NOZOMI NETWORKS

N2OS Configuration
Reference Guide

N2OS v26.6.0 — 2026-11-05

Legal notices

Information about the Nozomi Networks copyright and use of third-party software in the Nozomi Networks product suite.

Copyright

Copyright © 2013-2026, Nozomi Networks. All rights reserved. Nozomi Networks believes the information it furnishes to be accurate and reliable. However, Nozomi Networks assumes no responsibility for the use of this information, nor any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent, copyright, or other intellectual property right of Nozomi Networks except as specifically described by applicable user licenses. Nozomi Networks reserves the right to change specifications at any time without notice.

Third Party Software

Nozomi Networks uses third-party software, the usage of which is governed by the applicable license agreements from each of the software vendors. Additional details about used third-party software can be found at https://security.nozominetworks.com/licenses.

Contents

Chapter 1. Features **7**
Features 9
General 10
Retention 12
Chapter 2. Edit sensor configuration **17**
Edit sensor configuration 19
Chapter 3. Basic rules **21**
Basic configuration rules 23
Chapter 4. Garbage Collector **35**
Configure Garbage Collector 37
Chapter 5. Alerts **41**
Configure alerts 43
SIGN:MULTIPLE-ACCESS-DENIED 47
SIGN:MULTIPLE-UNSUCCESSFUL-LOGINS 48
SIGN:OUTBOUND-CONNECTIONS 49
SIGN:PROGRAM:CHANGE 52
SIGN:TCP-FLOOD 53
SIGN:UDP-FLOOD 56
SIGN:NETWORK-SCAN 57
Chapter 6. Incidents **65**
INCIDENT:PORT-SCAN 67
Chapter 7. Nodes **69**
Configure nodes 71
Chapter 8. Assets **77**
Configure assets 79
Chapter 9. Links **81**
Configure links 83
Chapter 10. Variables **89**
Configure variables 91
Chapter 11. Network **97**
Ports 99
Port configuration 99
Configure ports 102
Routes 106
Route configuration 106

Configure static routes....107
Chapter 12. Protocols....109
Configure protocols....111
Chapter 13. Vulnerability assessment....129
Configure vulnerability assessments....131
Chapter 14. Smart Polling....135
Configure Smart Polling....137
Chapter 15. Discovery....139
Configure Discovery....141
Chapter 16. Monitored subnets....143
Configure monitored subnets....145
Chapter 17. Customize node identifier generation....147
Customize node identifier generation....149
Chapter 18. Decryption....151
Overview....153
IEC 60870-5-7 / 62351-3/5....154
Overview....154
Configuration....155
Chapter 19. Trace....157
Configure trace....159
Configure continuous trace....162
Chapter 20. Time Machine....165
Configure Time Machine....167
Chapter 21. Retention....169
Configure retention....171
Chapter 22. Bandwidth throttling....179
Configure bandwidth throttling....181
Chapter 23. Synchronization....183
Configure synchronization....185
Troubleshooting....191
Chapter 24. Slow updates....195
Configure slow updates....197
Chapter 25. Session hijacking....199
Configure session hijacking protection....201
Chapter 26. Passwords....203
Configure passwords....205
Set maximum attempts....206

Enable pbkdf2 sha512 password hashing algorithm....207
Set lock time....208
Set history....209
Set password digits....210
Set password lower....211
Set password upper....212
Set password symbol....213
Set password min length....214
Set password max length....215
Enable expire for inactive users....216
Set user lifetime....217
Enable expiration for admin users....218
Enable password expiration....219
Set password lifetime....220
Chapter 27. Sandbox....221
Sandbox archive processing....223
Set the number of workers....224
Set the size of the sandbox tmpfs partition....225
Set the size of the sandbox tmpfs temporary partition....226
Set the size of the sandbox tmpfs pipes partition....227
Set sandbox strategies for vulnerability assessment....228
Set the minimum percentage of free disk....229
Set the maximum number of files to retain....230
Set the size of the asynchronous processing queues....231
Set the interval for stale file analysis....232
Set the maximum number of extracted files....233
Set the maximum level of nested extracted files....234
Set the maximum size of a single extracted file....235
Set the maximum overall size of extracted files....236
Set the maximum time to be spent for each file extraction....237
Define maximum recursion depth allowed for VBA extraction....238
Enable or disable the adaptive algorithm for file unzipping....239
Configuring handling of sandboxed zipped files....240
Configure high throughput protection....241
Chapter 28. Miscellaneous....245
Configure SSH key update interval....247
Configure paranoid mode for user login authentication....248
Configure identity provider....249
Chapter 29. N2OS Maintenance....251
Hardware information....253
Show CPU information....253
Show disk information....254
Show RAM information....255

Show motherboard information....256
Show network interface information....257
Show PSU information....258
Glossary....259

Chapter 1. Features



Features

*The **Features** page shows an overview of the current status of system features configuration and lets you fine tune specific values.*

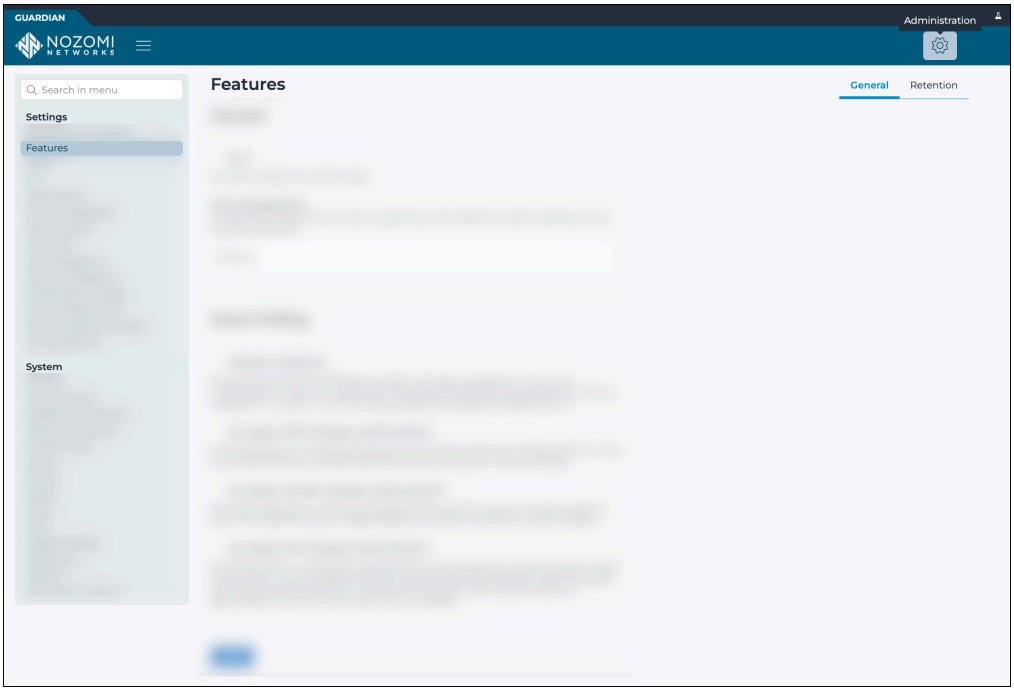


Figure 1. Features page

The **Features** page has these tabs:

- General (on page 10)
- Retention (on page 12)

General

*The **General** page lets you configure the behavior of a sensor.*

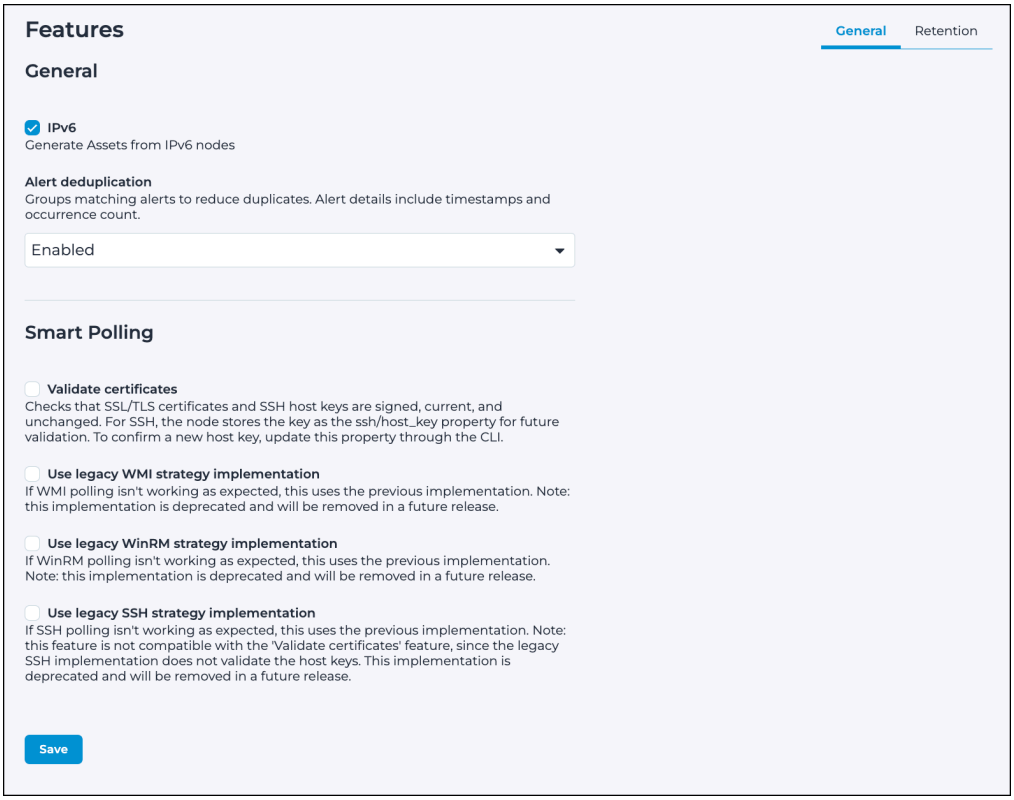


Figure 2. General page

IPv6

This checkbox enables the creation of assets with an *Internet Protocol version 6* address that are detected through passive monitoring.

Alert deduplication

This control lets you turn the deduplication of alerts on or off. When deduplication is enabled, Guardian groups identical events into a single alert instead of creating individual alerts for each event. This simplifies alert management and provides better visibility of anomalies.

By default, this feature is turned off. The default will change to enabled in v26.5.0. Leaving the status set to **Default** will result in the feature being turned on automatically when v26.5.0 is installed.

Validate certificates

By default, Smart Polling does not check that the / certificate is valid when connecting to an server, nor does it verify that the host key of an *secure shell (SSH)* server is known. You can enable these security checks here.

Smart Polling uses the address to contact servers. For certificate verification to succeed, the certificate must contain the server's address among the valid names.

For *SSH*, the host key is compared against the content of the `ssh/host_key` property of the polled node.

Use legacy WMI strategy implementation

If you find issues with your WMI Smart Polling plans that did not occur before, reverting to the legacy implementation of this strategy may resolve them. Contact technical support if this is the case.

Use legacy WinRM strategy implementation

If you find issues with your Smart Polling plans that did not occur before, reverting to the legacy implementation of this strategy may resolve them. Contact technical support if this is the case.

Use legacy SSH strategy implementation

If you find issues with your *SSH* Smart Polling plans that did not occur before, reverting to the legacy implementation of this strategy may resolve them. Contact technical support if this is the case.



Note:

The legacy implementation will not verify the *SSH* host key even if **Validate certificates** is turned on.

Retention

*The **Retention** tab lets you configure how much historical data Guardian stores for each data category.*

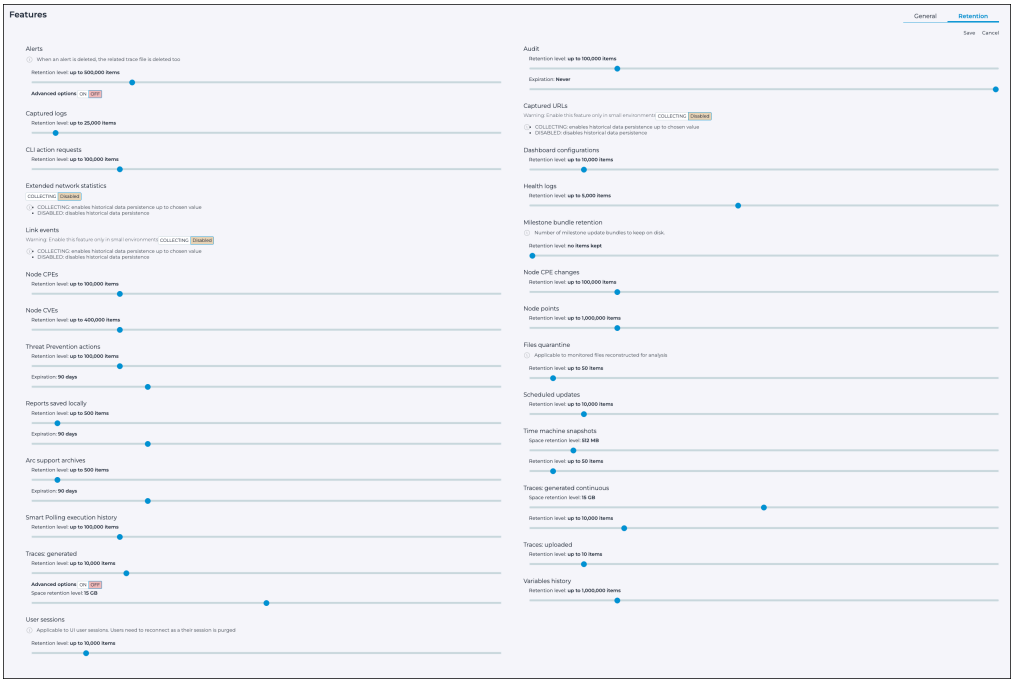


Figure 3. Retention tab

The **Retention** tab displays a set of data category cards in a two-column layout. Each card shows one or more sliders you can adjust to control how much data Guardian stores. Some categories also have a **COLLECTING** / **DISABLED** toggle, and some offer **Advanced options** that let you set per-subcategory limits within the overall total.

Expiration

Sets the maximum age of stored items in days. When an item exceeds this age, the system deletes it. Set to **Never** to keep items indefinitely.

Retention level

Sets the maximum number of items to store. When the limit is reached, the system deletes the oldest items to make room for new ones.

Space retention level

Sets the maximum disk space to use for storing items. When the limit is reached, the system deletes the oldest items.

