dictionary provides a view of the data structure defined through Entuity's configuration files. You can use the dictionary to assist you in specifying data export jobs and writing scripts. An understanding of the data model is also useful when configuring User Defined Polling

For the best results you should understand the different data constructs, how they relate to each other and how Entuity interprets them.

StormWorks Overview

Type is the underlying concept of StormWorks, not just the configuration of StormWorks. Types can be split into:

- **built-in types**, for example floats, strings.
- **Structs** which are defined within the StormWorks configuration files but are not extended from a Type. This means they are not written in the database but are used as a method of combining attributes, which is useful in some calculations.
- **user-defined types**, for example module, device. These are defined within the StormWorks configuration files, and are extended from other types. It is user-defined types that you are most concerned used with when defining data export jobs.

Structure of User Defined Types

Types are defined through StormWorks configuration files. These flat text files are divided into headed sections within which are the definitions for the different types, streams, attributes and other entities that are used to configure StormWorks. Also specified are the relationships between these types, e.g. which type extends from another, which type is associated with another.

Different aspects of constructing a type:

- Associations define relationships between types, this ties related information collected through different types together. For example, the types port and device have a many to one relationship. A device can have many ports, a port can only have one device.
- Attributes define data that can be polled. Attributes can be held:
  - directly against a type. This implies history data for that attribute is not required.
  - against a stream within a type. This is for time-series data.
- Streams define properties of the polling process. A stream can only be connected to one type, although through type inheritance it can appear otherwise. For example, in the inheritance example (see *Type Extension and Inheritance*) the stream *chassisInventory* is defined against the ***device*** type, and is then inherited by the ***SwitchDevice, RouterDevice*** and ***LBDevice*** types.

  StormWorks also has the concept of a virtual stream. A virtual stream does not contain directly polled data, but rather data calculated from polled data. For example, Entuity polls for device uptime, it uses this value to check whether the device has been continuously up since it was last polled, or whether the device has been down. The virtual stream **v deviceUpTime** maintains the poll timestamp, the amount of time the device was up and the amount of time the device was in an unknown state. You can then use this virtual stream when reporting on device uptime.
- Types can be connected to more than one stream, for example ***port*** could have two streams:
  - *portData* that records in-bound and out-bound octets, port speed and duplex information collected every two minutes.
  - *shortUtilization* that records short term utilization (actually based on the octets, speed information and time stamp collected through portData) calculated every two minutes.
- Event State Engine processes the specified poll data from the specified stream.
- Collectors describe a method of how an attribute can be polled. An attribute can have a number of collectors, this allows different methods for attaining the same type of information. Entuity uses the collectors priority level to determine which attribute method to try first, Entuity works through the attribute collectors until a method is successful.
- Transform allows data to be converted from one data type to another, e.g. to change an integer to a string.

Entuity Data Dictionary provides a view into the Entuity data structure to assist in defining data export jobs and scripts. Event State, Collector and Transform details collect and manage data. They are not available for export and are not viewed through the Entuity Data Dictionary.

Type Extension and Inheritance

The majority of StormWorks types are extended from other StormWorks types. They inherit the characteristics of those types, e.g. streams and attributes. This allows type definitions to be built from the general to the specific, which makes for easier type definition and maintenance.

When types are configured and compiled to Entuity different instances of the same object can have different characteristics. For example switches, routers and load balancers are classed as devices, but the device instance of each reflects their different inheritances. All are extended from the same type, ***DeviceEx***. ***DeviceEx*** contains generic device information, e.g. attributes, associations that are common to three device types. These common attributes are inherited; specific device type characteristics are defined against ***LBDevice*** and ***SwitchDevice***. ***RouterDevice*** only inherits attributes and streams and does not have specifically defined against it.

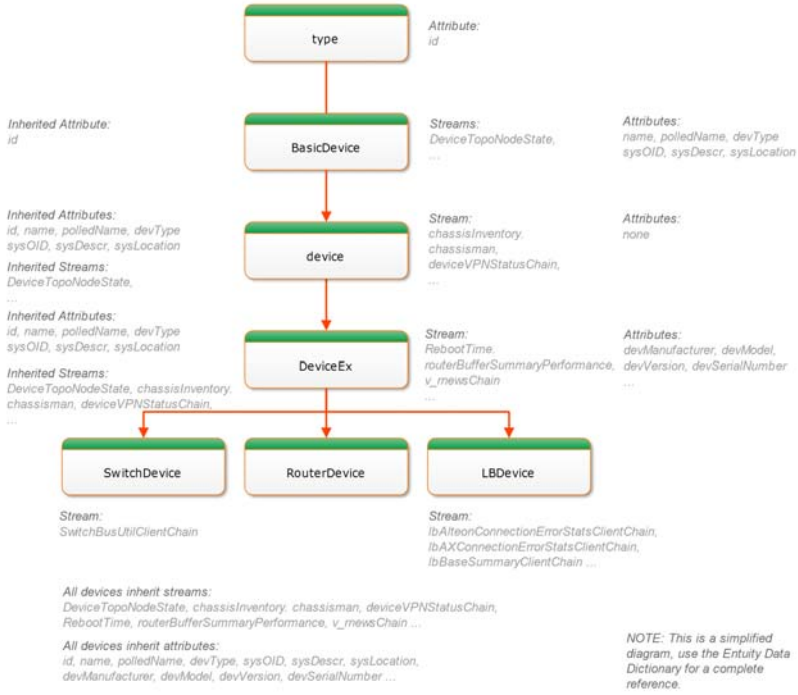

Figure 8 Type Extension and Attribute and Stream Inheritance

Figure 8 Type Extension and Attribute and Stream Inheritance expands the device example:

- ***type*** is the original type on which almost all other types are extended from. The exceptions are types used as structs.
- ***BasicDevice*** which holds characteristics general to devices.
- ***device*** which holds characteristics general to devices.
- ***DeviceEx*** which extends the device definition.
- ***SwitchDevice*** and ***RouterDevice*** have characteristics specific to switches and routers, respectively.

The original ***type*** has two attributes, *id* and *type*. All subsequent types extend from this type and inherit these attributes.

The ***BasicDevice*** and ***device*** types inherits these attributes but also extends the definition. ***device*** defines data that can be collected against devices in general. Implicit in this is that device itself will be extended by types that refer to particular devices. So, device includes the stream *chassisInventory* that are common to all devices.

DeviceEx extends *device* having additional attributes and streams defined against it. It also used for reporting, rather than *device*.

The ***SwitchDevice***, ***RouterDevice*** and ***LBDevice*** types inherit all the characteristics of *DeviceEx*, ***device, BasicDevice*** and ***type***. They also extend the definition by including characteristics, in this example streams, unique to each.

Once these types are compiled into Entuity they affect the characteristics of a discovered device. For example, when Entuity discovers a new switch, it first discovers it as a ***device***, then ***BasicDevice***, ***DeviceEx*** and finally as a ***SwitchDevice***. The object instance created is of ***SwitchDevice***, but as it is derived from a composite type it can still be referred to as a ***device*** and retains its original object identifier. Each discovery cycle is scheduled so a new cycle only starts when the previous cycle has completed. This explains why the attributes associated with a newly discovered device appear in stages.

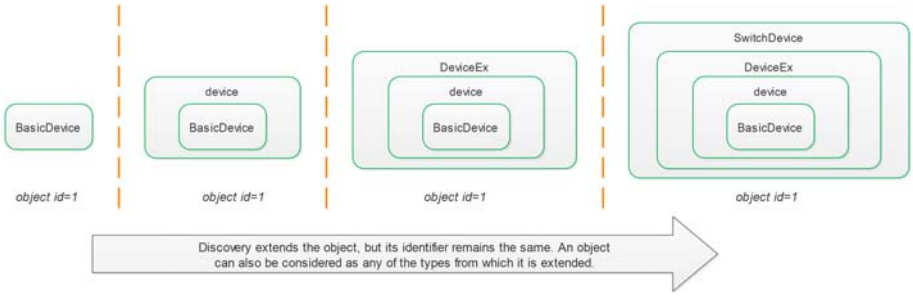

Figure 9 Object Extension

Making Associations Between Types

Associations define relationships between types, tying related information collected through different types together. For example, the default StormWorks configuration includes an association between the types ***port*** and ***device***. A device has one or more ports and this relationship is defined through the DevicePort Association. This association identifies ***device*** as the primary (parent) type and ***ports*** as the secondary (child) type. It also indicates that there is a one-to-many relationship between them.

Connections between types are mediated through association attributes (see *Defining Association Attributes*).

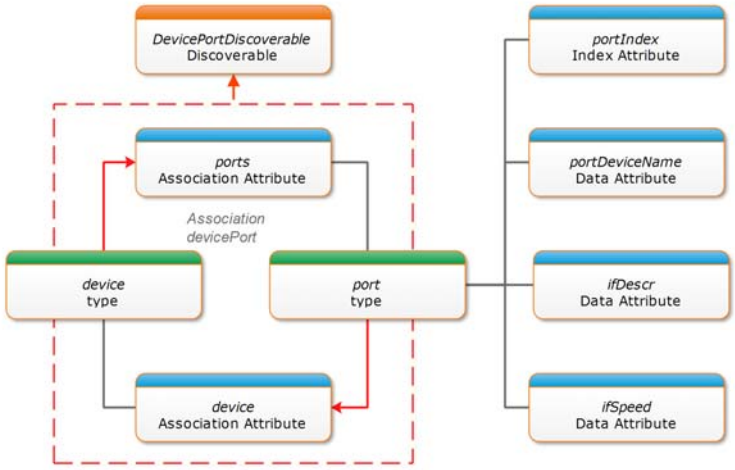

Figure 10 Types, Attributes and Associations

Defining Attributes of Types

StormWorks attributes can be grouped by their usage:

- Data attributes store information collected from network objects.
- Index attributes identify discovered network objects.
- Association attributes make associations between types.