Discard rate

Sets the maximum rate of incoming items per minute. The system discards new items when they exceed this threshold.

Data categories

The following data categories appear on the **Retention** tab. Categories that require optional features only appear when you have licensed and enabled those features on this sensor.

Data category	Controls	Notes
Left column (top to bottom)		
Alerts	Retention level	Deleting an alert also deletes the related trace file. Advanced options let you set a separate sub-limit for alerts outside the active security profile.
Captured logs	Retention level	
CLI action requests	Retention level	
Extended network statistics	Space retention level	Can be toggled on or off.
Link events	Retention level, Discard rate	Can be toggled on or off. Enable only in small environments.
Node CPE changes	Retention level	
Node points	Retention level	
Reports saved locally	Retention level, Expiration	
Arc support archives	Retention level, Expiration	
Smart Polling execution history	Retention level	
Threat Prevention actions	Retention level, Expiration	Tracks device actions taken from an alert (for example, blocking or unblocking a USB device), including their status.

Data category	Controls	Notes
Traces: generated	Retention level, Space retention level	Advanced options let you set sub-limits for traces generated by high risk alerts, medium risk alerts, low risk alerts, and by user request. The space retention level control is not available on systems where traces use tmpfs-based storage.
User sessions	Retention level	Applies to UI user sessions only. Users must reconnect when their session is purged.
Right column (top to bottom)		
Audit	Retention level, Expiration	
Captured URLs	Retention level, Discard rate	Can be toggled on or off. Enable only in small environments.
Dashboard configurations	Retention level	
Health logs	Retention level	
Milestone bundle retention	Retention level	Controls how many milestone update bundles are kept on disk. Set to 0 to disable retention (no bundles are kept).
Node CPEs	Retention level	
Node CVEs	Retention level	Requires Vulnerability Assessment.
Files quarantine	Retention level	Applies to monitored files reconstructed for analysis.
Scheduled updates	Retention level	
Time machine snapshots	Space retention level, Retention level	
Traces: generated continuous	Space retention level, Retention level	
Traces: uploaded	Retention level	

Data category	Controls	Notes
Variables history	Retention level	



Chapter 2. Edit sensor configuration



Edit sensor configuration

In Central Management Console (CMC) and Guardian sensors, use the command-line Interface (CLI) to configure the Nozomi Networks solution.

You can access the *command-line interface (CLI)* in two ways:

- use the `cli` command in a text-console when connected to the sensor, either directly or through *SSH*
- in the web GUI, select `Administration > Settings > CLI`

Examples:

A command issued via the `cli` command in a shell:

```
root@ch-lab-sg-guardian-stable:~ # cli
> conf.user configure vi gc old_ghost_nodes 3600
{"result"=>true}
success
```

A command issued (through pipe) to the `cli` command in a shell:

```
root@ch-lab-sg-guardian-stable:~ # echo conf.user configure vi gc
 old_ghost_nodes 3600 | cli
> {"result"=>true}
success
>root@ch-lab-sg-guardian-stable:~ #
```

There are cases, on Remote Collector sensors for example, where `cli` command doesn't work; in those cases or to fine-tune user-defined configuration or mass-import rules from other systems it's required to manually edit the `/data/cfg/n2os.conf.user`. In this section we will see how to change and apply a configuration rule.

Please log into the shell console, either through *SSH*, or shell console, and issue the following commands.

- Use vi or nano to edit `/data/cfg/n2os.conf.user`
- Edit a configuration rule with the text editor, see the next sections for some examples
- Write configuration changes to disk and exit the text editor

Next sections cover all the necessary details about the supported configuration rules.



Chapter 3. Basic rules



Basic configuration rules

Set traffic filter

Product	Guardian
Syntax	`conf.user configure bpf_filter <bpf_expression>`
Description	Set the *Berkeley Packet Filter (BPF)* filter to apply on incoming traffic to limit the type and amount of data processed by the sensor.
Parameters	`bpf_expression`: the Berkeley Packet Filter expression to apply on incoming traffic. A *BPF* syntax reference can be accessed on the sensor at `https://<sensor_ip>/#/bpf_guide`
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Monitor traffic on network interfaces

Product	Guardian
Syntax	`traffic_stop_monitor <list_of_network_interfaces>`
Description	Enables the network traffic monitor. When a monitored network interface stops receiving traffic, Guardian adds an entry to the health log.
Parameters	`list_of_network_interfaces`: a comma-separated list of network interfaces, for example `eth0,eth1`
Where	**CLI**
To apply	In a shell console, use vi or nano to edit the `/data/cfg/n2os.conf.user` file, add the configuration line, then execute: `service n2osids stop`
Note	Guardian generates the health log entry after about one minute without packets on the interface. An interface that never received any packet does not trigger an entry.

Enable or disable management filters

Product	Guardian
Syntax	`conf.user configure mgmt_filters [on|off]`

Description	With this rule you can switch off the filters on packets that come from/ to N2OS itself. Choose 'off' if you want to disable the management filters (default: on).
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Enable or disable TCP/UDP deduplication

Product	Guardian
Syntax	`conf.user configure probe deduplication enabled [true|false]`
Description	It can enable or disable the deduplication analysis that N2OS does on TCP/ UDP packets. it can be either true, to enable the feature, or false, to disable it. (default: true)
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set TCP deduplication time delta

Product	Guardian
Syntax	`conf.user configure probe deduplication tcp_max_delta <delta>`
Description	Set the desired maximum time delta, in milliseconds, to consider a duplicated TCP packet.
Parameters	`delta`: The value of the maximum time delta (default: 1)
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set UDP deduplication time delta

Product	Guardian
Syntax	`conf.user configure probe deduplication udp_max_delta <delta>`
Description	Set the desired maximum time delta, in milliseconds, to consider a duplicated UDP packet.
Parameters	`delta`: The value of the maximum time delta (default: 1)

Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Rename fallback zones

Product	Guardian
Syntax	`conf.user configure vi zones default [private|public] <zone_name>`
Description	Set the *private* or *public* fallback zone name, for nodes not matching any zone. Details on zones feature can be viewed in the **Graph** page in **Network**.**Remark**: zones can be configured through the GUI, which is the preferred way. Refer to **Zone configurations**, in the **Administrator Guide**.
Parameters	`zone_name`: the name of the *private* or *public* fallback zone
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Add Zone

Product	Guardian
Syntax	`ids configure vi zones create <subnet>[,<subnet>] <zone_name>`
Description	Add a new zone containing all the nodes in one or more specified subnetworks. More subnetworks can be concatenated using commas. The subnetworks can be specified using the CIDR notation (`<ip>/<mask>`) or by indicating the end IPs of a range (both ends are included: `<low_ip>-<high_ip>`). **Remark**: zones can be configured through the GUI, which is the preferred way. Refer to **Zone configurations**, in the **Administrator Guide**.
Parameters	• `subnet`: The subnetwork or subnetworks assigned to the zone; both IPv4 and IPv6 are supported • `zone_name`: The name of the zone
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Assign a level to a zone

Product	Guardian
Syntax	`ids configure vi zones setlevel <level> <zone_name>`
Description	Assigns the specified level to a zone. All nodes pertaining to the given zone will be assigned the level. **Remark**: zones can be configured through the GUI, which is the preferred way. Refer to **Zone configurations**, in the **Administrator Guide**
Parameters	• `level`: The level assigned to the zone • `zone_name`: The name of the zone
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Set the nodes ownership for a zone

Product	Guardian
Syntax	`ids configure vi zones setis_public [true|false] <zone_name>`
Description	Sets the specified nodes ownership for a zone. It can be either true, for public ownership, or false, for private ownership. All nodes belonging to the given zone are overwritten inheriting the value. **Remark**: zones can be configured through the GUI, which is the preferred way. Refer to **Zone configurations**, in the **Administrator Guide**
Parameters	`zone_name`: The name of the zone
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Assign a security profile to a zone

Product	Guardian
Syntax	`ids configure vi zones setsecprofile [low|medium|high|paranoid] <zone_name>`

Description	Assigns the specified security profile to a zone. The visibility of the alerts generated within the zone will follow the configured security profile. Refer to **Security profile**, in the **Administrator Guide**.
Parameters	`zone_name`: The name of the zone
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Add custom protocol

Product	Guardian
Syntax	`conf.user configure probe custom-protocol <name> [tcp|udp] <port>`
Description	Add a new protocol specifying a port and a transport layer. Names shall always be unique, so when defining a custom protocol both for udp and tcp, use two different names.
Parameters	• `name`: The name of the protocol, it will be displayed through the user interface; **DO NOT** use a protocol name already used by N2OS. E.g. one can use MySNMP, or Myhttp • `port`: The transport layer port used to identify the custom protocol
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Disabling a protocol

Product	Guardian
Syntax	`conf.user configure probe protocol <name> enable false`
Description	Completely disables a protocol. This can be useful to fine tune the sensor for specific needs.
Parameters	`name`: The name of the protocol to disable
Where	**CLI**
To apply	It is applied automatically

Set MAC grouping

Product	Guardian
Syntax	`conf.user configure probe macgroup <pattern> <mac>`
Description	This command assigns a new MAC number, defined by macgroup, to all nodes whose original MAC number matches a specified pattern. The original MAC numbers are not retained.
Parameters	`pattern`: The MAC address pattern to be matched. The pattern defines either a fully specified MAC address (e.g., 00:11:22:33:44:55), or a MAC address prefix (e.g., 00:11:22:*). Prefixes can be defined only at the byte level, finer grained patterns (e.g., 00:1*) are not supported and must be expressed via multiple configuration rules. `mac`: The MAC address to be used whenever the MAC address pattern is matched.
Where	**CLI**
To apply	In a shell console, execute both: `service n2osids stop` AND `service n2ostrace stop`

Set IP grouping

Product	Guardian
Syntax	`conf.user configure probe ipgroup <ip>/<mask>`
Description	This command permits to group multiple ip addresses into one single node. This command is particularly useful when a large network of clients accesses the SCADA/ICS system. To provide a clearer view and get an effective learning phase, you can map all clients to a unique node simply by specifying the netmasks (one line for each netmask). Configure trace (on page 159) will still show the raw IPs in the provided trace files. **Warning**: This command merges all nodes information into one in an irreversible way, and the information about original nodes is not kept.
Parameters	`ip`/`mask`: The subnetwork identifier used to group the IP addresses
Where	**CLI**
To apply	In a shell console, execute both: `service n2osids stop` AND `service n2ostrace stop`

Set IP grouping for Public Nodes

Product	Guardian

Syntax	`conf.user configure probe ipgroup public_ips <ip>`
Description	This command permits to group all public IP addresses into one single node (for instance, use 0.0.0.0 as the 'ip' parameter). This command is particularly useful when the monitored network includes nodes that have routing to the Internet. The Configure trace (on page 159), will still show the raw IPs in the provided trace files. **Warning**: This command merges all nodes information into one in an irreversible way, and the information about original nodes is not kept.
Parameters	`ip`: The ip to map all Public Nodes to
Where	**CLI**
To apply	In a shell console, execute both: `service n2osids stop` AND `service n2ostrace stop`

Skip Public Nodes Grouping for a subnet

Product	Guardian
Syntax	`conf.user configure probe ipgroup public_ips_skip <ip>/<mask>`
Description	This is useful when the monitored network has a public addressing that has to be monitored (i.e. public addressing used as private or public addresses that are in security denylists).
Parameters	`ip/mask`: The subnetwork identifier to skip
Where	**CLI**
To apply	In a shell console, execute both: `service n2osids stop` AND `service n2ostrace stop`

Set special Private Nodes allowlist

Product	Guardian
Syntax	`conf.user configure vi private_ips <ip>/<mask>`
Description	This rule will set the is_public property of nodes matching the provided mask to false. This is useful when the monitored network has a public addressing used as private (e.g. violation of RFC 1918).
Parameters	`ip/mask`: The subnetwork identifier to treat as private; both IPv4 and IPv6 are supported

Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set GUI logout timeout

Products	CMC, Guardian
Syntax	`conf.user configure users max_idle_minutes <timeout_in_minutes>`
Description	Change the default inactivity timeout of the GUI. This timeout is used to decide when to log out the current session when the user is not active.
Parameters	`timeout_in_minutes`: amount of minutes to wait before logging out. The default is 10 minutes.
Where	**CLI**
To apply	It is applied automatically

Enable Syslog capture feature

Product	Guardian
Syntax	`conf.user configure probe protocol syslog capture_logs [true|false]`
Description	With this configuration rule you can enable (option true) the passively capture of the syslog events. It is useful when you want to forward them to a SIEM, for more details see **Syslog forwarder integration** in the **Administrator Guide**.
Where	**CLI**
To apply	It is applied automatically

Enable Guardian HA

Product	Guardian
Syntax	`conf.user configure guardian replica-of <other_guardian_id>`

Description	With this configuration rule you can enable the Guardian HA mode for two Guardians that sniff the same traffic and are connected to the same CMC. During normal operations, only the primary Guardian syncs with the CMC; if it stops synchronizing the secondary Guardian will start synchronize the records from the last primary Guardian update. This rule should only be configured on the secondary sensor.
Parameters	`other_guardian_id`: The id of the other Guardian, it can be found on the CMC with the query `appliances | where host == <appliance_hostname> | select id`
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Disabling Vulnerability Assessment for some nodes

Product	Guardian
Syntax	`conf.user configure va_notification matching [id|label|zone|type|vendor]=<value> discard`
Description	With this configuration rule you can disable Vulnerability Assessment for node matching the specified rules. The effect of this configuration rule is to discard the matching of CVE identifiers. The types are as follows. • `id`: the id of a node, it can be an IP address, a netmask in the CIDR format or a MAC address. • `label`: the label of a node. • `zone`: the zone in which a node is located. • `type`: the type of a node. • `vendor`: the vendor of a node.
Parameters	• `value`: If a simple string is specified the match will be performed with an "equal to" case-sensitive criterion. The matching supports two operators: • ^: starts with • '[': contains These operators must be specified right after the = symbol and their match is case-insensitive. Examples: • `va_notification matching id=192.168.1.123 discard` • `va_notification matching id=192.168.1.0/24 discard` • `va_notification matching label=^abc discard`
Where	**CLI**
To apply	In a shell console execute: `service n2osva stop`

Enabling IPv6 Assets

Product	Guardian
Syntax	`conf.user configure vi ipv6_assets [enabled|disabled]`
Description	With this configuration rule you can enable assets generation also when nodes are IPv6. By default, this feature is disabled.
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Change the maximum percentage of Variables in the Network Elements pool

Product	Guardian
Syntax	`conf.user configure vi machine_limits_variables_quota <n>`
Description	With this configuration rule you can change the maximum percentage of Variables in the Network Elements pool, the default is 0.6 meaning that no more than 60% of Network Elements can be Variables.
Parameters	`n`: the percentage of variables expressed as a number from 0.0 to 1.0, e.g. `vi machine_limits_variables_quota 0.7`
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Tuning Backend Web Server workers

Products	CMC, Guardian
Syntax	`conf.user configure http_workers <n>`
Description	With this configuration rule you can change the number of Ruby Web Server workers. With a higher workers count the CMC/Guardian can handle more Web UI requests concurrently, at the expense of increased memory footprint.
Parameters	`n`: The new number of workers
Where	**CLI**
To apply	In a shell console execute: `service webserver stop`

Configure which files detected on the networks are sent to sandbox

Product	Guardian
Syntax	**conf.user configure vi sandbox_extraction <json_value>**
Description	The json object can have the following attributes: * **disabled_protocols** - A JSON array of protocol that are disabled with regards files detection * **enabled_protocols** - A JSON array of protocol that are enabled with regards files detection * **disabled_file_extensions** - A JSON array of file extensions that are disabled with regards files detection * **enabled_file_extensions** - A JSON array of file extensions that are enabled with regards files detection
Parameters	• **json_value**: A json object to configure the handling of the detected files.
Where	**CLI**
To apply	In a shell console execute: **service n2osids stop**



Chapter 4. Garbage Collector



Configure Garbage Collector

Configure the Garbage Collector (GC) to discard nodes, assets, and links that are no longer useful, saving system resources.