Defining Data Attributes

Data attributes are the attributes on which we want to collect data and display in Entuity and report on. These attributes define data that can be polled. They are held:

- directly against a type. This implies history data for that attribute is not required.
- against a stream within a type. This is for time-series data.

Defining Index Attributes

Index attributes are populated during the discovery process, through calling of the association discoverable (which in turn calls their collector). Index attributes are the key fields of a type's attributes. For example it may be the OID that identifies the discovered instance of an object, e.g. a particular flash card.

Defining Association Attributes

Association attributes are used to make associations between types. As the name implies, these are attributes of the association and do not 'belong' to a type.

Associations involve primary and secondary types, and the association should reflect the relationship between the two types. For example, device to ports is a one to many

relationship. ***device*** is the primary type, ***port*** the secondary type. In StormWorks this association requires two attributes.

- *ports* holds a list of ports. This is all of the ports on the current device. It is used when making the association from the primary type ***device*** to ***port***. It is the primary attribute.
- *device*, is a supplied attribute derived from ***device*** type (see *sw_device.cfg*). It contains all of the attribute information relating to the current device. It is used when making the association from the secondary type ***port*** to ***device***. It is the secondary attribute.

Understanding StormWorks Streams

Streams are the main data gathering mechanism of the StormWorks architecture. Streams consist of a number of attributes amongst which is a filter that specifies which classes of infrastructure objects are to be monitored.

Streams define properties of the polling process. A stream can only be connected to one type, although through type inheritance it can appear otherwise. A type can have one or more streams connected to it.

An example stream definition:

```
[Stream rnewsPort]
KeepTime=8h
ObtainRate=3min
StaleAgeOut=2d
Attributes=ifNUcastPkts
ConnectTo=port
Backup=true
Description=RNEWS port data
Filter=eq(obj://ref/device/devType, 168)
ClientData=showInParent=0\ndisplayName=Traffic\nTime-
stamp.isHidden=1\niconImageURL=http://%host%/EOS/linkstat.gif
EventStreamClientData=isHidden=1
```

Where:

- *Stream* is the keyword that defines the section as a stream definition.
- *Name* is the particular stream's name, e.g. `rnewsPort`.
- *KeepTime* is the length of time to keep the data, defined by an integer followed by:
  - **min** minutes
  - **h** hours
  - **d** day
  - **w** week
  - **m** month

  Care should be taken that KeepTime is compatible with rollup definitions.

- *ObtainRate* is the polling interval.
- *StaleAgeOut* is the length of time Entuity keeps stale data before deleting it.
- *Attributes* is a comma delimited list of attribute names.
- *ConnectTo* sets the object type the stream connects to.
- *Backup* determines whether Entuity backups the stream, by default this is set to false.
- *Description* is a text description of the stream. This must always be completed, and where the stream is displayed in the client then so may be the description.
- *Filter* can either be defined as part of the section or as a reference to a function. A filter uses attributes of the type (e.g. the device sysoid). Only when the object instance meets the filter condition is a stream created.
- *ClientData* contains a set of commands and values for use by the client. Each instruction is separated from the next by \n. Lines can be broken into smaller, readable units by placing = at the start of continuation lines. Possible commands include:
  - displayName
  - isSampleList
  - leadPropertySheet
  - isHidden
  - showInParent
  - Timestamp.isHidden
  - iconImageURL=http://%host%/EOS/linkstat.gif
- *EventStreamClientData* is set to hidden to suppress event values associated with a type being displayed.

Instantiating Streams

For each discovered object that matches the filter, StormWorks creates a stream collector which collects time series sample data for that object.

Stream collectors instantiated from the same stream specification are stored in the same table, which has a name of the form dss_* where * is the stream name. These tables are created dynamically at start-up time. Individual rows are differentiated by their stream collector ids (column name dsStreamInstId) that in turn reference a row in the dsStreamInst table. This table contains the data that connects a stream collector to a unique infrastructure object, a stream and various data relating to the last time samples from this stream were collected and written to the database.

To verify that the intended data is being collected by the stream, Entuity compares the summarized samples captured between two contiguous poll periods. When the values are:

- different a new row is written to the corresponding stream collector table and columns updated in the dsStreamInst table to reflect the time at which this operation took place.
- the same then just the last update time is changed to avoid storing redundant data.

So, it is not sufficient to check the table corresponding to stream collectors arising from a stream alone. The master stream collector table, dsStreamInst, must be checked to ascertain

if a new sample has been written or only the time stamp relating to the stream collector updated.

Appendix F Groovy Used in Entuity

Entuity allows you to develop Groovy scripts for use with the Event Management System, Entuity Reports and Entuity Configuration Management module. The Entuity server includes a Groovy installation which runs those scripts when loaded to the server.

You can develop Groovy scripts for use with the Event Management System, Entuity Reports and Entuity Configuration Management module. This section identifies Groovy concepts that are useful when developing those scripts, for a full introduction to Groovy refer to http://groovy.codehaus.org/. The codehaus website is the primary authority for the Groovy language and provides tutorials, script documentation and useful downloads.

Statement Delimiters

If you place two Groovy statements on the same line they require a delimiter (semi-colon) to separate them:

```
println("Hello "); println("World!");
```

The second delimiter is not required but for consistency you might want to include it.

Statements placed on their own line do not require an end of statement delimiter, although can use one. For example these two lines are both valid and functionally identical:

```
println("hello")
println("hello");
```

Entuity recommend that you place each Groovy statement on its own line as this will make the code easier to read. You also do not have to use semi-colons.

Fundamental Data Types

There are many standard data types of which the most relevant are:

- String
- Character
- Boolean
- Integer
- Long
- Float
- Double

Variables

To instantiate a variable (create a variable at runtime) the *def* keyword is used. Variables should also be initialized during instantiation:

- `def a = "Hello world!";` … a String
- `def b = false;` … a Boolean
- `def c = 13;` … an Integer
- `def d = 23L;` … a Long
- `def e = 12.34F` … a Float
- `def f = 23.45` … a Double
- `def g = 23.45D` … a Double

To avoid confusions between Floats and Doubles the standard Entuity reports force all floating point variables to be Floats.

Printing to the Screen

When debugging a scripts it can be particularly useful to print arbitrary information to a debug screen, for example the value of a variable at an intermediate stage of its processing. You can use either `println()` or `print()` which display the supplied string as either with or without an appended newline respectively.

Boolean evaluation

These following Boolean statements are valid:

- A < B
- A <= B
- A == B
- A != B
- A >= B
- A > B

The value of a Boolean variable can be inverted by preceding it with a "!".

Operators

The standard +, -, *, / operators are supported plus the following:

```
--      Auto-decrement
!       Not
%       Modulo
&       Numerical and
++      Auto-increment
^       Numerical xor
|       Numerical or
1.~     2.Bitwise complement
```

Control Structures

Note that *Statement* can be a single statement or multiple statements separated either by semicolons and/or newlines all enclosed in {} :

```
if (Boolean) Statement
if(Boolean) Statement else Statement
while (Boolean) Statement
for (Variable in List) Statement
switch (Variable) {
case Value: Statement; break;
case Value: Statement; break;
default: Statement;
}
```

In a switch structure the *Variable* may be any data type including Strings. The *case* may also be followed by a closure rather than just a static value so more sophisticated testing can be performed than in a Java switch structure.

Ternary Operator

A ternary operator is a shortcut expression that is equivalent to an if/else branch assigning some value to a variable.

When a decision needs to be made as to what to write to a variable a ternary operator can often be used to shorten the syntax. For example, a very common requirement when writing reports is to protect against empty/null values being returned by the Data API queries. If the mean utilization of a port were requested by a query for a period of time during which the device were totally unreachable the mean utilization would be returned as an empty string. If an attempt were to be made to convert this empty string to a Float using the *toFloat()* method then a runtime error would be raised and the report would crash. This can be handled using the following syntax:

```
def a = "12.34";
def b = (a != "")?a.toFloat():null;
```

If variable *a* were a null string then *b* would be set to a null rather than raising a runtime error.

ArrayLists

One important form way of collecting data together is using an ArrayList. This is a linear list of variables. Technically, it is not essential that every element of an ArrayList have the same data type but in most applications there is uniformity across the elements. A collection of components is returned as an ArrayList from the Entuity Data API. The records returned from the Report Query to the main report are usually in the form of an ArrayList.

An empty ArrayList can be instantiated as follows:

```
def a = [];
```

An ArrayList can be populated during instantiation:

```
def a = [1,2,3,4];
```

An ArrayList can have elements added after instantiation:

```
def a = [1,2];
a.add(3);     … the result would be an ArrayList containing [1,2,3]
a.add(0,9);     … this syntax specifies where to add the new element,
in this case at the beginning
                    the result would therefore be [9,1,2,3]
```

One array list can be appended to another:

```
def a = [1,2];
def b = [3,4];
a.addAll(b);     … the result would be [1,2,3,4]
```

An element of an ArrayList can be obtained using its index:

```
def a = ["a","b","c","d"];
println(a[3]);     … this would display "d"
```

Maps

A map, also known as an associative array, contains a list of data types that can be indexed by key that can be any fundamental data type. This means that it can be used, for example, to hold the values associated with enumerations. For example, if an enumeration for a port state might be 1=Up, 2=Down, 3=Testing, 4=Failed. If the results returned from a Data API call were to be a String with the meaning of the status and there was a requirement to convert this to the corresponding numeric value the following map syntax could be used:

```
def meaning = [Up:1,Down:2,Testing:3,Failed:4];
def status = "Down";
println(meaning[status]);     … the result would be 2
```

Another situation where maps can be useful is where two separate Data API queries have been used to pull back different information about the same components and the two separate datasets need to be merged. The results from the first query could be placed into a map using either the StormWorks ID or component name as the key. The second query would then insert its results into the same map using the same key format. It should be noted, however, that if a corresponding map element does not already exist for a given key the entry must be created using an *add()* method. This means that there must be an explicit test for the existence of an element performed before it is written to and that a suitable element must be created if it does not already exist. This test can take advantage of a null return when a nonexistent element is referenced.

Converting between Data Types

Groovy supports automatic data type conversions using implied rules. In this example the float data type is automatically converted to a string before it is appended to the previous string using the '+' operator:

```
def a = 12.34F;
println("The average utilization is " + a);
```

Automatic type conversions between numeric types happens implicitly:

```
def a = 12.34F;
def b = 20;
println(a + b);      … this would actually display the result as a
Double
```

Typecasting allow one numeric type to be explicitly converted to another:

```
def a = 12.34F;
println((Integer)a);      … this would display "12" as the Float as been
cast to an Integer
```

Numeric Coercion

The resulting data type from various numeric operations needs to be understood and accommodated. The result of performing addition, subtraction or multiplication between common numeric data types is as follows:

+ - *	Integer	Long	Float	Double
Integer	Integer	Long	Double	Double
Long	Long	Long	Double	Double
Float	Double	Double	Double	Double
Double	Double	Double	Double	Double

Table 103 Addition, Subtraction and Multiplication Coercion Rules

However the coercion rules for division are:

/	Integer	Long	Float	Double
Integer	BigDecimal	BigDecimal	Double	Double
Long	BigDecimal	BigDecimal	Double	Double
Float	Double	Double	Double	Double

Table 104 Coercion Rules for Division

Double	Double	Double	Double	Double

Table 104 Coercion Rules for Division

For most practical purposes BigDecimal and Double data types can be thought of as synonymous.

This means that 1/3 in Groovy evaluates to 0.75 whereas in Java it would evaluate to 0 as the Java result would also have been an Integer.

If you want to explicitly divide one Integer by another and get an Integer results use the intdiv() method so *a*/*b* would be expressed as *a.intdiv(b)*

String operators and methods

Strings are one of the most important data types used within report queries. All the data return from Flex queries via the Data API is in the form of Strings no matter whether it actually represents an Integer, Float or any other data type. Unlike in Java, Strings can be enlarged after their instantiation. Strings can be concatenated using the "+" operator:

```
def a = "Hello" + " " + "World!";      … this would result in the String
"Hello World!"
```

There are many String methods available including all of those inherited from java:

```
def a = "12.34";

def b = "56.78";

println(a.toFloat() + b.toFloat());      … this would display 69.12

def c = "39";

c.toInteger();      … convert a String to an Integer … must be an
integer format string to succeed

… float format strings would fail to convert

def d = "   12.34   ";      … a numeric string with white space on either
or both ends

d.trim().toFloat();      … this could strip off the white space to allow
the float to be created

def e = "00123:xyz";

println(e.substring(6));      … displays xyz … the entire string from
character 6 onwards

println(e.substring(6,8));      … display xy … the substring between
characters 6 and 8

def g = "abcDEF";

println(g.toUpperCase());      … displays ABCDEF

println(g.toLowerCase());      … displays abcdef
```

Closures

A Groovy closure is an open, anonymous, block of code that can take arguments, return a value and be assigned to a variable. A closure may reference variables declared in its surrounding scope.

Closures cannot be used when defining expressions for fields within a report definition.

Iteration

All the elements of an ArrayList or similar collection can be iterated over with each element being separately processed by a closure. For example, the numeric contents of a simple ArrayList could be totaled as follows:

```
def a = [3,9,5,18,23,12];
def total = 0;
a.each {
    total += it;
}
println(total);     … displays 70
```

For each iteration through the ArrayList the special variable `it` is set to the next value of the element of the ArrayList. A modification to this syntax would be:

```
def a = [3,9,5,18,23,12];
def total = 0;
a.each { b ->
    total += b;
}
println(total);     … displays 70
```

This shows that instead of using *it* a new variable *b* is being set to the value of the next element instead.

If the index of each element is also required to be know the eachWithIndex technique can be used:

```
def a = [3,9,5,18,23,12];
a.eachWithIndex {val,index ->
    println(index + ":" + val);
}
```

This would display:

```
0:3
1:9
```

```
2:5
3:18
4:23
5:12
```

A similar iteration technique can be used with a map:

```
def meaning = [Up:1,Down:2,Testing:3,Failed:4];
meaning.each {key,val ->
    println("Key:" + key + ", Value:" + val);
}
```

The result of which would be to display:

```
Key:Up, Value:1
Key:Down, Value:2
Key:Testing, Value:3
Key:Failed, Value:4
```

Sorting

An ArrayList can be sorted:

```
def a = [3,9,5,18,23,12];
println(a.sort {it});     … displays [3, 5, 9, 12, 18, 23]
println(a.sort {-it});     … displays [23, 18, 12, 9, 5, 3]
```

Filtering

An ArrayList can be filtered:

```
def a = [3,9,5,18,23,12];
println(a.grep {it<10});     … displays [3, 5, 9]
```

Dynamic Objects

The use of Dynamic Objects in Groovy report queries is especially important given a restriction that prevents Static Objects from being used. Objects allow data and methods to be gathered together and treated as a single entity. Dynamic Objects are widely used in the standard Entuity reports to hold attributes of the same component. The constructor used to create a DynamicObject is supplied in a utility class called com.entuity.jasper.Utils . Once a DynamicObject has been instantiated it can have any arbitrary set of fields added dynamically:

```
def obj = new com.entuity.dataapi.dynamic.DynamicObject();
obj.name = "Router A";
obj.id = 1234;
```

The above example created an object called *obj* and gave it a String field called *name* and an Integer field called *id*. If an ArrayList is to be populated with Dynamic Objects a new DynamicObject must be created for each element of the ArrayList.

The following code fragment was taken from the Routing Summary report and shows a multi-server request for data where the Entuity server on which each device is being managed needs to be recorded along with the attributes of that component. An ArrayList called *allResults* containing all the routing devices is populated with Dynamic Objects:

```
def query = forUserServersInParallel ($P{eyeServer}, { it.runFlex-
Query([
    'viewId=' + $P{view},
    'type=DeviceEx',
    'DeviceEx.filter=' + (($P{device} == null) || ($P{device} == "--
AllDevices--"))?'eval(this,    var,    variable    obj    =
DeviceEx(this);(obj.devType == 168) || (obj.devSysCapabilities &
2))':'(eval(this, var, variable obj = DeviceEx(this);obj.name ==
"$P!{device}"))'),
    'DeviceEx.attr=name'
]) })

def allResults = []

query.each {
    def eyeServer = it.server;
    if (it.ok) {
        if (it.result.content.devices) {
            it.result.content.devices.each {
                 def entry = new com.entuity.dataapi.dynamic.Dynamic-
Object();
                entry.server = eyeServer;
                entry.deviceName = it.name;
                allResults.add(entry);
            }
        }
    }
}
```

Each of the resulting Dynamic Objects that is placed into *allResults* has the ID of the Entuity server identified by a field called *server* and the name of the device in a field called *deviceName*. This approach had to be used because the server ID was not available along with the other attributes returned by the *runFlexQuery()* method.

Referencing fields within an object safely

To access the *ok* field in a simple object *obj* the syntax would be:

```
obj.ok
```

However, if obj were actually to have a null value then this reference would result in a Null Pointer Exception (NPE). This is a problem when iterating over objects in an ArrayList where it is possible that one or more of the objects might actually have a null value. The problem becomes compounded when the objects are nested several level deep as is the case with the results of a Flex query performed through the Entuity Data API. A typical iteration through devices might look as follows:

```
query.result.content.devices.each {}
```

If any of the levels in this nested sequence has a null value an NPE will be raised and, unless there is explicit protection around the operation the query will fail. It is possible to test each and every level of the nest before referencing the full sequence but this is a laborious and ugly prospect. Fortunately there is simple Groovy syntax that provide a safe method without any extra lines of code using the safe dereferencing operator "?." . The same iteration could be safely specified using:

```
query?.result?.content?.devices.each {}
```

If any level of the sequence returns a null then the whole result is a null which safely results in zero loops of the closure being iterated around.

ResultsBean

A ResultsBean has a similarity to a DynamicObject but is used exclusively in report queries to pass collections of records and other fields back to the report. The Report Query in the Routing Summary report ends with the following lines:

```
def result = new com.entuity.dataapi.dynamic.ResultsBean(allResults)
result.deviceCount = allResults.size();
return result;
```

This takes a previously create ArrayList of DynamicObjects called *allResults* and creates a new ResultsBean that contains it. An additional field called *deviceCount* is then added to the ResultsBean to allow the count of devices to be returned and used in the Title Band of the report. It should be noted that this is in contrast to the contents of the DynamicObject that represents each record which is used to populate the Details Band.

Controlling the numeric precision of floating point numbers

There are two main ways to specify the number of decimal places and other formatting aspects of floating point numbers in reports. The first approach is the easiest and can be used if the number is being displayed on its own and not as part of a longer string. If the number is either passed to the report from the report query as a Float or Double it will

automatically configure the Text Field to that data type when dragged into a band on the report. The precision can be defined within the properties of the text field. Even if the number if passed as a String it can still be converted to a Float or Double using a suitable expression within the Text Field. If this is the approach taken then ensure that the data type setting of the Text Field matches the resulting data type of the expression.