Clean up old ghost nodes

Product	Guardian
Syntax	`conf.user configure vi gc old_ghost_nodes <seconds>`
Description	Set the threshold after which idle nodes that are also not confirmed and not learned are discarded by the . **Note:** in Adaptive Learning, the works also if nodes are learned, since they all are. **Note:** The will not remove nodes that Smart Polling created.
Parameters	`seconds`: Number of seconds after which cleanup occurs (the default is 3600, the equivalent of one hour).
Where	**CLI**
To apply	It is applied automatically

Clean up old public nodes

Product	Guardian
Syntax	`conf.user configure vi gc old_public_nodes <seconds>`
Description	Determines how long to keep public nodes that are inactive. Expressed in seconds.
Parameters	`seconds`: Number of seconds after which cleanup occurs (default is 259200, the equivalent of three days).
Where	**CLI**
To apply	It is applied automatically

Clean up old inactive nodes

Product	Guardian
Syntax	`conf.user configure vi gc old_inactive_nodes <seconds>`

Description	Determines how long to keep nodes that are inactive. Expressed in seconds. Inactivity is calculated as the difference between the current time and the last activity time. **Note:** When a node that has been deleted by the appears again in the network it will be considered new, as a consequence, according to the learning mode, an alert could be raised. For a better result, use Adaptive Learning and choose a reasonably long interval for this setting. In some cases, such as during certain data integrations or when data originates from Smart Polling, `last_activity_time` may be set to 0. In these situations, garbage collection uses the `created_at` time instead. **Note:** The will not remove nodes that Smart Polling created.
Parameters	`seconds`: Number of seconds after which cleanup occurs (by default it's disabled).
Where	**CLI**
To apply	It is applied automatically

Clean up old inactive links

Product	Guardian
Syntax	`conf.user configure vi gc old_inactive_links <seconds>`
Description	Determines how long to keep links that are inactive. Expressed in seconds. Inactivity is calculated as the difference between the current time and the last activity time. **Note:** When a link that has been deleted by the appears again in the network it will be considered new, as a consequence, according to the learning mode, an alert could be raised. For a better result, use Adaptive Learning and choose a reasonably long interval for this setting. In some cases, such as during certain data integrations or when data originates from Smart Polling, `last_activity_time` may be set to 0. In these situations, garbage collection uses the `created_at` time instead.
Parameters	`seconds`: Number of seconds after which cleanup occurs (by default it's disabled).
Where	**CLI**
To apply	It is applied automatically

Clean up old ghost links

Product	Guardian
Syntax	`conf.user configure vi gc old_ghost_links <seconds>`

Description	Determines how long to wait before removing inactive ghost links. A ghost link is one that has not shown any application payload since its creation. This could be a connection attempt whose endpoint is not responding on the specified port; or it could be a link with a successful handshake but without application data transmitted (in this case, transferred data would still be greater than 0).
Parameters	`seconds`: Number of seconds after which cleanup occurs (by default it's disabled).
Where	**CLI**
To apply	It is applied automatically

Clean up old inactive variables

Product	Guardian
Syntax	`conf.user configure vi gc old_inactive_variables <seconds>`
Description	Determines how long to keep variables that are inactive. Expressed in seconds. Inactivity is calculated as the difference between the current time and the last activity time. **Note:** When a *variable* that has been deleted by the appears again in the network it will be considered new, as a consequence, according to the learning mode, an alert could be raised. For a better result, use Adaptive Learning and choose a reasonably long interval for this setting.
Parameters	`seconds`: Number of seconds after which cleanup occurs (by default it's disabled).
Where	**CLI**
To apply	It is applied automatically

Clean up old sessions

Product	Guardian
Syntax	`conf.user configure vi gc sessions_may_expire_after <seconds>`
Description	Determines how long to wait before a session is considered stale and it's resources may be collected. Expressed in seconds.
Parameters	`seconds`: Number of seconds after which clean up may occur. By default, set to 100 seconds.
Where	**CLI**

To apply	It is applied automatically

Clean up old nodes that have private MAC addresses

Product	Guardian
Syntax	`conf.user configure vi gc nodes_with_private_mac <seconds>`
Description	Determines how long to keep inactive nodes that have private addresses, expressed in seconds.
Parameters	`<seconds>`: Number of seconds after which clean up occurs (by default it's disabled).
Where	**CLI**
To apply	It is applied automatically

Preserve nodes with imported fields and manual edits

Product	Guardian
Syntax	`conf.user configure vi gc keep_imported_nodes <value>`
Description	Determines whether nodes having imported fields may be considered for deletion. By default they are not (so, explicitly, if a node has a field that was imported or manually edited, it won't be automatically garbage-collected).
Parameters	If `<value>` is `0`, the rule is de-activated. If `<value>` is `1`, the rule is activated (the default behavior).
Where	**CLI**
To apply	It is applied automatically.

Chapter 5. Alerts



Configure alerts

Configure maximum number of victims

Product	Guardian
Syntax	`conf.user configure alerts max_victims <num>`
Description	Define the maximum number of victims that each alert can contain. Victims exceeding the given value are not stored. Default value is 10.
Parameters	`num`: Maximum number of victims stored for each alert
Where	**CLI**
To apply	It is applied automatically

Configure maximum number of attackers

Product	Guardian
Syntax	`conf.user configure alerts max_attackers <num>`
Description	Define the maximum number of attackers that each alert can contain. Attackers exceeding the given value are not stored. Default value is 10.
Parameters	`num`: Maximum number of attackers stored for each alert
Where	**CLI**
To apply	It is applied automatically

Show/hide credentials

Product	Guardian
Syntax	`conf.user configure alerts hide_username_on_alerts [true|false]`
Syntax	`conf.user configure alerts hide_password_on_alerts [true|false]`
Description	These flags determine whether usernames or passwords should be presented in the alert. By default, the credentials are visible. Affected alert types: `SIGN:MULTIPLE-ACCESS-DENIED`, `SIGN:MULTIPLE-UNSUCCESSFUL-LOGINS`, `SIGN:CLEARTEXT-PASSWORD`, `SIGN:PASSWORD:WEAK`.

Where	**CLI**
To apply	It is applied automatically

Configure maximum length of description

Product	Guardian
Syntax	`conf.user configure alerts max_description_length <nchars>`
Description	Define the maximum number of characters that can contain the description of each incident. When an incident is appendding an alert description, the append is performed only if the incident description length is smaller then the limit.
Parameters	`nchars`: Maximum number of characters allowed in the description of each incident
Where	**CLI**
To apply	It is applied automatically

Configure MITRE ATT&CK mapping rules

Product	Guardian
Syntax for MITRE ATT&CK for ICS mappings	`conf.user configure alerts mitre_attack ics_mapping <path>`
Syntax for MITRE ATT&CK Enterprise mappings	`conf.user configure alerts mitre_attack enterprise_mapping <path>`
Description	Customize the rules used to assign MITRE ATT&CK techniques to alerts by means of an external file. The file has the following format. Each line defines a rule; the rule specifies an alert type ID followed by a semicolon and a comma-separated list of MITRE ATT&CK technique IDs. For instance, the line `SIGN:PROGRAM:TRANSFER;T0843,T0853` instructs Guardian that alerts of type `SIGN:PROGRAM:TRANSFER` must be assigned both the `T0843` and `T0853` MITRE ATT&CK techniques.
Parameters	`path`: The path to the file containing the rules
Where	**CLI**

To apply	It is applied automatically

Enable storing of alerts not visible under the current security profiles

Product	Guardian
Syntax	`conf.user configure alerts save_invisible_alerts [true|false]`
Description	Alerts are not stored into the database, if they are not visible under the current security profile and they are not part of an incident. This command can change this behavior and allow the above alerts to be stored.
Where	**CLI**
To apply	In a shell console execute: `service n2osalert stop`

Configure red assertions behavior

Product	Guardian
Syntax	`conf.user configure assertion_element_monitoring [enabled|disabled]`
Description	By default, an assertion is boolean; it is either red or green, and upon turning red the assertion may send alerts. This configuration allows the assertions to actively monitor lists of elements (such as nodes, labels, etc.), and send new alerts for any additional elements that break the assertion, and when an alert is closed and the triggering asserted element is still present. Disabled by default.
Where	**CLI**
To apply	It is applied automatically

Configure packet rules alerts limit

Product	Guardian
Syntax	`conf.user configure packet_rules alerts_per_session [limit]`
Description	Alerts risen by packet rules have a limit per alert class and per session This limit, which is 1 by default, can be relaxed or removed entirely. A value of -1 means unlimited. A value of 0 will completely disable these alerts.
Where	**CLI**

To apply	It is applied automatically

SIGN:MULTIPLE-ACCESS-DENIED

In this section we will configure the Multiple Access Denied alert.

The detection is enabled by default and works accordingly to the following parameters.

Set interval and threshold - 1

Product	Guardian
Syntax	`conf.user configure vi multiple_events protocol <protocol> <interval> <threshold>`
Description	Set the detection configuration for a specific protocol.
Parameters	• `protocol`: Name of the protocol to configure. Can be 'all' to apply the configuration globally. • `interval`: maximum time in seconds for the event to happen in order to trigger the detection. Default: 30[s] for OT devices, 15[s] for the rest." • `threshold`: number of times for the event to happen in order to trigger the detection. Default: 20 for OT devices, 40 for the rest.
Where	**CLI**
To apply	It is applied automatically

For example, we can configure the detection of a multiple access denied alert for the SMB protocol with an interval of 10 seconds and threshold of 35 attempts with the following command:

```
conf.user configure vi multiple_events protocol smb 10 35
```

SIGN:MULTIPLE-UNSUCCESSFUL-LOGINS

In this section we will configure the Multiple Unsuccessful Logins alert.

The detection is enabled by default and works accordingly to the following parameters.

Set interval and threshold - 2

Product	Guardian
Syntax	`conf.user configure vi multiple_events protocol <protocol> <interval> <threshold>`
Description	Set the detection configuration for a specific protocol.
Parameters	• `protocol`: Name of the protocol to configure. Can be 'all' to apply the configuration globally. • `interval`: maximum time in seconds for the event to happen in order to trigger the detection. Default: 30[s] for OT devices, 15[s] for the rest." • `threshold`: number of times for the event to happen in order to trigger the detection. Default: 20 for OT devices, 40 for the rest.
Where	**CLI**
To apply	It is applied automatically

For example, we can configure the detection of a multiple unsuccessful login alert for the SMB protocol with an interval of 10 seconds and threshold of 35 attempts with the following command:

```
conf.user configure vi multiple_events protocol smb 10 35
```

SIGN:OUTBOUND-CONNECTIONS

In this section we will configure the outbound connections limit.

Guardian can detect a sudden increase of outbound connections from a specific learned source node. An alert is raised by default when 100 new outbound connections are observed over a 60-seconds interval.

By default, the detection is only performed when the node is being protected. Optionally, the detection can also be performed when the node is being learned.

Optionally, we can prevent the system from creating additional destination nodes in order to preserve resources. Such nodes creation limit is disabled by default.

Some of the configuration parameters listed below can be applied either globally or to individual nodes. The configuration of an individual node has higher priority and overrides the global configuration.

Perform detection when source node is being learned

Product	Guardian
Syntax	`conf.user configure vi outbound_connections_limit learning [true|false]`
Description	Specify whether the detection has to be performed also when the source node is being learned or only when it is being protected. Select `true` for detection also when the source node is learned, or `false` for detection only when the source node is being protected. By default `false`.
Where	**CLI**
To apply	It is applied automatically

Enable/disable nodes creation limit

Product	Guardian
Syntax global	`conf.user configure vi outbound_connections_limit enabled [true|false]`
Syntax individual node	`conf.user configure vi node <ip> outbound_connections_limit enabled [true|false]`
Description	Enable (option `true`) or disable (option `false`) the destination nodes creation limit.
Parameters	`ip`: The IP of the source node
Where	**CLI**

To apply	It is applied automatically

Set connections count

Product	Guardian
Syntax global	`conf.user configure vi outbound_connections_limit connections <count>`
Syntax individual node	`conf.user configure vi node <ip> outbound_connections_limit connections <count>`
Description	Set the outbound connections limit, in number of connections.
Parameters	• `ip`: The IP of the source node • `count`: The amount of outbound connections from a node to be observed in order to trigger the detection (default: 100)
Where	**CLI**
To apply	It is applied automatically

Set observation interval

Product	Guardian
Syntax global	`conf.user configure vi outbound_connections_limit interval <value>`
Syntax individual node	`conf.user configure vi node <ip> outbound_connections_limit interval <value>`
Description	Set the outbound connections observation interval, in seconds.
Parameters	• `ip`: The IP of the source node • `value`: The time interval during which the new outbound connections are observed.
Where	**CLI**
To apply	It is applied automatically

For example, we can configure the outbound connections limit to prevent a source node from creating additional destination nodes when 70 outbound connections are observed during a 30-seconds interval with the following configuration commands:

```
conf.user configure vi outbound_connections_limit enabled true
conf.user configure vi outbound_connections_limit connections 70
conf.user configure vi outbound_connections_limit interval 30
```

SIGN:PROGRAM:CHANGE

In this section we will configure the Program change alert.