If the floating point number is mixed with other text then the simple formatting controls within a Text Field cannot be used. An alternative approach is to convert the Float or Double into a string using the DecimalFormat class which also allows control over its numeric precision display. An example of the required syntax follows:

```
def val = 12.34567F;
def str = new java.text.DecimalFormat("#.##").format(val);
```

In this case the value of the String variable *str* would be set to *12.35* as it would also perform rounding.

Handling dates and times

Timestamps used within Entuity are of the UNIX format which means they are integers representing the number of seconds since 00:00 1st Jan 1970 GMT. The Java standard for a timestamp, which is inherited by Groovy, is a long containing the number of milliseconds since 00:00 1st Jan 1970 GMT which means it is a factor of 1000 greater than the UNIX version. This can become relevant when converting a timestamp that was obtained from Flex queries via the Data API into a descriptive string.

Groovy and Entuity Reports

The part of a report that has the most dependence on a programming language is the Report Query. Although the primary function of a Report Query is to gather data from one or more Entuity servers for use in a report it also, in most cases, needs to manipulate and post-process the raw data before it is ready to be used. The primary way in which data is obtained from Entuity servers for use in reports is via the Entuity Data API which is implemented as a Java class. Groovy can be thought of as a superset of Java in that it can take advantage of all existing Java class libraries but also adds a layer of sophistication to ease its implementation. Although there is an option to use either Java or Groovy for reports the standard reports supplied with Entuity use Groovy as it allows the coding to be shorter, clearer and simpler to write.

Glossary

802.1p

An IEEE standard for providing quality of service (QoS) in 802-based networks. 802.1p uses three bits (defined in 802.1q) to allow switches to reorder packets based on priority level. It also defines the Generic Attributes Registration Protocol (GARP) and the GARP VLAN Registration Protocol (GVRP). GARP lets client stations request membership in a multicast domain, and GVRP lets them register into a VLAN.

AAL (ATM Adaptation Layer)

AAL enhances the service provided by the ATM layer to a level required by the next higher layer. It performs the functions for the user, control and management planes and supports the mapping between the ATM layer and the next higher layer.

Advanced Actions

Advanced Actions, also known as user menus and user actions, are defined through configuration files. Actions may be automatically triggered through Entuity raising an appropriate event, or interactively through advanced action menus, available both from the menu bar and context menus.

Agent

Intelligent management software embedded in a network device. In network management systems, agents reside in all managed devices and report the values of specified variables to management stations.

Antenna / Radio

Each Wireless Access Point has one or more Antennas. Each Antenna is attached to an 802.11 radio within the Access Point. Wireless Hosts communicate with the network via a wireless association with an Antenna/Radio. Each Antenna/Radio can have multiple hosts simultaneously attached. Each Antenna/Radio operates in a chosen 802.11 compatibility mode such as 802.11a, 802.11b or 802.11g. Additionally, each Antenna/Radio has a single SSID assigned. Each Antenna/Radio operates on a chosen radio channel and with a specified transmit power setting, which is measured in mW. Many controller based installations use dynamic optimization algorithms to pick a suitable channel and power setting. Frequent auto-adjustment of these setting indicates that there are problems being encountered with the quality of the wireless communications.

AP (Access Point) / WAP (Wireless Access Point)

A device that has one or more 802.11 radios and Wireless Antennas. For example, laptops, PDAs, connect to a wired LAN through an AP, which is a hardware device or software that acts as a communication hub.

It bridges traffic from wireless attached hosts to/from an Ethernet interface that connects to an access layer switch port. APs provide heightened wireless security and extend the physical range of a wireless LAN. The access layer switch will see the MAC addresses of the individual wireless attached hosts (the MAC address of the wireless NICs) plus the MAC of the Access Point Ethernet interface.

AR System

BMC Remedy Action Request System (AR System) is a framework within which applications are built by AR System administrators. Applications consist of a set of AR System forms that are linked using workflow rules designed for the application. These forms contain fields which Entuity can be configured to populate.

ARs

Entuity integrates with AR System to generate Action Requests (ARs). The sample integration with the Remedy Help Desk includes ARs of the type incident.

ARP

ARP (Address Resolution Protocol) is the layer 2 standard for TCP/IP. It is used to obtain a node's physical address when only its logical IP address is known.

ATM

ATM (Asynchronous Transfer Mode) is a packet-switching technology, that delivers high-speed performance together with a scalable architecture. Its use of small packets (fixed length cells of 53 bytes), provide for low latency so sound and vision arrive together. It can also handle bursty, non time-sensitive data, translating variable length packets to fixed size packets.

Attribute

In Entuity an attribute is a property of an object that is defined through Entuity Configurable Framework. Attribute data can be charted using the Attribute Grapher and is available to Report Builder.

Autonomous Wireless Access Point (AWAP)

A Wireless Access Point (WAP) that embodies all of its necessary control functionality in a self-contained manner. AWAPs are usually connected to switched access layer ports and

can coexist with ordinary wired connections to end user hosts and servers on the same switch. AWAPs do not require wireless controllers and do not interact with them if they exist.

Backbone

The part of a network that acts as the primary path for traffic that is most often sourced from, and destined for, other networks.

BECN (Backward Explicit Congestion Notification)

BECN is a bit in the header of a frame-relay frame that is set when frames are sent on the data path backwards from destination to source. It indicates congestion to the source node.

WAN News combines BECN and FECN values to determine congestion on a data path.

Bandwidth

The upper limit of the rate at which data can be transferred.

BMC Atrium CMDB

The BMC Atrium Configuration Management Database (BMC Atrium CMDB) is a data repository that provides a working model of your enterprise IT infrastructure.

BMC Cell

BMC Impact Manager instance. A cell receives events from Entuity and displays them in the BMC IX.

BMC II Web Services Server

BMC Impact Integration Web Services Server. You can connect to the BMC II Web Services at the end point as defined by the URL format, http://*webServerHostName*:*webServerPortNumber*/*webServiceName*, e.g. http://decade:6080/impactManager.

BMC IX

BMC IX (BMC Impact Explorer) displays events received from Entuity.

BMC ProactiveNet Performance Management

BMC ProactiveNet Performance Management which receives events from Entuity.

Blackout

Blackout is complete loss of the network, as opposed to a brownout, which is degradation in the performance of the network.

BPDU

Bridge Data Protocol Units are special frames that contain spanning tree information. There are two types of BPDU, Topology Change Notification (TCN) BPDU contains topology change information, Configuration BDU contain configuration information.

Bridge

A device that interconnects local or remote networks. Bridges form a single logical network, centralizing network administration. They operate at the physical and link layers of the OSI Reference Model.

Brownout

Brownouts, also known as soft faults, are typically caused by cabling faults, faulty transceivers, faulty NIC cards and configuration errors such as duplex/half-duplex mismatches. These problems cause a percentage of the packets traversing that particular area of the network to be corrupted. The total number of packets discarded as a percentage of packets is directly related to the severity of the brownout.

Burst

Burst is the access rate of the physical connection to the Frame Relay carrier network.

Central Server

A central server is an Entuity server trusted by remote Entuity server(s). A user logged into the central Entuity server is able to view information collected by the remote Entuity server(s), according to their user account access rights. A remote Entuity server responds to requests from a trusted central Entuity server, and freely shares information with it.

An Entuity server can be configured to perform both roles, be both a remote and central Entuity Server. This allows administrators to create both hub-n-spoke and fully meshed deployments.

A central Entuity server can also act as a central license server. From it you can allocate, and de-allocate, license credits to its remote servers.

Configuration of central and remote servers is through the Multi-Server Administration area of the Entuity web UI.

CDP (Cisco Discovery Protocol)

CDP is primarily used to obtain protocol addresses of neighboring devices and discover the platform of those devices. CDP can also be used to show information about the interfaces your router uses. CDP is media- and protocol-independent, and runs on all Cisco-manufactured equipment including routers, bridges, access servers, and switches.

Entuity uses CDP as a method when maintaining links on maps and identifying trunk ports.

CI

Within BMC Atrium CMDB a Configuration Item (CI) is a collection of objects related to the specific functionality of a larger system.

CIR

Committed Information Rate is the rate (in bps) that the network agrees to transfer information over a permanent virtual circuit (PVC) in Frame Relay. The CIR applies to the rate of data entering the network.

Cisco IOS IP SLA Operations

Cisco IOS IP SLA Operations are created on devices by Entuity (via SNMP). Entuity currently fully supports DHCP, DNS, HTTP, HTTP Raw, ICMP Echo, ICMP Path Echo, TCP, UDP Echo, UDP Jitter and UDP Jitter VoIP operations. Entuity can also monitor operations other than these ten, for example FTP. The completeness of the returned data depends upon how close the operation's data structure corresponds to Entuity's default representation of the IP SLA operation data structure.

These are the ten fully supported operations:

- DHCP, Verify availability of dynamic IP addresses.
- DNS, DNS server functionality check.
- HTTP, Web page availability.
- HTTP Raw, Web page availability.
- ICMP Echo, Simple connectivity tests.
- ICMP Path Echo, Simple connectivity tests.
- TCP, Connect Application availability.
- UDP Echo, Simple connectivity tests.
- UDP Jitter, Detailed latency measurements (requires IP SLA on both devices).
- UDP Jitter VoIP, Detailed latency measurements (requires IP SLA on both devices).

Client

A computer that requests a service from another. In Entuity the Java client is Component Viewer which requests, for example, information from the Entuity server on the devices on your network.

Collisions

Collisions occur when two transmitters attempt to send data at the same time. The greater the number of collisions the poorer network performance appears.

Component Viewer

Component Viewer is the Entuity Java client, available through the web UI Tools menu. Through it you can quickly scan the network for both current and historical performance

data. It creates an intuitive hierarchy which lets you easily view configuration settings, check status information and launch fault, utilization and traffic volume history graphs.

Context Menus

Context menus are available from the Entuity web UI and Component Viewer. The contents of the menu are dependent on the position of the mouse when you clicked the right button.

Core Ports

Entuity considers core ports, as WAN ports, administratively up ports which have a configured IP addresses (i.e. layer 3 ports) on devices which are routers or have router capability, or trunks and uplinks that are administratively up.

By default the port status event, Port Operationally Down, is only enabled for core ports.

Current Configuration

The device configuration (either startup- or running) currently being processed.

DLCI (Data Link Connection Identifier)

A unique logical identifier assigned to a PVC end point in a frame relay network. It identifies a particular PVC endpoint within a user's access channel therefore allowing multiple connections to many destinations over a single, physical channel.

Data Management Kernel (DMK)

The DMK supports Entuity's intelligent discovery function. It includes out of the box data models for a wide range of managed devices including hundreds of Ethernet switches and routers. These customizable data models define the attributes of each managed element, its possible dependencies in relation to other elements of the network, and the specific details to retrieve for each element. The DMK manages these data models and automatically applies updates and changes to the Entuity database schema.

Data Path

A data direction on each PVC is a data path. For example, a PVC that connects points A and B has two data paths, from A to B and from B to A. WAN News analyzes the data paths separately.

Data Rollup

Data Rollup is a method of taking polled data and bundling it into larger more manageable units, e.g. rolling 24 hourly datapoints into one daily sample. If Entuity generated monthly reports from live polled data then this would cause a significant increase on the processing overhead, i.e. instead of one datapoint for each day there would be hundreds.

DE (Discard Eligibility)

DE is a bit in the header of a frame-relay frame that indicate the frame may be discarded in preference to other frames if congestion occurs. It is usually set by a network node if the user is offering data (frames) at a higher rate than has been negotiated. This maintains the committed quality of service within the network. Frames with the DE bit set are considered to be excess data.

Derived Events

IA derived event is an event derived from an existing event definition. It retains the event identifier of the original definition, unlike a custom event which has its own unique identifier. Derived events are defined as part of an action. They useful for adding additional information to an incoming event, and can also be called from an incident.

Devices

In Entuity devices refers to network devices, for example switches and routers.

Device Support Datasets

Device support datasets define the attributes of each managed element, its device type, its possible dependencies in relation to other elements of the network, and the specific details to retrieve for each element. This comprehensive library streamlines modeling and ultimately shows exactly what you own, where it is deployed and how it is connected.

Datasets are available through these types of vendor files, all have a .vendor extension. These vendor files are, listed in ascending order of priority:

- `newbin.vendor`, which is created in *entuity_home*\etc when Entuity discovers devices with sysoids for which there is not a device support dataset. These generic device support datasets should be considered temporary definitions, and only used until Entuity supply an appropriate vendor file.

  Device support datasets in `newbin.vendor` have the lowest priority when Entuity is determining which vendor device definition to use to manage a device type.

- `bin.vendor` has the second lowest priority when Entuity is determining the source of device information.Device support datasets in bin.vendor have the second lowest priority when Entuity is determining which of those available to use to manage a device type.
- exotica vendor files are installed to *entuity_home*\`etc\exotica`. Exotica files are only used by Entuity when they are copied to *entuity_home*\`etc`, either manually or during Entuity configuration, e.g. when selecting a module.

  Device support datasets in exotica vendor files have the highest priority when Entuity is determining which vendor device definition to use to manage a device type.These files use a simple naming convention, using the vanilla filename, with a plus sign in the filename and identifying name, e.g. `SOLSERV+managed Host.vendor`.

  During Entuity upgrades configure identifies and removes exotica files from the installation that are now part of the updated bin.vendor.

`vendinfo` identifies the vendor device support datasets available to Entuity and the decisions made when more than one vendor file is available for a particular sysoid; which device support dataset Entuity uses to manage that device type (as identified through its sysoid).

Device Types

In Entuity every device has a type, which you can view through the web interface and Component Viewer. The device type is derived from its vendor file information, and helps to determine how Entuity manages a device. Device types include hubs, switches and routers. There are also two Unclassified device types, Basic Management and Ping Only, and also Full Management.

Unclassified device types have two distinct roles:

- Basic Management and Ping-only, is used for those devices Entuity has taken under management at the Basic Management and Ping-only level.
- Full Management, is used for those devices Entuity has taken under management at the Full level but for which there is no vendor file information but Entuity can generate a suitable generic device type. These are uncertified devices.

Domains

Domains and domain filters are terms used within Component Viewer, in fact supplied domains are now only used within Component Viewer to group objects in its Explorer tree, e.g. the routers domain. In the web UI, where you manage views In Entuity, domain filters are referred to by the more apt term view content filters as they determine the type of object that can potentially appear in a view.

DHCP Operation

The IP SLA DHCP operation measures the round trip time (RTT) taken to discover a DHCP Server and obtaining a lease from it. After obtaining an IP Address, Cisco IOS IP SLA releases the IP address that was leased by the server.

The Dynamic Host Configuration Protocol (DHCP) is an Internet protocol for automating the configuration of computers that use TCP/IP. DHCP can be used to automatically assign IP addresses, to deliver TCP/IP stack configuration parameters such as the subnet mask and default router.

Drop Box

Drop box acts as a temporary repository for objects, for example gauges, charts, links, device metrics, that you want to include to new reports, dashboards.

Duplex

A full-duplex link with one telegrapher at each end, transmitting alternately in each direction.

Dynamic Thresholds

Dynamic thresholds enable Entuity to alert the user to deviations from what Entuity's previous polling has established as normal behavior for that hour on that day. Entuity establishes normal behavior for a given attribute on a given port by maintaining the last four weeks worth of polled data, and applying an averaging algorithm.

EIR

The Excess Information Rate (EIR) is the sustainable rate of information in excess of CIR, that the network will deliver if there is available bandwidth. The total information rate is CIR + EIR.

Frame Relay allows data rates in excess of the CIR to be successfully used on occasions. It is also possible that the amount of data that can be transferred per measurement interval (Tc) may be limited to less than the burst (or access rate) of the physical connection to the carrier network.

EIR defines how many bits per second beyond the CIR the data rate may be exceeded. This is may be policed by the carrier ingress switch per Tc on a pro-rata basis. This means that although data can be transmitted for periods of time at the burst rate of the physical port it would not be possible to continue transferring data at this rate successfully on a continuous basis if the CIR+EIR were to be less than the burst rate.

Entuity

Entuity is both the name of the network management software and the company producing it. Entuity software is designed for networks of any size and complexity, from the smallest, simplest corporate infrastructure to the largest multinational. Every customer can access the full functionality of our cornerstone solution, incorporating fault, performance and inventory management.

entuity_home

entuity_home is used within the Entuity documentation to indicate the Entuity server's root folder. The root folder is set by Entuity `install`, in Windows environments the default is `C:\Entuity`. You can view its current setting through *destination* in *entuity_home*`\etc\entuity.cfg`. Within Entuity configuration files it is represented by the variable *ENTUITY_HOME*.

Ethernet

IEEE standard network protocol that specifies how data is placed on and retrieved from a common transmission medium. Forms the underlying transport vehicle used by several upper-level protocols, including TCP/IP and XNS.

Events

Events are alerts and alarms that are generated through Entuity monitoring the network. Event Viewer displays events and they can also be reported on.

Expect

Expect is a Unix automation and testing tool, written by Don Libes as an extension to the Tcl scripting language, for interactive applications such as telnet, ftp, passwd, fsck, rlogin, tip, ssh, and others. It uses Unix pseudo terminals to wrap up subprocesses transparently, allowing the automation of arbitrary applications that are accessed over a terminal. With Tk, interactive applications can be wrapped in X11 GUIs.

Eye of the Storm® (EYE)

Until Entuity 12.5 the software was known as Eye of the Storm (EYE).

Entuity Remedy AR System Integration

The Entuity Remedy AR System integration allows forwarding of event and managed object information from Entuity to one or more AR System servers.

Entuity allows two types of forwarding:

- automatic generation of Action Requests (ARs), derived from Entuity events, to particular application forms on target AR System servers
- interactive generation of Action Requests (ARs), initiated from Entuity. The specified application forms on target AR System servers are opened for editing, with default data populated from the current Entuity managed object(s) or event(s).

Entuity can also pass to AR System a URL identifying the managed object that is the source of the AR. From AR System you can open Entuity's Component Viewer with the focus on the managed object.

Factory Default

The shipped values of event thresholds are the factory defaults. You can amend a factory default, which if done at the root level effectively changes the default value for all objects against which that threshold can be set. For example, if you amend a threshold setting for a device event at the Entuity (system) level, all devices on that server will have a new default value.

FEC

Forwarding Equivalence Class (FEC) is central concept to MPLS. An FEC is a set of packets that a single router forwards to the same next hop, using the same interface and with the same handling (e.g. queuing). The FEC is determined only once, at the ingress to an LSP, rather than at every router hop along the path.

FECN (Forward Explicit Congestion Notification)

FECN is a bit in the header of a frame relay frame that is set to indicate to the destination node that congestion is occurring on the network. WAN News combines BECN and FECN values to determine congestion on a data path.