The detection is enabled by default and works according to the following parameters.

Set the maximum steps number for ethernet-ip online edits

Product	Guardian
Syntax	`conf.user configure probe protocol ethernetip online_edits_max_steps <n_steps>`
Description	When online edits are detected in ethernet-ip traffic, this is the maximum number of steps for the alert to be emitted (even if no commit has been seen yet).
Parameters	`n_steps`: The maximum number of steps for the alert to be emitted (default: 10).
Where	**CLI**
To apply	In a shell console execute: `n2os_ids_stop`

SIGN:TCP-FLOOD

This section describes the TCP flood detection configuration.

> **Note:**
> The configurations described here apply to all TCP FLOOD detections except:
> - the detection for TCP flooding through FIN packets,
> - the detection for TCP flooding through packets with no TCP flags.
>
> These detections raise an alert when 1000 such packets are seen within 1 second, and are not configurable.

A node is considered to be under a TCP flood attack when:

- The number of SYN-ACK packets during the observation interval is greater than the detection counter threshold, and
- during the observation interval, the ratio between ACK packets and SYN-ACK packets falls below the trigger threshold

A TCP flood attack is considered terminated when:

- The number of SYN-ACK packets during the observation interval returns below the detection counter threshold, or
- during the observation interval, the ratio between between ACK packets and SYN-ACK packets returns above the exit threshold

The detection of flooding is not guarded by the packets duplication detection. In other words, duplicated packets can still trigger a flooding alert. This is because the detection of duplication is based on SYN numbers, which do not change during a flooding event; deduplicating these packets will cause false negatives as it will be inhibiting the flooding detection on duplicate packets.

Set detection counter

Product	Guardian
Syntax	`conf.user configure vi tcp_syn_flood_detection counter <value>`
Description	Set the SYN-ACK packets counter threshold, in number of connections.
Parameters	`value`: The amount of SYN-ACK packets to be observed in order to trigger the detection (default: 100)
Where	**CLI**
To apply	It is applied automatically

Set observation interval

Product	Guardian
Syntax	`conf.user configure vi tcp_syn_flood_detection interval <value>`
Description	Set the observation interval, in seconds.
Parameters	`value`: The time interval during which the count of SYN-ACK packets and ACK packets are observed, in seconds (default: 10).
Where	**CLI**
To apply	It is applied automatically

Set trigger threshold

Product	Guardian
Syntax	`conf.user configure vi tcp_syn_flood_detection trigger_threshold <value>`
Description	Set the trigger threshold.
Parameters	`value`: The ratio between ACK packets and SYN-ACK packets, which when it is reached triggers the flood detection (default: 0.1).
Where	**CLI**
To apply	It is applied automatically

Set exit threshold

Product	Guardian
Syntax	`conf.user configure vi tcp_syn_flood_detection exit_threshold <value>`
Description	Set the exit threshold.
Parameters	`value`: The ratio between ACK packets and SYN-ACK packets, which when it is reached terminates the flood detection (default: 0.4).
Where	**CLI**
To apply	It is applied automatically

For example, with the commands below the TCP flood detection would trigger when 200 SYN-ACK packets are observed during a 15-seconds observation interval and the ratio between ACK packets and SYN-ACK packets falls below 0.3. Then the detection would terminate when the ratio returns above 0.5.

```
conf.user configure vi tcp_syn_flood_detection counter 200
conf.user configure vi tcp_syn_flood_detection interval 15
conf.user configure vi tcp_syn_flood_detection trigger_threshold 0.3
conf.user configure vi tcp_syn_flood_detection exit_threshold 0.5
```

SIGN:UDP-FLOOD

In this section we will configure the UDP flood detection.

The detection is enabled by default and it triggers when a victim receives 20'000 UDP packets per second for at least 10 seconds.

Enable/disable detection

Product	Guardian
Syntax	`conf.user configure vi udp_flood_detection enabled [true|false]`
Description	Enable (option `true`) or disable (option `false`) the UDP flood detection.
Where	**CLI**
To apply	It is applied automatically

Set detection threshold

Product	Guardian
Syntax	`conf.user configure vi udp_flood_detection packets_per_second <threshold>`
Description	Set the UDP flood detection threshold, in packets per second.
Parameters	`threshold`: The amount of UDP packets per second to be transmitted to a victim for at least 10 seconds in order to trigger the detection (default: 20000)
Where	**CLI**
To apply	It is applied automatically

For example, we can configure the UDP flood detection to trigger when a victim receives 40'000 UDP packets per second for at least 10 seconds with the following configuration command:

```
vi udp_flood_detection packets_per_second 40000
```

SIGN:NETWORK-SCAN

DDOS Defense

In this section we will configure the detection of a DDOS attack.

The detection is enabled by default and an alert is raised at most every 5 minutes, when under one minute more than 20 nodes have been created.

Set analysis interval

Product	Guardian
Syntax	`conf.user configure vi ddos_defense interval <threshold>`
Description	Set the analysis interval for the detection.
Parameters	`threshold`: The analysis interval is measured in minutes. Default: one minute.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set max created nodes

Product	Guardian
Syntax	`conf.user configure vi ddos_defense max_created_nodes <max_nodes>`
Description	Number of created nodes that, if created in less time than the analysis interval, will trigger the alert.
Parameters	`max_nodes`: Number of created nodes that trigger the detection. Default: 20.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set alert threshold

Product	Guardian
Syntax	`conf.user configure vi ddos_defense alert_threshold <threshold>`
Description	Interval to wait in order to raise an additional alert.

Parameters	`threshold`: Minutes to wait for another alert to be raised. Default: 5 minutes.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set alert threshold

Product	Guardian
Syntax	`conf.user configure vi ddos_defense alert_threshold <threshold>`
Description	Interval to wait in order to raise an additional alert.
Parameters	`threshold`: Minutes to wait for another alert to be raised. Default: 5 minutes.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

TCP Port Scan

In this section we will configure the detection for the TCP Port scan.

The detection is enabled by default and an alert is emitted according to the configuration parameters described below.

Set attempts threshold

Product	Guardian
Syntax	`conf.user configure vi port_scan_tcp attempts_threshold <threshold>`
Description	Set the number of scan attempts that will trigger the alert.
Parameters	`threshold`: Number of scan attempts that will trigger the alert. Default: 100.
Where	**CLI**
To apply	It is applied automatically

Set observation interval

Product	Guardian

Syntax	`conf.user configure vi port_scan_tcp interval <interval>`
Description	Set the analysis interval for the detection algorithm.
Parameters	`interval`: Analysis interval in seconds for the detection algorithm. Default: 10 seconds.
Where	**CLI**
To apply	It is applied automatically

Set trigger threshold

Product	Guardian
Syntax	`conf.user configure vi port_scan_tcp trigger_threshold <threshold>`
Description	Set the trigger threshold for the detection algorithm. An alert is raised only if the ratio between the number of established connections and total attempts is smaller than the trigger threshold.
Parameters	`threshold`: Trigger threshold as described above for the detection algorithm. Default: 0.1.
Where	**CLI**
To apply	It is applied automatically

Set out of sequence threshold

Product	Guardian
Syntax	`conf.user configure vi port_scan_tcp out_of_sequence_threshold_number <threshold>`
Description	Set the number of out of sync fragments which trigger this feature of the detection algorithm.
Parameters	`threshold`: Number of out of sync fragments. Default: 10.
Where	**CLI**
To apply	It is applied automatically

Set out of sequence interval

Product	Guardian

Syntax	`conf.user configure vi port_scan_tcp out_of_sequence_interval <interval>`
Description	Set the analysis interval of the out of sync recognition feature of the detection algorithm.
Parameters	`interval`: Analysis interval in seconds. Default: 10 seconds.
Where	**CLI**
To apply	It is applied automatically

Set out of sequence max rate

Product	Guardian
Syntax	`conf.user configure vi port_scan_tcp out_of_sequence_threshold_max_rate <rate>`
Description	Set the period of time during which additional alerts due to out of sync fragments are not raised.
Parameters	`rate`: Timespan in minutes to mute additional alerts due to ouf of sync fragments. Default: 5 minutes.
Where	**CLI**
To apply	It is applied automatically

Set ignored port ranges

Product	Guardian
Syntax	`conf.user configure vi port_scan_tcp ignore_ports <port_ranges>[,<port_ranges>]`
Description	Set the victims' ports or port ranges which must not participate in the detection algorithm.
Parameters	`port_ranges`: ports can be entered as a list of comma separated values and ranges as a pair of ports separated by a dash. Example: 1000,1200-1300,1500. Default: none.
Where	**CLI**
To apply	It is applied automatically

For example, we can configure the detection for the TCP Port scan with the following commands:

```
      conf.user configure vi port_scan_tcp attempts_threshold 50
      conf.user configure vi port_scan_tcp interval 20
      conf.user configure vi port_scan_tcp trigger_threshold 0.2
      conf.user configure vi port_scan_tcp
out_of_sequence_threshold_number 15
      conf.user configure vi port_scan_tcp out_of_sequence_interval 20
      conf.user configure vi port_scan_tcp
out_of_sequence_threshold_max_rate 10
      conf.user configure vi port_scan_tcp ignore_ports
1000,1200-1300,1500
```

UDP Port Scan

In this section we will configure the detection for the UDP Port scan.

The detection is enabled by default and an alert is emitted according to the configuration parameters described below.

Set fast threshold

Product	Guardian
Syntax	`conf.user configure vi port_scan_udp fast_threshold <threshold>`
Description	Set the number of attempts which will trigger the alert for the fast detection algorithm.
Parameters	`threshold`: Attempts triggering the alert for the fast detection algorithm. Default: 500.
Where	**CLI**
To apply	It is applied automatically

Set slow interval

Product	Guardian
Syntax	`conf.user configure vi port_scan_udp slow_interval <interval>`
Description	Set the analysis interval for the slow detection algorithm.
Parameters	`interval`: Analysis interval for the slow detection algorithm. Default: 60 seconds.

Where	**CLI**
To apply	It is applied automatically

Set fast interval

Product	Guardian
Syntax	`conf.user configure vi port_scan_udp fast_interval <interval>`
Description	Set the analysis interval for the fast detection algorithm.
Parameters	`interval`: Analysis interval for the fast detection algorithm. Default: 1 second.
Where	**CLI**
To apply	It is applied automatically

Set fast different ports threshold

Product	Guardian
Syntax	`conf.user configure vi port_scan_udp fast_different_ports_threshold <threshold>`
Description	Set the number of different ports that should be tested by the attacker for the fast detection algorithm to trigger the alert.
Parameters	`threshold`: Minimum number of different ports to be tested by the attacker to trigger the alert for the fast detection algorithm. Default: 250.
Where	**CLI**
To apply	It is applied automatically

Set unreachable ratio

Product	Guardian
Syntax	`conf.user configure vi port_scan_udp unreachable_ratio <ratio>`
Description	The slow detection algorithm will issue an alert only if the ratio between the number of unreachable requests and the total requests is greater than this value.

Parameters	• `ratio`: Critical ratio for the slow detection algorithm to trigger an alert. An alert is raised if the ratio between the number of unreachable requests and the total requests is greater than the critical ratio. Default: 0.1.
Where	**CLI**
To apply	It is applied automatically

For example, we can configure the detection for the UDP Port scan with the following commands:

```
      conf.user configure vi port_scan_udp slow_threshold 200
      conf.user configure vi port_scan_udp slow_interval 30
      conf.user configure vi port_scan_udp fast_threshold 400
      conf.user configure vi port_scan_udp
fast_different_ports_threshold 150
      conf.user configure vi port_scan_udp fast_interval 3
      conf.user configure vi port_scan_udp unreachable_ratio 0.2
```

Ping Sweep

In this section we will configure the detection for the ICMP/Ping Sweep scan.

The detection is enabled by default and an alert is emitted when more than 100 request are issued in less than 5 seconds with a total number of recorded victims equal to 100.

Set request number

Product	Guardian
Syntax	`conf.user configure vi ping_sweep max_requests <threshold>`
Description	Set the number of requests that will trigger the alert.
Parameters	`threshold`: Number of request that will raise the alert. Default: 100.
Where	**CLI**
To apply	It is applied automatically

Set interval

Product	Guardian
Syntax	`conf.user configure vi ping_sweep interval <interval>`

Description	Set the interval during which the maximum number of requests should be issued in order to trigger the alert.
Parameters	`interval`: Interval in seconds for the maximum requests to be issued. Default: 5 seconds.
Where	**CLI**
To apply	It is applied automatically

For example, we can configure the detection for the ICMP/Ping Sweep scan with an analysis interval of 10 seconds for a threshold of 200 requests with 150 victims recorded with the following commands:

```
conf.user configure vi ping_sweep max_requests 200
conf.user configure vi ping_sweep interval 10
```

Treck Stack

In this section we will configure the detection for the Treck TCP/IP Fingerprint scan via ICMP 165.

The detection is enabled by default and an alert is emitted at most once every 20 minutes.

Set alert interval

Product	Guardian
Syntax	`conf.user configure vi treck_stack once_every <threshold>`
Description	Set the minimum interval between two raised alerts, in minutes.
Parameters	`threshold`: Minutes to wait for another alert to be raised. Default: 20 minutes.
Where	**CLI**
To apply	It is applied automatically

For example, we can configure the detection for the Treck TCP/IP Fingerprint Scan via ICMP 165 with an interval between two emitted alerts of one hour (60 minutes) with the following command:

```
conf.user configure vi treck_stack once_every 60
```

Chapter 6. Incidents



INCIDENT:PORT-SCAN

Port scan incident

In this section we will configure the parameters of a Port Scan Incident.

The detection is enabled by default and an incident is raised when more than 6 correlated alerts are triggered, independently from their creation time.