Filters

Filters in Entuity act by filtering in those objects specified in the filter. There are three types of filters, view, event and Flex Report.

Entuity uses these types of filter:

- View content filters are applied to the views, restricting the components available from a view to those that meet the criteria.
- Event Filters restrict the events available through a view.
- Flex Report filters restrict the data included to the report.

Flow Collector

The Flow Collector is the set of processes within an Entuity Integrated Flow Analyzer responsible for the receiving, processing and storage of flow records.

Administrators can enable/disable an Entuity server's Flow Collector through `configure`, a decision which should be made according to the role the administrator wants the server to perform in the management of the network.

Frame Relay

A fast packet protocol that relies on physical component and higher level software reliability. The network discards any frame with bit errors. Frame relay services include PVCs (Permanent Virtual Circuit) and SVCs (Switched Virtual Circuit).

Full Duplex

A full-duplex link with one telegrapher at each end, transmitting alternately in each direction.

Generic Device Type

Entuity uses the concept of an underlying generic object against which are mapped the characteristics of different device types, e.g. routers, switch, firewalls, BladeCenters. This allows complete management of devices that have characteristics of one or more of the traditional types of devices, e.g. a router with switching capabilities.

Half-Duplex

A type of communication channel using a single circuit which can carry data in either direction but not both directions at once.

Host Identifier

Your Entuity representative requires the host identifier of the Entuity server machine before they can generate your license. The host identifier associates the Entuity license with the physical footprint of the machine. Entuity install and configure programs both display the host identifier, alternatively you can run the command line program `hostIdent` (which is included with the software but is also available from the Support website).

Hot Standby Router Protocol (HSRP)

Hot Standby Router Protocol (HSRP) establishes a framework between network routers to achieve default gateway failover if the primary gateway becomes unavailable in close association with a rapid-converging routing protocol like EIGRP or OSPF. By multicasting packets, HSRP sends its hello messages to the multicast address 224.0.0.2 (all routers) using UDP port 1985, to other HSRP-enabled routers, defining priority between the routers. The primary router with the highest configured priority will act as a virtual router with its own IP and MAC address, which the hosts on the local segment will be configured to use as a gateway to the destination in question. If the primary router should fail, or the link to the destination drop, the router with the next-highest priority would take over communications through alternative routes within seconds, without major interruption to network connectivity.

HSRP and VRRP on some routers have the ability to trigger a failover if one or more interfaces on the router go down. This can be useful for dual branch routers each with a single serial link back to the head end. If the serial link of the primary router goes down, you would want the backup router to take over the primary functionality and thus retain connectivity to the head end.

Hypervisor

A hypervisor, also called virtual machine monitor (VMM), allows multiple operating systems to run concurrently on a host computer. The hypervisor presents to the guest operating systems a virtual operating platform and monitors the execution of the guest operating systems. Multiple instances of a variety of operating systems may share the virtualized hardware resources. Hypervisors are installed on server hardware whose only task is to run guest operating systems.

Infrastructure Ports

Entuity considers infrastructure ports, as:

Entuity considers infrastructure ports, as router ports, as uplinks which are ports connecting routers with switches and as trunk ports which are ports connecting switches together.

- Router ports.
- Uplinks, ports connecting routers with switches.
- Trunk ports, ports connecting switches together.

Interface

This is the entity on a node which is polled, such as a physical port. Nodes are likely to have more than one interface.

IP

In TCP/IP, the standard for sending the basic unit of data, an IP datagram, through the Internet.

IP Link

IP links may be autoDiscovered or created manually. They represents a link of some form at layer 3 or above e.g. a pair of IP addresses, an IP address and a URL.

IP Peering

IP Peering provides visibility into your WAN links, i.e. leased line, Frame Relay DLCIs, ATM VCCs, using subnet masking. It also reflects any manual IP pairings you may have made in Entuity.

ISO

International body that is responsible for establishing standards for communications and information exchange; developed the OSI reference model. ISO is not an acronym, but the Greek word for "equal."

Java Web Start

A technology for simplifying deployment of Java applications. It allows you to download and launch the Entuity client from your Web browser or shortcuts placed on your PC.

Key Metrics Gauge

From Entuity's Explorer you can access the Device and Port Summary pages, both of which display Key Metric graphs. Key metrics vary according to the managed object, e.g. Device CPU utilization, Port Inbound Utilization%.

These graphs are of two forms a:

- green only gauge is used with metrics that do not have thresholds.
- green and red gauge is used with metrics that have thresholds. When the indicator is pointing to the red area then the threshold has been crossed. The relative size of the red and green areas of the gauge is fixed, i.e. the red area does not take a larger or smaller proportion of the total area of the gauge on changes to the threshold level.

  You can view the current threshold value by passing the cursor over the data value below the graph.

You can click on each key metric gauge to view a larger graph.

LAP (Lightweight Wireless Access Point)

A low cost Wireless Access Point (WAP) that delegates much of the control functionality usually embodied within an Autonomous WAP to a WC. LAPs are usually connected to switched access layer ports and can coexist with ordinary wired connections to end user

hosts and servers on the same switch. The associations between the LAPs and WCs are negotiated dynamically and can change under fault conditions.

A LAP is an AP that is designed to be connected to a wireless LAN (WLAN) controller (WLC). The LAP provides dual band support for IEEE 802.11a, 802.11b, and 802.11g and simultaneous air monitoring for dynamic, real-time radio frequency (RF) management. In addition, Cisco Aironet 1000 Series LAPs handle time-sensitive functions, such as Layer 2 encryption, that enable Cisco WLANs to securely support voice, video, and data applications.

Entuity Wireless currently supports Cisco LAP, part of the Cisco Unified Wireless Network architecture.

Leased Line

A leased line is a dedicated point-to-point connection over a WAN via a router at the subscriber's premises to the telecommunications provider.

Entuity identifies a leased line, by default, when both of these conditions are true:

- The interface type is either IANAifType 22 (propPointToPointSerial) or 23 (PPP).
- The WAN port is not:
- A Frame Relay port.
- An ATM port.
- An ISDN port. These are identified as having an associated lower layer protocol port (found from the ifStack table) of ifType 81 (ds0). This indicates the port is a layer on top of either basic rate or primary rate ISDN.

Link Layer Discovery Protocol (LLDP)

The IEEE 802.1AB Link Layer Discovery Protocol (LLDP), provides a solution for the configuration issues caused by expanding LANs. It runs over the data link layer and specifically defines a standard method for Ethernet network devices to advertise information about themselves to other nodes on the network and store the information they discover. LLDP is available as a technology link type on the Entuity maps.

Load Balancers

Load balancers are devices that control and optimize traffic flow over your network. For example directing traffic away from over utilized servers to those less utilized, improving mission critical service delivery, providing fall over protection.

Entuity delivers a similar level of fault, performance and inventory management for load balancers as provided for other standard Entuity device types, e.g. routers, switches, hubs. For example device reports include load balancers, you can build your own reports using Flex Reports, device and port events apply and full load balancer details are viewable through Component Viewer.

Entuity currently manages F5 Labs Big IP 6400 Load Balancer. Entuity delivers additional polling of the device ports using F5 lab's propriety MIB, returning additional port identification, port status, port traffic and port utilization data. The full integration of this

additional data within Entuity allows administrators to set up utilization and traffic events against this data.

Log Files

Entuity process messages are written to their individual log files, in *entuity_home*/`log`. For example, `applicationMonitor` writes to `applicationMonitor.log`. When the log file becomes full, it automatically wraps to another file with up to four versions, e.g. `applicationMonitor.log.1`, `applicationMonitor.log.2`, `applicationMonitor.log.3`.

Management Level

Every device under Entuity management is managed according to its management level, which is set when the device is added to Entuity but can be subsequently amended. Each managed device costs one license object.

These are the management levels:

- Full Management (all interfaces), Entuity manages all interfaces on the device.
- Full Management (management interfaces only), Entuity only manages the management interface.
- Full Management (no interfaces)
- Basic Management Entuity collects only basic system information and the full IP address table via SNMP. This management level is used when Entuity does not have the appropriate device support dataset (vendor file), cannot generate an appropriate dataset or you only want the device placed under basic management. Entuity does not manage any ports or modules on the device.
- Ping Only, devices only under ping management, SNMP data is not collected for these devices.

Managing Agent

Handles requests for information or action from the management station on a node. A protocol links the management station and the Managing Agent; for Entuity users this must be SNMP.

MIB (Management Information Base)

Entuity supports SNMP MIBs only. MIBs are present within nodes on a network, and comprise a logical collection of managed objects arranged in a tree structure. Managing agents on an element use MIBs to store information regarding the element, e.g. the speed at which packets of information are transferred.

All managed objects within a MIB share a common root.

Mobility Controller

An SNMP manageable hardware device, manufactured by Aruba, that controls and coordinates the operation of a group of Aruba Wireless Access Points. In an Aruba wireless network deployment all wireless equipment discovery and real-time monitoring is performed via the Mobility Controllers rather than via SNMP/ping monitoring of the individual Access Points.

Multicast

Network communication between a single sender and multiple receivers.

My Network

Supplied view that contains the entire set of managed object's the user is permitted to view. Different users may have different devices in their My Network view, reflecting their different access permissions.

Node

An SNMP managed device attached to a network, from which data can be retrieved. For example, node devices such as hubs, routers, bridges, or network printers.

OID

An Object Identifier is a sequence of integers that represent the position of an object in the hierarchical structure of objects in a MIB.

OMF (Open Modeling Framework)

Flexible Entuity framework that allows the fast integration and management of new types of managed objects, e.g. new device types. For example, the BladeCenter device type is implemented through the OMF.