For example, we can configure the parameters for the Port Scan Incident with the following command, where we identify the minimum number of alerts for the incident to be triggered, and the maximum time interval in milliseconds in which they need to occur:

```
     conf.user configure alerts incidents portscan {"min_alerts": 25,
"max_time_interval": 1500}
```

Configure the port scan incident

Product	Guardian
Syntax	`conf.user configure alerts incidents portscan <json_obj>`
Description	Configure the port scan incident by providing the configuration in a JSON object.
Parameters	`json_obj`: JSON object containing the keys 'min_alerts' and 'max_time_interval', which are respectively the minimum number of alerts which trigger the detection and the maximum time interval in which they need to occur.
Where	**CLI**
To apply	In a shell console execute: `service n2osalert stop`



Chapter 7. Nodes



Configure nodes

Set node label

Product	Guardian
Syntax set	`ids configure vi node <ip> label <label>`
Syntax erase	`ids configure vi node <ip> label`
Description	Set the label to a node, the label will appear in the `Graph`, in the `Nodes`, in the `Process > Variables`
Parameters	• `ip`: The IP address of the node • `label`: The label that will be displayed in the user interface
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Set default live traffic label formatting

Product	Guardian
Syntax	`conf.user configure vi default_node_live_label <operation>[:<param>][,<operation>[:<param>]]`
Description	The default formatting operation(s) applied to labels coming from live traffic. More operations can be applied sequentially. See also the "Set protocol-specific live traffic label formatting" configuration for details
Parameters	• `operation`: The operation to apply • `param`: The specific operation parameter
Where	**CLI**
To apply	It is applied automatically

Set protocol-specific live traffic label formatting

Product	Guardian
Syntax	`conf.user configure vi node_live_label <protocol> <operation>[:<param>][,<operation>[:<param>]]`

Description	Protocol-specific formatting operation(s) applied to labels coming from live traffic. More operations can be applied sequentially.
Parameters	`protocol`: The name of the protocol `operation`: The operation to apply. There are several operations categories: • Invalid character replace operations (the default `param` is `' '`): ◦ `utf8`: only printable utf8 characters ◦ `ascii`: only printable ascii characters ◦ `alnum`: only alphanumeric characters `a-zA-Z0-9` ◦ `alnum_underscore`: only `alnum` + underscore • String operations:`prefix`: it keeps only the prefix of a string identified by param. The default `param` is `'.'` • Validation operations: ◦ `strict`: checks if the label is changed from it's first value or, if present, from the last `mark` operation. If changed, the label will be set to empty ◦ `mark`: resets the `strict` history to the current operation (the label will not be changed) `param`: The specific operation parameter as above. For an example, see the table below.
Where	**CLI**
To apply	It is applied automatically

Operations	Input label	Output label	Comment
`utf8:-`	lab¿1 ¿test¿.	lab-1 -test-.	The `utf8` operation replaces not allowed characters with the set parameter '-'
`alnum`	test1	test1	Unchanged because all characters are valid
`alnum`	lab,1	lab 1	The `alnum` operation replaces a not allowed character
`alnum,strict`	lab,1		The `strict` operation detects a change between the initial input 'lab,1' and the alnum output 'lab 1', the label is cleared
`alnum,mark,utf8,strict`	lab,1	lab 1	The `mark` operation sets `alnum` output (lab 1) as default, for the following `strict` operation. The `utf8` operation has no effect, and `strict` detects no changes so has no effect either

Operations	Input label	Output label	Comment
`prefix`	host_n.domain.tld	host_n	The `prefix` operation keeps only the hostname part

Set node Device ID with priority

Product	Guardian
Syntax	`ids configure vi node <ip> device_id_with_priority <device_id>;<priority>`
Description	Adds the Device ID to the set of node Device IDs. The final Device ID, used for node grouping under Assets is the one with the highest priority
Parameters	• `ip`: The IP address of the node • `device_id`: The device id • `priority`: the priority of the Device ID. If missing, it will be se to the lowest priority value
Where	**CLI**
To apply	It is applied automatically

Override node Device ID

Product	Guardian
Syntax	`ids configure vi node <ip> device_id_override <device_id>`
Description	Adds the Device ID to the set of node Device IDs, giving it the maximum priority value. This Device ID will be used for node grouping under Assets
Parameters	• `ip`: The IP address of the node • `device_id`: The device id (with the maximum priority)
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Enable or disable node

Product	Guardian

Syntax	`ids configure vi node <ip> state [enabled|disabled]`
Description	This directive permits to disable a node. This setting has effect in the graph: a disabled node will not be displayed.
Parameters	`ip`: The IP address of the node
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Enable or disable same ip node separation

Product	Guardian
Syntax	`conf.user configure check_multiple_macs_same_ip enable [true|false]`
Description	This directive permits to enable the separation of L3 nodes with same IP but different MAC address. The nodes with the desired IP addresses will be treated as L2 nodes and appear as distinct assets. If the nodes already exist as L3 nodes upon the application of the configuration, they will be deleted and the new logic will start to execute with empty statistics. The values of `true` or `false` enables, respectively disables, the feature.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Configure same ip node separation

Product	Guardian
Syntax	`conf.user configure check_multiple_macs_same_ip ip <ip_address>`
Description	Selects the ip of the nodes which should be separated as per the strategy described in the previous box.
Parameters	`ip_address`: The IP of the node to be configured
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Allow grouping into a single asset of nodes having the same IP address and distinct MAC addresses when check_multiple_macs_same_ip enable is set to true

Product	Guardian
Syntax	`conf.user configure check_multiple_macs_same_ip group_by_ip [true|false]`
Description	Groups nodes that share the same IP address but have different MAC addresses into a single asset. This option is intended for environments with static IP addresses only. Dynamic IP networks are out of scope.
Parameters	The values of `true` or `false` enables, or disables, the feature (default: `false`).
Where	**CLI**
To apply	It is applied automatically

Require confirmed MAC addresses to separate L3 nodes with same IP but different MAC address when check_multiple_macs_same_ip enable is set to true

Product	Guardian
Syntax	`conf.user configure check_multiple_macs_same_ip require_confirmed_macs [true|false]`
Description	When enabled, the separation of L3 nodes with the same IP address but different MAC addresses is allowed only for nodes with confirmed MAC addresses. This option is intended for environments with static IP addresses only. Dynamic IP networks are out of scope.
Parameters	The values of `true` or `false` enables, or disables, the feature (default: `false`).
Where	**CLI**
To apply	It is applied automatically

Delete node

Product	Guardian
Syntax	`ids configure vi node <ip> :delete`
Description	Delete a node from the environment

Parameters	`ip`: The IP of the node to delete
Where	**CLI**
To apply	It is applied automatically

Define a cluster

Product	Guardian
Syntax	`conf.user configure vi cluster <ip> <name>`
Description	This command permits to define an High Availability cluster of observed nodes. In particular, this permits to: accelerate the learning phase by joining the learning data of two sibling nodes, and to group nodes by cluster in the graph.
Parameters	• `ip`: The IP of the node • `name`: The name of the cluster
Where	**CLI**
To apply	It is applied automatically

Chapter 8. Assets



Configure assets

Hide built-in asset types

Product	Guardian
Syntax	`conf.user configure vi hide_built_in_asset_types true`
Description	Hides built-in asset types visible from the dropdown in the asset configuration modal
Where	**CLI**
To apply	It is applied automatically

Configure asset label extraction strategy

Product	Guardian
Syntax	`vi asset_field_extraction name {"default_strategy": <strategy>}`
Description	Lets you define the strategy to be used to obtain the label for an asset **Note:** This is useful in Emerson Ovation networks, where DROP names are defined at the L2 communication level, even though the most representative node within the asset is typically at the L3 level.
Parameters	`strategy`: It can be one of the following: • `"highest_quality"`: (Default) Takes the label from the most representative node, even if the node has an empty label • `"highest_quality_non_empty"`: Takes the label from the most representative node that does not have an empty label
Where	**Shell**
To apply	In a shell console execute: `service n2osids stop`



Chapter 9. Links



Configure links

Set link last activity check

Product	Guardian
Syntax set	`vi link <ip1> <ip2> <protocol> :check_last_activity <seconds>`
Syntax erase	`vi link <ip1> <ip2> <protocol> :check_last_activity :delete`
Description	Set the last activity check on a link, an alert will be raised if the link remains inactive for more than the specified seconds
Parameters	• `ip1`, `ip2`: The IPs of the two nodes involved in the communication • `protocol`: The protocol • `seconds`: The communication timeout
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Set link persistency check

Product	Guardian
Syntax	`vi link <ip1> <ip2> <protocol> :is_persistent [true|false]`
Description	Set the persistency check on a link, if a new handshake is detected an alert will be raised
Parameters	• `ip1`, `ip2`: The IPs of the two nodes involved in the communication • `protocol`: The protocol
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Set link alert on SYN

Product	Guardian
Syntax	`vi link <ip1> <ip2> <protocol> :alert_on_syn [true|false]`

Description	Raise an alert when a TCP SYN packet is detected on this link
Parameters	• `ip1`, `ip2`: The IPs of the two nodes involved in the communication • `protocol`: The protocol
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Set link to track availability

Product	Guardian
Syntax set	`vi link <ip1> <ip2> <protocol> :track_availability <seconds>`
Syntax erase	`vi link <ip1> <ip2> <protocol> :track_availability :delete`
Description	Notify the link events when the link communication is interrupted or resumed.
Parameters	• `ip1`, `ip2`: The IPs of the two nodes involved in the communication • `protocol`: The protocol • `seconds`: Interval to checking if the link is available or not
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Delete link

Product	Guardian
Syntax	`ids configure vi link <ip1> <ip2> :delete`
Description	Delete a link
Parameters	`ip1`, `ip2`: The IPs identifying the link
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Delete protocol

Product	Guardian
Syntax	`ids configure vi link <ip1> <ip2> <protocol> :delete`
Description	Delete a protocol from a link
Parameters	• `ip1`, `ip2`: The IPs identifying the link • `protocol`: The protocol of the link to delete
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Learn function code

Product	Guardian
Learn	`ids configure vi link <ip1> <ip2> <protocol> fc <func_code>`
Learn or Unlearn	`ids configure vi link <ip1> <ip2> <protocol> fc <func_code> is_learned [true|false]`
Description	Learn or unlearn a function code from a protocol
Parameters	• `ip1`, `ip2`: The IPs identifying the link • `protocol`: The protocol of the link • `func_code`: The function code to be learned or unlearned
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Delete function code

Product	Guardian
Syntax	`ids configure vi link <ip1> <ip2> <protocol> fc <func_code> :delete`
Description	Delete a function code from a protocol

Parameters	• `ip1`, `ip2`: The IPs identifying the link • `protocol`: The protocol of the link • `func_code`: The function code to be deleted
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Enable link_events generation

Product	Guardian
Syntax	`conf.user configure vi link_events [enabled|disabled]`
Description	Enable or disable the generation of link_events records, this feature can have an impact on performance, enable it carefully
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Disabling the persistence of links

Product	Guardian
Syntax	`conf.user configure vi persistence skip_links true`
Description	With this configuration rule you can disable the persistence of links, thus saving disk space in cases with a large number of links.
Where	**CLI**
To apply	It is applied automatically

Enable link ports collection for the specified set of protocols

Product	Guardian
Syntax	`conf.user configure vi enable_link_ports <protocol_name>[,<protocol_name>]`

Description	For enabled protocols, the set of source and destination ports found in the underlying sessions is collected and shown as links attributes (`from_ports` and `to_ports`). By default enabled only on unrecognized protocols (i.e. links named `other`). If the command is used, the total list of protocols to be enabled shall be specified (and so including `other` too if applicable).
Parameters	• `protocol_name`: The name of the protocol to enable
Where	**CLI**
To apply	It is applied automatically

Set max number of collected link ports

Product	Guardian
Syntax	`conf.user configure vi max_link_ports <max_collected_ports>`
Description	When the ports collection is enabled for links, sets the maximum number of collected ports for each link.
Parameters	`max_collected_ports`: The maximum number of collected ports for each link. Default 32. Max 128.
Where	**CLI**
To apply	It is applied automatically



Chapter 10. Variables



Configure variables

Enable or disable default variable history

Product	Guardian
Syntax	`ids configure vi variable default history [enabled|disabled]`
Description	Set if the variable history is enabled or not, when not set it's disabled. The amount of the history maintained can be configured in "Variable history retention" section in Configure retention (on page 171). **Note:** Enabling this functionality can negatively affect Guardian's performance, depending on the amount of variables and the update rate.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Enable or disable variable history

Product	Guardian
Syntax	`ids configure vi variable <var_key> history [enabled|disabled]`
Description	Define the amount of samples shown in the graphical history of a variable. Set if the variable history is enabled or not, when not set it's disabled. The amount of the history maintained can be configured in "Variable history retention" section in Configure retention (on page 171). **Note:** Enabling this functionality can negatively affect Guardian's performance, depending on the amount of variables and the update rate.
Parameters	• `var_key`: The variable identifier
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`
Note	You can also change this configuration from the Web UI.

Set variable label

Product	Guardian
Syntax	`ids configure vi variable <var_key> label <label>`
Description	Set the label for a variable, the label will appear in the `Process` sections

Parameters	• `var_key`: The variable identifier • `label`: The label displayed in the user interface
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`
Note	You can also change this configuration from the Web UI.

Set variable unit of measure

Product	Guardian
Syntax	`ids configure vi variable <var_key> unit <unit>`
Description	Set a unit of measure on a variable.
Parameters	• `var_key`: The variable identifier • `unit`: The unit of measure displayed in the user interface
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`
Note	You can also change this configuration from the Web UI.

Set variable offset

Product	Guardian
Syntax	`ids configure vi variable <var_key> offset <offset>`
Description	The offset of the variable that will be used to map the 0 value of the variable.
Parameters	• `var_key`: The variable identifier • `offset`: The offset value used to calculate the final value of the variable
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`
Note	You can also change this configuration from the Web UI.

Set variable scale

Product	Guardian
Syntax	`ids configure vi variable <var_key> scale <scale>`
Description	The scale of the variable that is used to define the full range of the variable.
Parameters	• `var_key`: The variable identifier • `scale`: the scale value used to calculate the final value of the variable
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`
Note	You can also change this configuration from the Web UI.

Set variable last update check

Product	Guardian
Syntax set	`vi variable <var_key> :check_last_update <seconds>`
Syntax remove	`vi variable <var_key> :check_last_update :delete`
Description	Set the last update check on a variable, if the variable value is not updated for more than the specified seconds an alert is raised
Parameters	• `var_key`: The variable identifier • `seconds`: The timeout after which a stale variable alert will be raised
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`
Note	You can also change this configuration from the Web UI.