OSI Model

A model for networks developed by International Standards Organization (ISO). The network is divided into seven layers, each layer building on the services provided below it.

Packet

Any logical block of data sent over a network; it contains a header consisting of control information such as sender, receiver, and error-control data, as well as the message itself. May be fixed or variable length.

PCR (Peak Cell Rate)

PCR is the maximum short term data throughput supported by an ATM port; the limit to which traffic can burst.

Percentile Utilization

Percentile Utilization indicates that for a defined percent of the time, e.g. 95, port utilization is below this value. It is useful for monitoring the sustained utilization of the port.

The 95th percentile is derived by ordering the utilization data by value, from highest to lowest. Application of a least square fit method removes spikes that would distort the analysis. The top 5% values are discarded, leaving the 95th percentile. This value is calculated for both inbound and outbound utilization.

Policy Group

Entuity licensing is enabled by grouping related types of managed objects into groups. These Policy Groups are then assigned a license credit quota. Before Entuity manages an object it first checks whether the license allows its management and then whether a credit is required. When a license credit is required, Entuity checks that the policy group to which the object's type is associated has available credits. For example, before Entuity manages a device it checks the device licensing policy group for available credits.

Polling

Devices on the network are accessed by the system at regular, pre-defined, intervals in order to retrieve required data. This is referred to as polling the devices.

Polling Engine

The Polling Engine (or Core Management Engine) is the set of processes within an Entuity server responsible for all general network management tasks excluding flow collection (e.g. network discovery, inventory, monitoring, event management).

Administrators can enable/disable an Entuity server's Polling Engine through `configure`, a decision which should be made according to the role the administrator wants the server to perform in the management of the network.

Port

Entuity considers ports as interfaces on network devices, e.g. routers, and as endpoints in communications systems. In IP an upper-layer process that receives information from lower layers. Ports are numbered, and each numbered port is associated with a specific process. For example, SMTP is associated with port 25.

TCP and UDP transport layer protocols used on Ethernet use port numbers to distinguish between (demultiplex) different logical channels on the same network interface on the same computer.

Protocol

A set of formal rules detailing how to transmit data across a network. Example protocols include TCP, UDP and IP.

PVC (Permanent Virtual Circuit)

PVC is a Frame Relay virtual connection providing the user with the equivalent of a physical connection to a destination address, using shared facilities. Virtual circuits can be permanent (PVC) or switched (SVC).

Reachability

Availability Monitor sends an ICMP ping to the management IP address of managed devices, by default every two minutes. Devices that respond are considered reachable, those that do not respond, after the set number of retries, are considered unreachable. When Availability Monitor (`applicationMonitor`) is not running, then the reachability of the device is Unknown for that period, although Entuity maintains the last known state of the device.

Reboot

Entuity uses the device sysuptime to calculate when the device was last rebooted, or more accurately when the device last came up after being rebooted.

Reconciliation Rules

Within BMC Atrium reconciliation rules are applied by the reconciliation engine to improve accuracy and efficiency of maintaining IT environment data in the CMDB. Reconciliation is used to identify and merge CI information and relationship form imported dataset with production dataset.

Remedy Help Desk / Service Desk

Entuity Remedy AR System Integration for Remedy AR System 7.0 includes a sample configuration which integrates with the Remedy Service Desk application.

Remote Server

A remote server is an Entuity server configured to trust another central Entuity server. A user logged into the central Entuity server is able to view information collected by the remote Entuity server(s), according to their user account access rights. A remote Entuity server responds to requests from a trusted central Entuity server, and freely shares information with it.

An Entuity server can be configured to perform both roles, be both a remote and central Entuity Server, allowing administrators to create both hub-n-spoke and fully meshed deployments.

Configuration of central and remote servers is through the Multi-Server Administration area of the Entuity web UI.

Router

A device that routes data between networks. Routers connect multiple LAN segments to each other or to a WAN.

Routers may be equipped to provide frame relay support to the LAN devices they serve. These routers can:

- encapsulate LAN frames in frame relay frames and send those frames to a frame relay switch for transmission across the WAN.
- receive frame relay frames from the WAN, strip the frame relay frame off each frame producing the original LAN frame, and forward it to the end device.

Running-config

The configuration controlling the current operation of a piece of Cisco hardware. This may be different to the start-up config if changes have been made since start-up and the changes have not been saved. The running-config can be saved as the startup-config replacing any previous start-up config. The running config is held in DRAM. If the machine is restarted without the running-config being saved, all changes are lost.

Sample Interval

In Entuity the period between two data samples. This may be between two pollings of a port, or between two rolled up data samples.

SCR (Sustainable Cell Rate)

SCR is the long term data throughput of an ATM port. Traffic can burst above this limit up to the PCR.

Server

Any computer whose function in a network is to provide user access to files, printing, communications, and other services. Servers usually have more memory, more disk storage, and a more advanced processor than a single-user desktop PC.

Where Entuity manages an application, Entuity can manage the application server as a device.

Services

Services is a method of grouping together collections of ports that provide a service and associating with them other ports which use that service. For example, a service maybe e-mail, with one port designated as the provider of the service and all others in the group defined as consumers.

SLA

A Service Level Agreement (SLA) is a set of rules and metrics which can be used to measure the efficiency and performance of an object. That object may be a department, a server, a

network or any other functional component of an organization. If an object adheres to its associated set of rules and metrics, then it can be said to be conforming to its SLA. Similarly, if the object breaches the set of rules and metrics, then this means that it is no longer conforming to its SLA.

SNMP

Standardized method of managing and monitoring network devices on TCP/IP based internets. SNMP defines the formats of a set of network management messages, and the rules by which those messages are exchanged. The network management messages are used to make requests for performing network management functions and to report on events that occur in the network. Also, SNMP defines the allowable data types for MIBs, they way in which MIBs can be structured, and a set of standard objects that can be used in implementing a MIB.

Spanning Tree

Spanning tree provides a vendor neutral technology for visibility into your network. When correctly implemented Entuity discovers bridge links, switch to switch relationships, through polling the Bridge MIB. Complete spanning tree connectivity relies on a contiguous set of Entuity managed devices.

Spare Ports

By default Entuity spare port calculations include ports that have been unused for forty days or more, include ports that have system uptime of less than forty days and are currently unused and exclude ports that have been unused for less than forty days but have a system uptime of forty days or more.

By default Entuity spare port calculations:

- Include ports that have been unused for forty days or more.
- Include ports that have system uptime of less than forty days and are currently unused.
- Exclude ports that have been unused for less than forty days but have a system uptime of forty days or more.

The forty day threshold is configurable through the reporting section of entuity.cfg. Entuity distinguishes between physical and virtual ports using interface type. If required System Administrators can amend the virtual port identifier.

SNMP Agent

Management code that resides in the device, controls the operation of the device, and responds to SNMP requests.

SSL

An SSL Certificate consists of a public key and a private key. The public key is used to encrypt information and the private key is used to decipher it. When a browser points to a

secured domain, an SSL handshake authenticates the server and the client and establishes an encryption method and a unique session key. They can begin a secure session that guarantees message privacy and message integrity.

Startup-config

The initial configuration when a piece of Cisco hardware starts-up. If there have been no changes to the configuration since start-up, this will be the same as the running-config. The startup-config is also referred to as the saved config. The startup-config is held in NVRAM.

Static Thresholds

Static threshold settings allow you to configure the trigger points which when crossed cause Entuity to raise events. You can set thresholds against an individual event, a managed object, view or all objects on an Entuity server.

StormWorks

Entuity Configurable Framework is the internal Entuity engine, also known as the Data Management Kernel (DMK). It runs as the ***DsKernelStatic*** process.Entuity Configurable Framework enables the delivery of functionality through a highly configurable set of core services. The configuration files, found in *entuity_home*\etc, prefixed with ***sw_*** define and configure Entuity Configurable Framework services.

Entuity assigns all of the objects it manages their own Entuity Configurable Framework identifier. Entuity Configurable Framework identifiers are sequentially assigned, do not consider the object type and are unique within each Entuity server. *StormWorks ID* is visible from the object's web UI Advanced tab, and is used in creating dashboards to the user, for example during Data Export, Map Export, running of Flex Reports.

Stream Attributes

Information Entuity collects from your network is stored within Entuity as an attribute of the managed object, for example a port's name, a port's utilization are stored as attributes. Stream attributes are to maintain a history of a metric, for example Entuity maintains a history of port utilization.

SVG

Scalable Vector Graphics (SVG) is a graphics file format and Web development language based on XML. SVG is used by Entuity's reports to dynamically generate, high-quality graphics from real-time data.

Switch

A switch is a network device that selects a path or circuit for sending a unit of data to its next destination. It is usually simpler and faster than a router, which requires knowledge about the network to determine the route.

A switch may also include the function of the router, a device or program that can determine the route and specifically what adjacent network point the data should be sent to.

SynOptics Network Management Protocol (SONMP)

SONMP is also known as the Nortel Discovery Protocol (NDP), a Data Link Layer network protocol for discovery of Nortel (Avaya and Ciena) devices. It is available as a technology link type for the Entuity maps.

System Capabilities

Entuity determines the switching capability of a device by checking the group dot1dtp, specifically the mandatory scalar value dot1dTpLearnedEntryDiscards. dot1dtp is only present when the device supports transparent bridging, which implies it has Ethernet switching capability.

Entuity determines the routing capability of a device by checking for the ip-forwarding variable from the ip group in the MIB of the device. When ip-forwarding has a value of 1, this implies the device is acting as a gateway and so has routing capability.

Entuity determines whether the device type is hub by comparing its type to device types detailed in the vendor files.

TCP

Connection-oriented protocol that provides a reliable byte stream over IP. A reliable connection means that each end of the session is guaranteed to receive all of the data transmitted by the other end of the connection, in the same order that it was originally transmitted without receiving duplicates.

TCP/IP

Combination of TCP and IP protocols common to many different computer systems and so often used for communication between them.

TFTP

Trivial File Transfer Protocol (TFTP) is a very simple file transfer protocol, with the functionality of a very basic form of FTP. It uses UDP as its transport protocol and has no authentication or encryption mechanisms.

Ticker

Ticker allows you to view real time output at the device and port level, viewing data changes as they occur. You can select to view data activity for one or more client devices or ports.

For monitored:

- Ports you can select from a list of MIB variables the particular variable(s) you want to use to monitor the port. Entuity is supplied with a default number of MIB variables for use with ports and you can also add your own MIB variables to this list.
- Devices you can create your own list of MIB variables on which to monitor the device.

traceroute

Entuity includes two types of traceroute functionality, identified in the Entuity client as TraceRoute from Client and TraceRoute from Server.

TraceRoute from Entuity Client, calls the traceroute utility installed on the Entuity client machine and performs a live traceroute from the Entuity client to the target IP address.

TraceRoute from Entuity Server, uses data collected by `applicationMonitor`. This traceroute information is updated every two minutes, so calling TraceRoute from Server does not initiate a live traceroute but instead interrogates the data returned from the last `applicationMonitor` traceroute.

`applicationMonitor` uses Entuity's own implementation of traceroute functionality. This implementation performs ICMP pings in a similar way to a standard traceroute but with this key difference. When performing a traceroute `applicationMonitor` increments TTL values by one, until the pings reach the edge of an invisible cloud. At this point `applicationMonitor` increase the TTL value to 32. When this results in the ping reaching its target, the response from the target includes the actual number of hops required to reach target.

Traps

Traps can be used by network components to signal abnormal conditions. Entuity can both receive and forward SNMP traps.

Entuity can be configured to:

- Generate events in Event Viewer then traps are received.
- Forward traps to up to six concurrent recipients.

Entuity also supply a more advanced SNMP trap forwarding integration module. Contact your Entuity sales representative for details.

Trivial Change

A difference between a current-configuration file and a previously archived one that is not considered important by the system because it matches a set of rules codified as patterns in an "ignore file". Trivial changes may include comments such as timestamps in a configuration file.

Root Cause Analysis (RCA)

RCA isolates IT related problems using vector differencing. This involves the building of a dependency chain of objects and monitoring the object states in that chain. In the event of

state changes (where each object state change is a vector), differencing the dependency chain state vectors enables Entuity to determine the true cause of the event. Entuity can then raise the appropriate event.

For example, if an application becomes unavailable because a switch has failed then Entuity raises an event relating to the switch failure in Event Viewer. Entuity does not raise events for the application being unavailable as changes in state in the dependency chain are attributed to the switch failure.

Trunk Ports

Trunk ports, i.e. ports connecting switches together.

Entuity identifies a trunk port by:

- reading the MIB.
- ***macman*** identifying the switch port as having more than ten MAC addresses and also having associated VLANs.
- using CDP Trunk Port Discovery, a CISCO proprietary method.

When one or more of these methods identifies a trunk port, Entuity also considers it as a trunk port.

Unclassified Devices

Entuity managed devices for which Entuity does not have a device support dataset, provided through individual vendor, bin.vendor or newbin.vendor files, are included to Entuity as Unclassified devices under Full Management, or Unclassified devices under Ping-only and Basic Management.

Unclassified generically managed devices use an Uncertified device type, created by Entuity and held in newbin.vendor. These are Entuity managed devices and do incur a license charge. System Administrators should contact their Entuity support representative for a vendor file which would ensure Entuity fully manages these devices.

Unicast

Unicast is network communication between a single sender and a single receiver.

Uplink Detection

Entuity considers an uplink as trunking on a connection to a router or layer 3 switch, which is visible through spanning tree. This technology attempts to link layer 3 with layer 2.

Where links between switches and routers are not done using VLAN trunking and spanning tree then the spanning tree technology will not detect them. This is typically at smaller satellite offices, which do not need the greater port density and much greater speed available from router on a stick and even greater speed available from layer 3 switching.

Uplinks

Ports connecting routers with switches.

Uptime

By default Entuity polls devices every five minutes, retrieving device *sysuptime*. Entuity checks as to whether the device has been continually up since the last poll, and modifies the device's uptime value accordingly.

When *sysuptime* indicates the device has been down during the polling interval but is now up, from *sysuptime* alone Entuity cannot identify for how long the device was down. Entuity takes this unknown time, and adds fifty percent of it to the known uptime value, with the remaining fifty percent considered UNKNOWN. For example where *sysuptime* has a value of two minutes. Entuity cannot determine the state of the device over the first three minutes of the polling interval. Entuity adds ninety seconds to the *sysuptime* value, giving an uptime value of two hundred and ten seconds and records the device state as UNKNOWN for ninety seconds.

Device uptime is visible through Component Viewer, and is used in many reports, e.g. Routing Summary, Switching Summary.

Utilization

In Entuity port utilization is expressed as a percentage of actual traffic volume against the maximum volume that can be handled by the port.

UUID (Universally Unique ID)

A 16 byte value written to a system's planar at manufacturing time to uniquely identify a system across time and space.

Variable Binding

A variable binding, or VarBind, refers to the pairing of the name of a MIB variable to the variable's value. A VarBindList is a simple list of variable names and corresponding values. Some PDUs are concerned only with the name of a variable and not its value (e.g., the GetRequest-PDU). In this case, the value portion of the binding is ignored by the protocol entity. However, the value portion must still have valid ASN.1 syntax and encoding. It is recommended that the ASN.1 value NULL be used for the value portion of such bindings.

VCC (Virtual Channel Connection)

A VCC is an association established at the ATM Layer between two or more endpoints for the purpose of user-user, user-network, or network-network information transfer. The points at which the ATM cell payload is passed to the AAL for processing signify the endpoints of a VCC. Virtual Circuit is a more generic, non-ATM specific term.

VCI (Virtual Channel Identifier)

VPI and VCI together identify a virtual channel link on an ATM interface.

Vendor Files

Entuity identifies the device type of discovered devices by matching their sysoid to that held against the device support datasets. Device support dataset definitions are held in, listed here in order of precedence, individual vendor files, bin.vendor file, newbin.vendor file, and then uncertified file.

vendinfo identifies the vendor information available to Entuity and the decisions made when more than one vendor file is available for a particular sysoid; which vendor device definition Entuity uses to manage that device type.

File Type	Description
individual vendor files	When Entuity does not currently manage a device that you require it to, you can request your Entuity support representative for an appropriate vendor file. Those non-standard definitions are listed in entuity_home/etc/exotica. Only when a vendor file is moved to entuity_home/etc does Entuity use that definition.
bin.vendor file	File includes the default vendor file definition
newbin.vendor file	File includes device type definitions generated by earlier versions of Entuity.
uncertified file	File includes device type definitions created by Entuity, using proliferate with the -g parameter. Devices of this type are considered as Unclassified Devices.

View

All network objects within Entuity are displayed through views. View filters allow you to restrict the displayed objects in the view to the ones you are interested in. You can also use user profiles to control access to different views.

Virtual Channel Links (VCLs)

A VCC consists of the concatenated VCLs. A VCL is a means of unidirectional transport of ATM cells between the points where a VCI value is assigned and the point where the value is translated or removed. The VPI and VCI within the ATM cell header associates each cell with a particular VCL over a given physical link.

Virtual Circuit

A Virtual Circuit is a generic term for an association established between two or more endpoints for the purpose of user-user, user-network, or network-network information transfer. An example would be ATM's VCC.

Virtual Port

Entuity distinguishes between physical and virtual ports using interface type. If required System Administrators can amend the virtual port identifier.

VLAN

A logical association that allows users to communicate as if they were physically connected to a single LAN, independent of the actual physical configuration of the network.

VM Platforms

Entuity currently manages Oracle and VMware VMs through its VM Platform device type. Entuity communicates with VMs and their hypervisors through the VM's SDK. This requires specification of different connection attributes when compared to devices of other types. It also requires that all VMs are added to Entuity with a **Ping Only** management level, as this allows the selection of the VM Platform type and its connection configuration. When adding VMs using autoDiscovery care must be taken to ensure candidate device VMs are always added as **Ping Only**.

VPD (Vital Product Data)

VPD is information about a device that is stored on a computer's hard disk (or the device itself) that allows the device to be administered at a system or network level. Typical VPD information includes a product model number, a unique serial number, product release level, maintenance level, and other information specific to the device type. Vital product data can also include user-defined information, such as the building and department location of the device. The collection and use of vital product data allows the status of a network or computer system to be understood and service provided more quickly.

VPI (Virtual Path Identifier)

VPI identifies a virtual path leg on an ATM interface.

VRF (Virtual Routing and Forwarding)

VRF allows multiple instances of a routing table to co-exist within the same router at the same time. Because the routing instances are independent, the same or overlapping IP addresses can be used without conflicting with each other.

VTP (VLAN Trunk Protocol) Domain

A VTP domain consists of one or more connected switches that share the same VTP domain name. A switch can be configured to be in one and only one VTP domain. The vtpDomainTool generates a view that groups devices and VLANS by this VTP domain name.

Wireless Controller (WC)

A network attached device that coordinates traffic to and from Lightweight Wireless Access Points (LAPs). It provides centralized control over the configuration and dynamic behavior of potentially many LAPs.

Index

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Y

Z