Set variable quality check

Product	Guardian
Syntax set	`vi variable <var_key> :check_quality <seconds>`
Syntax remove	`vi variable <var_key> :check_quality :delete`
Description	Set the quality check on a variable, if the value quality remains invalid for more than the specified seconds an alert is raised

Parameters	• `var_key`: The variable identifier • `seconds`: The maximum amount of consecutive seconds the variable can have an invalid quality
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`
Note	You can also change this configuration from the Web UI.

Set variable to alert on quality

Product	Guardian
Syntax set	`vi variable <var_key> :alert_on_quality <quality>`
Syntax remove	`vi variable <var_key> :alert_on_quality :delete`
Description	Raise an alert when the variable has one of the specified qualities. Possible values are: invalid, not topical, blocked, substituted, overflow, reserved, questionable, out of range, bad reference, oscillatory, failure, inconsistent, inaccurate, test, alarm. Multiple values can be separated by comma.
Parameters	• `var_key`: The variable identifier • `quality`: The alert quality
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`
Note	You can also change this configuration from the Web UI.

Set a variable critical state

Product	Guardian
Syntax	`conf.user configure cs variable <id> <var_key> [<|>|=] <value>`
Description	Define a new custom critical state on a single variable that will raise on violation of defined range. For instance, if the > operator is specified, the variable will have to be higher than value to trigger the critical state.
Parameters	• `id`: A unique ID for this critical state • `var_key`: The variable identifier • `value`: The variable value to check for

Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set a multiple critical state

Product	Guardian
Syntax	`conf.user configure cs multi <id> variable <ci> <var_key> [<|>|=] <value>[ ^ variable <ci> <var_key> [<|>|=] <value>]`
Description	Creates a multi-valued critical state, that is an expression of "variable critical states", described above. The syntax is and AND (^) expression of the single-variable critical state.
Parameters	• `id`: A unique ID for this critical state • `ci`: Enumerate the variables `c1`, `c2`, `c3`, ..., etc • `var_key`: The variable identifier • `value`: The variable value to check for
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Control variables extraction at the protocol level

Product	Guardian
Syntax	`conf.user configure probe protocol <name> variables_extraction [disabled|enabled|advanced|global]`
Description	It allows the application of a variable extraction policy different from the global policy on a protocol basis. Note that if the Global policy is set to Disabled, it prevails on any protocol-specific setting. However, protocol specific policies prevail. Choices are whether variables extraction is `disabled`, `enabled`, enabled with advanced heuristics (`advanced`) or if it should inherit the global policy (`global`)
Parameters	• `name`: The name of the target protocol
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Control variables extraction at the global level for all zones

Product	Guardian
Syntax	`conf.user configure vi variables_extraction [disabled|enabled|advanced|global]`
Description	Same as for the protocol level variables extraction, except it sets the policy for the global level. Choices are whether variables extraction is `disabled`, `enabled`, enabled with advanced heuristics (`advanced`) or if it should inherit the global policy (`global`)
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Control variables extraction at the global level for specific zones

Product	Guardian
Syntax	`conf.user configure vi variables_extraction [disabled|enabled|advanced|global] <zones>`
Description	Same as for the protocol level variables extraction, except it sets the policy for the global level for the specified zones. Choices are whether variables extraction is `disabled`, `enabled`, enabled with advanced heuristics (`advanced`) or if it should inherit the global policy (`global`)
Parameters	• `zones`: Name of the zones for which the extraction should be enabled. If unspecified, the extraction is enabled for all the zones. and values are separated by a comma; for example: `[plant1,plant2]` or `[zone1,zone2,zone3]`. Brackets are required
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`
Note	You can also change this configuration from the Web UI.

Chapter 11. Network



Ports

Port configuration

A description of port types and their characteristics.

Primary management port

The primary management port:

- Is the primary interface for administrative access
- Must be the only one in the system
- Requires the default gateway
- Is intended for management purposes, with default restrictions in line with standard security policies
- Exposes:
    - Web on port 443
    - *SSH* access on port 22
- Can be used to connect to these sensors:
    - Arc
    - Guardian
    - Remote Collector

Management port(s)

A management port:

- Is a secondary interface for administrative access
- Permits multiple ports
- Requires no gateway
- Is intended for management purposes, with default restrictions in line with standard security policies
- Exposes:
    - Web on port 443
    - *SSH* access on port 22
- Can be used to connect to these sensors:
    - Arc
    - Guardian
    - Remote Collector

Active port(s)

An active port:

- Has an assigned address, used for operational data exchange
- Can be used:
    - To poll assets through Smart Polling
    - To communicate with the sensor through Arc endpoints
    - For communication for downstream sensors
    - As an ERSPAN target

- Cannot be used:
  - For Web access
  - For *SSH* access
- Exposes:
  - Port 443 and 6000 only for synchronization with downstream sensors
  - Port 6000 is enabled only when `n2os-enable-rc` is issued

Passive port(s)

A passive port:

- Is a port without an assigned
- Is intended to receive mirrored traffic, for example, SPAN

Supported formats

Table 1. IPv4 addresses

IP/CIDR format	`192.0.2.10/24`
Gateway format	`192.0.2.1`

Table 2. IPv6 addresses

IP/CIDR format	`2001:db8::10/64`
Gateway format	`2001:db8::1`
Compressed notation supported	`::1/128, fe80::1/64`

Important notes for the ports configuration

Note:
Reboot required: Most modifications require a system reboot to be applied.

Note:
Version compatibility: Validation: addresses and gateways must be of the same version (or *IPv6*).

Note:
Privileges: Most modification commands require administrative privileges.

Note:
Network interruption: Changes may temporarily interrupt network connectivity.

Note:
Confirmation required: The system will ask for confirmation before applying critical changes.

Common error messages

Error message	Resolution
`IP is required`	Specify an address with `--ip`.
`Default gateway IP is required`	Specify a default gateway with `--default-gateway`.
`Unknown port`	The specified port name does not exist.
`The IP is not in the required format`	Use a valid address or notation (for example, `192.0.2.10` or `192.0.2.0/24`).
`Gateway not in subnet`	The gateway must belong to the same subnet as the interface address.
`No management port found`	No management interface is currently configured.
`Port is not active`	You attempted to deactivate a port that is already passive.

Exit codes for the n2os-netconf commands

`0`: Success.

`1`: Error occurred (for example, invalid parameters, validation failed, or similar).

Configure ports

Assign roles to a sensor's network ports to create the desired configuration. The command applies only to the sensor where you run it and does not affect other sensors.

Show all current port configurations

Product	Guardian
Syntax	`n2os-netconf ports --show [--compact]`
Description	Shows the ports name and current configuration.
Parameters	`--compact`, Shows a compact version of the display without the **More Info** column.
Where	The text-console (access directly or through *SSH*)
To apply	Enter the command and press enter.

Identify the primary management port

Product	Guardian
Syntax	`n2os-netconf ports --get --mgmt-primary`
Description	Shows the name of the port that is currently defined as the primary management port.
Parameters	None
Where	The text-console (access directly or through *SSH*)
To apply	Enter the command and press enter.

Identify the IP address of the primary management port

Product	Guardian
Syntax	`n2os-netconf ports --get --mgmt-primary-ip`
Description	Shows the addresses of the primary management port.
Parameters	None
Where	The text-console (access directly or through *SSH*)
To apply	Enter the command and press enter.

Identify the management port

Product	Guardian
Syntax	`n2os-netconf ports --get --mgmt`
Description	Shows the names of the ports that are currently defined as management ports.
Parameters	None
Where	The text-console (access directly or through *SSH*)
To apply	Enter the command and press enter.

Identify the IP address of the management port

Product	Guardian
Syntax	`n2os-netconf ports --get --mgmt-ip`
Description	Shows the addresses of the management ports.
Parameters	None
Where	The text-console (access directly or through *SSH*)
To apply	Enter the command and press enter.

Set a port as the management port

Product	Guardian
Syntax	`n2os-netconf ports --change-mgmt PORT [--reboot-later]` **Important:** The port designated as the new management port must reside within the default gateway network.
Description	Transfers management configuration from one port to another, preserving address, gateway, and other settings. The port previously configured as management port will become a passive port.
Parameters	PORT, For example: `port1`
	`--reboot-later` Prevents automatic system reboot. User must manually reboot to apply changes.

Where	The text-console (access directly or through *SSH*). User must have root privileges. **All *SSH* connections use the current management port and will be lost after the change.** Do not run this command unless a serial connection, or a connection into the sensor through the new management port is available.
To apply	Enter the command and press enter. The sensor will reboot upon completion of the command to correctly apply this setting.

Set a port as a primary management port

Product	Guardian
Syntax	`n2os-netconf ports --set PORT --mgmt-primary --ip IP/CIDR --default-gateway IP [--reboot-later] [--wpa]`
Description	Sets the selected port to be the primary management port, assigning it an address and a subnet mask, and sets the default gateway to the specified address
Parameters	PORT, For example: `port1`
	IP/CIDR, For example: `192.167.1.15/24` or `2001:db8::10/64`
	IP, For example: `192.167.1.1` or `2001:db8::1`
	`--reboot-later` Prevents automatic system reboot. User must manually reboot to apply changes.
	`--wpa` Enables WPA authentication.
Where	The text-console (access directly or through *SSH*). User must have root privileges.
To apply	Enter the command and press enter.

Set a port as a secondary management port

Product	Guardian
Syntax	`n2os-netconf ports --set PORT --mgmt --ip IP/CIDR [--reboot-later] [--wpa]`
Description	Sets the selected port as a secondary management port assigning it an address and a subnet mask.
Parameters	PORT, For example: `port1`

	IP/CIDR, For example: `192.167.1.15/24` or `2001:db8::10/64`
	`--reboot-later` Prevents automatic system reboot. User must manually reboot to apply changes.
	`--wpa` Enables WPA authentication.
Where	The text-console (access directly or through *SSH*). User must have root privileges.
To apply	Enter the command and press enter.

Set a port as an active port

Product	Guardian
Syntax	`n2os-netconf ports --set PORT --active --ip IP/CIDR [--reboot-later] [--wpa]`
Description	Sets the selected port as an active port assigning it an address and a subnet mask.
Parameters	PORT, For example: `port1`
	IP/CIDR, For example: `192.167.1.15/24` or `2001:db8::10/64`
	`--reboot-later` Prevents automatic system reboot. User must manually reboot to apply changes.
	`--wpa` Enables WPA authentication.
Where	The text-console (access directly or through *SSH*). User must have root privileges.
To apply	Enter the command and press enter.

Set a port as a passive port

Product	Guardian
Syntax	`n2os-netconf ports --set PORT --passive [--reboot-later]`
Description	Sets the selected port as a passive port removing any addresses and subnet mask from it.
Parameters	PORT, For example: `port1`

	`--reboot-later` Prevents automatic system reboot. User must manually reboot to apply changes.
Where	The text-console (access directly or through *SSH*). User must have root privileges.
To apply	Enter the command and press enter.

Routes

Route configuration

An overview of static route configuration on the sensor.

Static routes direct network traffic to a specific destination through a defined gateway. On Guardian, static routes are configured using `n2os-netconf routes` commands from the text console. Routes are stored in the system configuration and applied automatically at boot.

You can configure two types of static routes:

- **Network routes**: Routes to a destination network, specified in CIDR notation
- **Host routes**: Routes to a single destination host address

Each route is identified by a name, which you use to update or delete it.

Configure static routes

Configure static routes on the sensor using the text console. Static routes are stored in the system configuration and applied automatically at boot. The command applies only to the sensor where you run it and does not affect other sensors.

Show all configured static routes

Product	Guardian
Syntax	`n2os-netconf routes --show`
Description	Shows all configured static routes, including the default gateway and any named static routes, with their type, destination, and gateway.
Parameters	None
Where	The text-console (access directly or through *SSH*)
To apply	Enter the command and press enter.

Add a network static route

Product	Guardian
Syntax	`n2os-netconf routes --set --name NAME --net NETWORK/CIDR --gateway IP`
Description	Adds or updates a static route for the specified destination network. The route is written to the system configuration and applied immediately to the running routing table. If a route with the same name already exists, the command shows the current and new values and prompts for confirmation before overwriting.
Parameters	• `--name NAME`: The route identifier. Must start with a letter and contain only letters, digits, and underscores. For example: `myRoute` or `corp_network` • `--net NETWORK/CIDR`: The destination network in CIDR notation. Host bits must be zero. For example: `203.0.113.0/24` or `2001:db8::/64` • `--gateway IP`: The next-hop gateway address. Must be in the same version as the destination (IPv4 or IPv6). For example: `192.0.2.1` or `2001:db8::1`; for link-local IPv6 gateways, include the interface scope, for example: `fe80::1%em0`
Where	The text-console (access directly or through *SSH*). User must have root privileges.

To apply	Enter the command and press enter. If the route already exists, confirm the overwrite when prompted.

Add a host static route

Product	Guardian
Syntax	`n2os-netconf routes --set --name NAME --host IP --gateway IP`
Description	Adds or updates a static route to a single host address. The route is written to the system configuration and applied immediately to the running routing table. If a route with the same name already exists, the command shows the current and new values and prompts for confirmation before overwriting.
Parameters	• `--name NAME`: The route identifier. Must start with a letter and contain only letters, digits, and underscores. For example: `hostRoute` • `--host IP`: The destination host address. For example: `203.0.113.5` or `2001:db8::5` • `--gateway IP`: The next-hop gateway address. Must be in the same version as the destination. For example: `192.0.2.1` or `2001:db8::1`
Where	The text-console (access directly or through *SSH*). User must have root privileges.
To apply	Enter the command and press enter. If the route already exists, confirm the overwrite when prompted.

Delete a static route

Product	Guardian
Syntax	`n2os-netconf routes --delete NAME`
Description	Deletes a static route by its name. The route is removed from the system configuration and from the running routing table. The command shows the route details and prompts for confirmation before deleting.
Parameters	`NAME`: The name of the route to delete. For example: `myRoute`
Where	The text-console (access directly or through *SSH*). User must have root privileges.
To apply	Enter the command and press enter. Confirm the deletion when prompted.

Chapter 12. Protocols



Configure protocols

Configure iec104s encryption key

Product	Guardian
Syntax	`conf.user configure probe protocol iec104s tls private_key <ip> <location>`
Description	Add a private key associated to the device running iec104s. For more information, see Configure IEC-62351-3 (on page 155).
Parameters	• `ip`: The IP of the device • `location`: The absolute location of the key
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set variable namespace for iec101 protocol decoder

Product	Guardian
Syntax	`conf.user configure probe protocol iec101 var_namespace <namespace>`
Description	iec101 CA size can vary across implementations, with this configuration rule the user can customize the setting for its own environment
Parameters	`namespace`: Namespace to be used in variable definition. It can be `la` or `ca` (default: `ca`)
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set CA size for iec101 protocol decoder

Product	Guardian
Syntax	`conf.user configure probe protocol iec101 ca_size <size>`
Description	iec101 CA size can vary across implementations, with this configuration rule the user can customize the setting for its own environment
Parameters	`size`: The size in bytes of the CA. Accepted values: 1,2 (default: 1)

Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set LA size for iec101 protocol decoder

Product	Guardian
Syntax	`conf.user configure probe protocol iec101 la_size <size>`
Description	iec101 LA size can vary across implementations, with this configuration rule the user can customize the setting for its own environment
Parameters	`size`: The size in bytes of the LA. Accepted values: 1,2 (default: 1)
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set the way the iec101 and iec104 Information Object Addresses (IOAs) are displayed

Product	Guardian
Syntax	`conf.user configure probe protocol [iec101|iec104] ioa_format [whole|split_in_2|split_in_bytes]`
Description	With `split_in_2`, which is the default, variables are created like `ioa-4-1002`, or with `split_in_bytes` they are named as `ioa-4-3-234`. When `whole` is set, variable addresses are represented as a unique 24 bits number.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set a dictionary file

Product	Guardian
Syntax	`conf.user configure probe protocol <protocol> dictionary <dictionary_file_name>`
Description	Based on the dictionary file set with this command, friendly names are associated to the extracted variables, for the specific protocol in scope.

Parameters	• `protocol`: The protocol can be can-bus or mvb • `dictionary_file_name`: The path for the dictionary file
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set an arbitrary amount of bytes to skip before decoding iec101 protocol

Product	Guardian
Syntax	`conf.user configure probe protocol iec101 bytes_to_skip <amount>`
Description	Based on the hardware configuration iec101 can be prefixed with a fixed amount of bytes, with this setting Guardian can be adapted to the peculiarity of the environment.
Parameters	`amount`: The amount of bytes to skip
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Enable the Red Electrica Espanola semantic for iec102 protocol

Product	Guardian
Syntax	`conf.user configure probe protocol iec102 ree [enabled|disabled]`
Description	There is a standard from Red Electrica Española which changes the semantic of the iec102 protocol, after enabling (choosing option enabled) this setting the iec102 protocol decoder will be compliant to the REE standard.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set the subnet in which the iec102 protocol will be enabled

Product	Guardian
Syntax	`conf.user configure probe protocol iec102 subnet <subnet>`

Description	The detection of iec102 can lead to false positives, this rules give the possibility to the user to enable the detection on a specific subnet
Parameters	`subnet`: A subnet in the CIDR notation
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Enable iec102 on the specified port

Product	Guardian
Syntax	`conf.user configure probe protocol iec102 port <port>`
Description	The detection of iec102 can lead to false positives, this rules give the possibility to the user to enable the detection on a specific port
Parameters	`port`: The TCP port
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set the subnet in which the iec103 protocol will be enabled

Product	Guardian
Syntax	`conf.user configure probe protocol iec103 subnet <subnet>`
Description	The detection of iec103 can lead to false positives, this rules give the possibility to the user to enable the detection on a specific subnet
Parameters	`subnet`: A subnet in the CIDR notation
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Enable iec103 on the specified port

Product	Guardian
Syntax	`conf.user configure probe protocol iec103 port <port>`
Description	The detection of iec103 can lead to false positives, this rules give the possibility to the user to enable the detection on a specific port

Parameters	`port`: The TCP port
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Force iec101 semantics inside iec103 protocol

Product	Guardian
Syntax	`conf.user configure probe protocol iec103 force_iec101_semantics true`
Description	Forces change of semantics for iec103 protocol to use ASDUs of iec101
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Allow to recognize as iec103 very fragmented sessions

Product	Guardian
Syntax	`conf.user configure probe protocol iec103 accept_on_fragmented true`
Description	Allow to accept as iec103 those packets that are always incomplete, thus allowing situations where the protocol is heavily fragmented to be recognized.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Enable the detection of cleartext passwords in HTTP payloads

Product	Guardian
Syntax	`conf.user configure probe protocol http detect_uri_passwords [true|false]`
Description	Guardian is able to detect if cleartext passwords and login credentials are present in HTTP payloads, such as strings containing `[scheme]://[username:password]@[host]`, e.g. `ftp://user:password@example.com`. The feature is disabled by default. Choose `true` to enable the feature and false to disable it.

Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Prevent the web_server role to be set on HTTP links

Product	Guardian
Syntax	`conf.user configure probe protocol http ignore_role_on_ports [tcp|udp]/port_number[,[tcp|udp]/port_number...]`
Description	In HTTP links, a `web_server` role is assigned by default to the server side of the connection. This configuration prevents this role from being assigned if the traffic is on the given ports. This can be useful to model cases where `http` is used as a transport for other means, and a web server is not actually responding on that link.
Parameters	`tcp` or `udp`: transport protocol `port_number`: the port number
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set the subnet in which the tg102 protocol will be enabled

Product	Guardian
Syntax	`conf.user configure probe protocol tg102 subnet <subnet>`
Description	The detection of tg102 can lead to false positives, this rules give the possibility to the user to enable the detection on a specific subnet
Parameters	`subnet`: A subnet in the CIDR notation
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set the port range in which the tg102 protocol will be enabled

Product	Guardian
Syntax	`conf.user configure probe protocol tg102 port_range <src_port>-<dst_port>`

Description	The detection of tg102 can lead to false positives, this rules give the possibility to the user to enable the detection on a specific port range
Parameters	• `src_port`: The starting port of the range • `dst_port`: The ending port of the range
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set the subnet in which the tg800 protocol will be enabled

Product	Guardian
Syntax	`conf.user configure probe protocol tg800 subnet <subnet>`
Description	The detection of tg800 can lead to false positives, this rules give the possibility to the user to enable the detection on a specific subnet
Parameters	`subnet`: A subnet in the CIDR notation
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Set the port range in which the tg800 protocol will be enabled

Product	Guardian
Syntax	`conf.user configure probe protocol tg800 port_range <src_port>-<dst_port>`
Description	The detection of tg800 can lead to false positives, this rules give the possibility to the user to enable the detection on a specific port range
Parameters	• `src_port`: The starting port of the range • `dst_port`: The ending port of the range
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Disable variable extraction for a Siemens S7 area and type

Product	Guardian
Syntax	`conf.user configure probe protocol s7 exclude <area> <type>`

Description	For performance reasons or to reduce noise it's possible to selectively exclude variables extraction for some areas and type.
Parameters	• `area`: The area, some examples are: DB, DI, M, Q • `type`: The type of the *Variable*, some examples are: INT, REAL, BYTE
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Enable the full TLS inspection mode

Product	Guardian
Syntax	`conf.user configure probe tls-inspection enable [true|false]`
Description	TLS inspection is normally performed only on https and iec104s traffic. Enabling (choosing the option `true`) the full inspection mode provides the following additional features: • TLS traffic found on any TCP port is inspected • an alert is raised when TLS-1.0 is used (when this mode is disabled, this is an https only check) • an alert is raised on expired certificates • an alert is raised on weak cipher suites • session ID, cipher suite and certificates are extracted into the relative link events
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Enable or disable the persistence of the connections for Ethernet/IP Implicit

Product	Guardian
Syntax	`conf.user configure probe protocol ethernetip-implicit persist-connection [true|false]`
Description	The Ethernet/IP Implicit decoder of Guardian is able to detect handshakes that are then used to decode variables. In some scenarios these handshakes are not common but it's very important to persist them so that Guardian can continue to decode variables after a reboot or an upgrade. By enabling (chosing option true) this option Guardian will store on disk the data needed to autonomously reproduce the handshake phase after a reboot.
Where	**CLI**

To apply	In a shell console execute: `service n2osids stop`

Enable detection of SSH banners flood

Product	Guardian
Syntax	`conf.user configure probe protocol ssh enable_banner_flood_detection [true|false]`
Description	With this option enabled, if an SSH server sends too many banners to a client per time interval, an alert is raised. By default this option is enabled.
Where	**CLI**
To apply	In a shell console execute: `n2os_ids_stop`

Enable or disable fragmented packets for modbus protocol

Product	Guardian
Syntax	`conf.user configure probe protocol modbus enable_full_fragmentation [true|false]`
Description	Modbus protocol is usually not fragmented, so this option is by default disabled (option false). If fragmented modbus packets can be present in the network, then full fragmentation can be enabled (choosing option true) to avoid generation of unexpected alerts.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Enable or disable the support for LACBUS-RTU

Product	Guardian
Syntax	`conf.user configure probe protocol modbus lacbus-rtu-protocol [true|false]`
Description	LACBUS-RTU is a protocol that extends Modbus. By default, this protocol is not parsed by Guardian. It can be enabled using this configuration, in which case LACBUS-RTU transmissions are still tagged as 'modbus', but the function codes specific to LACBUS-RTU will be processed, and the corresponding variables extracted.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Import the ge-egd produced data XML file for variables extraction

Product	Guardian
Syntax	`conf.user configure probe protocol ge-egd produced-data-xml <path>`
Description	The ge-egd protocol can extract process variables only after the XML file describing the produced data for the involved nodes is imported. Multiple imports are allowed as long as the XML files do not provide overlapping information for any producer node.
Parameters	`path`: The path of the produced data XML file to import
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Disable file extraction for SMB protocol

Product	Guardian
Syntax	`conf.user configure probe protocol smb file_extraction false`
Description	The SMB protocol decoder is able to extract files and analyze them for malware in a sandbox. If not needed, the user can disable such feature and improve the performance of the system especially in environments where SMB file transfer is heavily used.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Activate the extraction of GE asset information from modbus registers

Product	Guardian
Syntax	`conf.user configure probe protocol modbus ge_asset_info_from_registers true`
Description	Some General Electric devices send asset information (product name, firmware version, serial number, label, and FPGA version) encoded in register values with the Modbus protocol. By enabling this setting, Guardian is instructed to extract this data and enrich the corresponding nodes with it. This data is also used to produce CPEs for the corresponding devices.

Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Enable byte scan of traffic by unrecognized protocols in order to detect executables

Product	Guardian
Syntax	`conf.user configure probe enable-executable-extraction [true|false]`
Description	When this option is enabled, the traffic from unrecognized protocols is scanned in order to detect the presence of executables. If an executable is found, it is extracted and sent to Sandbox for analysis. This option is enabled by default. However, when a lot of traffic is flowing through unknown protocols, disabling it can improve performance.
Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Configure ARP flood

Product	Guardian
Syntax	`conf.user configure probe protocol arp flood <json_value>`

Description

A json object to configure ARP flood

- `enable` - Enable or disable flood detection. Possible values `true` or `false` by default is `false`.
- `packet_limit` - Define the number of packets per seconds (n) that trigger a flood detection (default = 1000).
- `packet_increase_limit` - Define the increase in the number of packets per seconds that trigger a flood detection (default = 1000). The difference between packet_limit and packet_increase_limit is that the first one checks the number of packets per second while the second checks the increase in the number of packets per second. For example if packet_limit = 3000 and packet_increase_limit = 500 if the arp packets increase by 100 each second no alert is triggered until they reach the threshold of 3000 packets per second, but if they increase in one second of 600, for example they pass from 1000 to 1600, then the flood is triggered.
- `packet_ratio_limit` - Define the ratio between the number of arp and total packets that trigger the flood . The flood is triggered when the ratio is greater than the specified value. In order to enable this detection method there is the need to enable also the counting of the total packets that can be done with probe packet-monitor enable true. A value of 0.1 means that the flood is triggered when the number of arp packets is 10% of the total packets. Default value is 1.0 (all packets are arp) in such a way that this threshold is disabled by default.
- `strict_mode` - When true all the above checks have to be verified at the same time to trigger the detection, while when strict_mode is false it is enough for a single check to trigger the flood. Possible values `true` or `false` by default is `false`.
- `repeat_mode` - Define how subsequent triggers are handled. Default value is `mute_for_period`. Possible values are:
  - `repeat`: alerts are raised at each detection
  - `mute_for_period`: after the first detection alerts are muted for a specified period
  - `mute_until_passes` no subsequent alerts are raised until the flood passes
- `mute_seconds` - The meaning depends on the repeat mode (default = 300 seconds):
  - `repeat_mode == mute_for_period` : It defines the number of seconds for which the alert is muted
  - `repeat_mode == mute_until_passes` : It defines the number of second for which the flood has to stay undetected in order to be considered passed

```
conf.user configure probe protocol arp flood {"enable": true}

conf.user configure probe protocol arp flood
 {"enable":true,"packet_limit":5}

conf.user configure probe protocol arp flood
 {"enable":true,"packet_limit":100000,"packet_increase_limit":
100000,"packet_ratio_limit":0.2}

conf.user configure probe packet-monitor enable true
```

Where	CLI
To apply	In a shell console execute: `service n2osids stop`

Enable or disable the counting of all the packets to be used as reference for arp flood

Product	Guardian
Syntax	`conf.user configure probe packet-monitor enable [true|false]`
Description	When this option is enabled, all the packets are counted to be used as reference for the relative ratio limit (default = false)

Set the maximum number of subnets per MAC address that can be stored in the ARP table reconstructed from the processed traffic

Product	Guardian
Syntax	`conf.user configure arp_tables max_subnets_per_mac <size>`

Description

This setting controls the maximum number of subnets per MAC address that can be stored in the ARP table reconstructed from the processed traffic.

- `enable` - Enable or disable flood detection. Possible values `true` or `false` by default is `false`.
- `packet_limit` - Define the number of packets per seconds (n) that trigger a flood detection (default = 1000).
- `packet_increase_limit` - Define the increase in the number of packets per seconds that trigger a flood detection (default = 1000). The difference between packet_limit and packet_increase_limit is that the first one checks the number of packets per second while the second checks the increase in the number of packets per second. For example if packet_limit = 3000 and packet_increase_limit = 500 if the arp packets increase by 100 each second no alert is triggered until they reach the threshold of 3000 packets per second, but if they increase in one second of 600, for example they pass from 1000 to 1600, then the flood is triggered.
- `packet_ratio_limit` - Define the ratio between the number of arp and total packets that trigger the flood . The flood is triggered when the ratio is greater than the specified value. In order to enable this detection method there is the need to enable also the counting of the total packets that can be done with probe packet-monitor enable true. A value of 0.1 means that the flood is triggered when the number of arp packets is 10% of the total packets. Default value is 1.0 (all packets are arp) in such a way that this threshold is disabled by default.
- `strict_mode` - When true all the above checks have to be verified at the same time to trigger the detection, while when strict_mode is false it is enough for a single check to trigger the flood. Possible values `true` or `false` by default is `false`.
- `repeat_mode` - Define how subsequent triggers are handled. Default value is `mute_for_period`. Possible values are:
  - `repeat`: alerts are raised at each detection
  - `mute_for_period`: after the first detection alerts are muted for a specified period
  - `mute_until_passes` no subsequent alerts are raised until the flood passes
- `mute_seconds` - The meaning depends on the repeat mode (default = 300 seconds):
  - `repeat_mode == mute_for_period` : It defines the number of seconds for which the alert is muted
  - `repeat_mode == mute_until_passes` : It defines the number of second for which the flood has to stay undetected in order to be considered passed

```
conf.user configure probe protocol arp flood {"enable": true}

conf.user configure probe protocol arp flood
 {"enable":true,"packet_limit":5}

conf.user configure probe protocol arp flood
 {"enable":true,"packet_limit":100000,"packet_increase_limit":
100000,"packet_ratio_limit":0.2}

conf.user configure probe packet-monitor enable true
```

Where	**CLI**
To apply	In a shell console execute: `service n2osids stop`

Enable/disable single protocol passive dissection

Product	Guardian
Syntax	`probe protocol protocol_name enable [true|false]`
Description	By default, protocols are enabled. In case there is an issue with a specific protocol dissection, this can be disabled.
Where	In a shell console, use **vi** or **nano** to edit the `/data/cfg/n2os.conf.user` file and add the configuration line.
To apply	In a shell console execute: `service n2osids stop`

Configure additional ports for ported protocols

Product	Guardian
Syntax	`probe protocol <protocol_name> ports tcp/<port_number>[,udp/port_number,tcp/port_number ...]`
Description	Ported protocols listen on specific ports to be tagged. This setting adds additional destination ports to tag the selected protocol. **Note:** This command is only applicable to some of the Nozomi Networks-ported protocols. If the setting does not work for the applicable protocol, contact Nozomi Networks.
Parameters	• `protocol_name`: the protocol in scope • `port_number`: the port to assign to the tagging. Multiple ports can be added, each with a `tcp/` or `udp/` prefix specifying the transport protocol
Where	In a shell console, use **vi** or **nano** to edit the `/data/cfg/n2os.conf.user` file and add the configuration line.
To apply	In a shell console execute: `service n2osids stop`

Turn on/off Device ID notifications based on protocol

Product	Guardian

Syntax	`conf.user configure probe protocol <protocol_name> notify_device_id <enabled>`
Description	Protocols may contain specific information (like serial numbers or other unique identifiers) that can be leveraged to obtain a better asset merging, also in absence of, for example, MAC addresses. The Nozomi field where the information is stored is Device ID (`device_id` in the nodes table): nodes sharing a device ID are aggregated into a single asset. This setting enables or disables the use of such information for a given protocol. Disabling such an aggregation can be used when the criterion does not work with particular corner cases (for example, the unique identifier is repeated in the network), or if it proves not to be accurate.
Parameters	`protocol_name`: values as listed in the `protocols` field of `nodes` `enabled`: [true|false] (default: true)
Where	**CLI**
To apply	It is applied automatically.

Enable or disable ASCII variable extraction for the Vnet/IP protocol

Product	Guardian
Syntax	`conf.user configure probe protocol vnet-ip ascii-variables [enable|disable]`
Description	When enabled, ASCII variables are extracted from the Vnet/IP protocol when they are present. When disabled, variables are not extracted even if they are present. This option can be used to avoid a potentially large number of variables when not explicitly required.
Parameters	`enable` | `disable`: respectively enable and disable variable extraction for the Vnet/IP protocol (default: `enable`).
Where	**CLI**
To apply	It is applied automatically.

Configure ports for tagging the encrypted Niagara Fox protocol

Product	Guardian
Syntax	`conf.user configure probe protocol niagara-fox encrypted_ports tcp/port_number[,tcp/port_number...]`

Description	Sets the ports used for tagging the encrypted version of the Niagara Fox protocol. An additional check on MAC addresses is still performed before tagging.
Parameters	`tcp`: transport protocol `port_number`: the port number
Where	**CLI**
To apply	It is applied automatically.



Chapter 13. Vulnerability assessment



Configure vulnerability assessments

Configure CVE matching

Product	Guardian
Syntax	`conf.user configure va cve enable [true|false|if_not_sync]`
Description	CVE matching depends on both the configuration value and whether the sensor is connected to an upstream system (CMC or Vantage). The behavior is as follows: • `true` CVE matching is always performed locally on the sensor, regardless of upstream connectivity • `false` CVE matching is completely disabled on the sensor • `if_not_sync` CVE matching follows the intended default behavior: ◦ Enabled locally only when the sensor is not connected upstream ◦ Delegated upstream when the sensor is connected **Important:** New sensors are explicitly configured with `if_not_sync` **Important:** When `if_not_sync` or `false` are not explicitly configured, CVE matching runs locally on the sensor. This can lead to increased CPU and memory usage on the sensor. Therefore, it is strongly recommended to explicitly configure `if_not_sync` when CVE processing can be delegated upstream.
Where	**CLI**
To apply	In a shell console execute: `service n2osva stop`

Configure CVE matching options

Product	Guardian
Syntax	`conf.user configure va cve options <json_value>`
Description	This command allows the configuration of diverse CVE matching options. The JSON object can have the following attributes: `skip_windows_if_hotfixes_missing`. This boolean controls whether CVE calculation should be skipped for Windows CPEs if hotfix information is not available (default: true).

Where	**CLI**
To apply	In a shell console execute: `service n2osva stop`

Enables the management of Microsoft Hotfixes

Product	Guardian
Syntax	`conf.user configure va hotfixes_enabled <flag>`
Description	Please consider that when this is set to true hotfixes are loaded and used to set CVEs status whereas when this flag is set to false, hotfixes are not loaded nor used by CVE calculation.
Parameters	`flag`: The management of Microsoft Hotfixes is enabled by default
Where	**CLI**
To apply	It is applied automatically

Disable the Microsoft Hotfixes resolution capabilities

Product	Guardian
Syntax	`conf.user configure va use_hotfix_resolution <flag>`
Description	Please consider that disabling the Microsoft Hotfixes resolution feature means that CVEs for Microsoft Windows machines will not be automatically closed through Smart Polling, and as a consequence those nodes might be assigned by Guardian a large number of obsolete CVEs.
Parameters	`flag`: Microsoft Hotfixes resolution is enabled by default
Where	**CLI**
To apply	It is applied automatically

Disable the CPE computation for a specific node

Product	Guardian
Syntax	`conf.user configure va cpe disable <node_id> [true|false]`
Description	Please consider that, when this command is used, the vulnerabilities assessment engine is completely disabled for that specific node and no CVEs will be assigned to the node itself.
Parameters	`node_id`: Node ID of the node targeting the rule

Where	**CLI**
To apply	It is applied automatically

Disable End Of Life CPEs calculation

Product	Guardian
Syntax	`conf.user configure va use_eol_cpe_calculation false`
Description	By default, when CVE associated to CPES calculation is perfomed, CPE that are referring to products that reached End Of Life are not taken into account. To disable this behaviour use this configuration.
Where	**CLI**
To apply	In a shell console execute: `service n2osva stop`



Chapter 14. Smart Polling



Configure Smart Polling

Enable/disable Smart Polling

Product	Guardian
Syntax	`conf.user configure sp enable [true|false]`
Description	When Smart Polling is enabled, it is automatically configured through Progressive Mode to create plans that can safely poll the devices known to the Guardian. The user can define additional manual plans to fine-tune the behavior of Smart Polling. When Smart Polling is disabled, all Progressive Mode plans are deleted, and manually-created plans are disabled. The user can still enable the manual plans and create new ones.
Where	**CLI**
To apply	It is applied automatically

Set the Smart Polling interval

Product	Guardian
Syntax	`conf.user configure sp polling_interval_s <interval>`
Description	Set the interval, in seconds, between Smart Polling runs. The default setting is 20 seconds.
Where	**CLI**
To apply	It is applied automatically

Set the log level for Smart Polling

Product	Guardian
Syntax	`conf.user configure sp log_level [DEBUG|INFO|WARNING|ERROR]`
Description	The default setting is `INFO` The supported levels are: • `DEBUG` • `INFO` • `WARNING` • `ERROR`
Where	**CLI**

To apply	It is applied automatically

Chapter 15. Discovery



Configure Discovery

Enable/disable Discovery

Product	Guardian
Syntax	`conf.user configure discovery enabled [true|false]`
Description	Enable/disable Discovery.
Where	**CLI**
To apply	It is applied automatically

Set the Discovery interval

Product	Guardian
Syntax	`conf.user configure discovery interval_s <interval>`
Description	Set the interval, in seconds, between Discovery runs. With the default settings, the intervals will be: • 1st interval - 5 minutes • 2nd interval - 6 minutes • 3rd interval - 7 minutes This incremental increase of 1 minute will continue until the interval is 60 minutes. If you manually set a value, such as **15** the interval will be fixed at 15 seconds, and will not increase.
Where	**CLI**
To apply	It is applied automatically

Set the Discovery duration

Product	Guardian
Syntax	`conf.user configure discovery duration_ms <duration>`
Description	This sets how long, in milliseconds (ms), Discovery waits for a response, The default setting is `7000`
Where	**CLI**
To apply	It is applied automatically



Chapter 16. Monitored subnets



Configure monitored subnets

Set the allowed subnets for Smart Polling and Discovery

Product	Guardian
Syntax	`conf.user configure smart_polling_and_discovery subnets allow […]`
Description	An empty allowlist is equivalent to `["0.0.0.0/0"]` (all subnets are allowed). Examples: • `conf.user configure smart_polling_and_discovery subnets allow ["10.13.0.0/16"]` • `conf.user configure smart_polling_and_discovery subnets allow ["10.13.4.0/24", "10.13.8.0/24"]` `conf.user configure smart_polling_and_discovery subnets allow []` will reset the settings to the default. When allowed and denied subnets are nested, the smallest subnet has priority. For example, in the example below the subnet 10.20.0.0/24 will be allowed, but the subnet 10.20.1.0/24 will be denied. • conf.user configure smart_polling_and_discovery subnets allow ["10.20.0.0/16"] • conf.user configure smart_polling_and_discovery subnets deny ["10.20.1.0/24"]
Where	**CLI**
To apply	It is applied automatically

Set the denied subnets for Smart Polling and Discovery

Product	Guardian
Syntax	`conf.user configure smart_polling_and_discovery subnets deny […]`

Description	Examples: • `conf.user configure smart_polling_and_discovery subnets deny ["10.13.0.0/16"]` • `conf.user configure smart_polling_and_discovery subnets deny ["10.13.4.0/24", "10.13.8.0/24"]` `conf.user configure smart_polling_and_discovery subnets deny []` will reset the settings to the default. When allowed and denied subnets are nested, the smallest subnet has priority. For example, in the example below the subnets 10.0.0.0/24 and 10.128.36.0/24 will be allowed, but the subnet 10.128.30.0/24 will be denied. • conf.user configure smart_polling_and_discovery subnets allow ["10.0.0.0/8", "10.128.36.0/24"] • conf.user configure smart_polling_and_discovery subnets deny ["10.128.0.0/16"]
Where	**CLI**
To apply	It is applied automatically

Chapter 17. Customize node identifier generation



Customize node identifier generation

All the entities that communicate in a network are called *nodes* and a Guardian assigns to each node a unique identifier, or *NodeID* in short. Generally, the NodeID is just an ip address (or a mac address), but in some special network topologies, extra information must be included in a NodeID to further differentiate nodes.

Note: NodeIDs generated with different settings will cause inconsistencies and should not coexist. These options should be manually set at sensor deploy time or on a Guardian with a clean configuration.

Include VLAN number in NodeID

Nodes can have their NodeID "decorated" with the VLAN ID of their zone.

Product	Guardian
Syntax	`nodeid_factory zone`
Description	Nodes included in a zone, which has a non-zero VLAN id, will get a NodeID of the form `ip@vlan`.

Include Remote Collector/Arc provenance in NodeID

Packets forwarded by Remote Collectors or Arc sensors carry a special "provenance" attribute that the Guardian uses to track precisely where the traffic was captured. The configuration directive `nodeid_factory include_capture` by default will use a standard NodeID for nodes seen by local capture devices, and append an explicative suffix `_from:...` to nodes appearing in remotely captured traffic.

Product	Guardian
Syntax	`nodeid_factory include_capture [local-traffic-tag] [format-string]`
Description	Enable decoration of NodeIDs with packet provenance information.
Parameters	• the optional local-traffic-tag is the provenance name for locally captured traffic: leave empty or use `no_localhost` to disable NodeID decoration on local traffic. • the format-string is the template for decorating remotely captured NodeIDs. A pair of curly braces `{}` will be expanded to the actual provenance. The default format is `"_from:{}"`

Notes	For packets captured by a Remote Collector, the default provenance is the ip of the Remote Collector itself. Alternatively, a Guardian can use the `site` of the Remote Collector (and fall back to the ip, when the site is undefined), by adding the directive `remote_capture_forward_packet_src true` to the Guardian configuration. For packets captured by Arc sensors, the provenance is a unique identifier of the Arc instance, followed by the ip that uploaded the data to the Guardian (so, it may not belong to the machine where Arc is running). The Guardian may use both `include_capture` and `zone` if the configuration contains both `nodeid_factory zone` and `nodeid_factory include_capture` in this order.

Chapter 18. Decryption



Protocol passive decryption

Nozomi Networks offers tools to decrypt network protocols and apply standard deep packet inspection (DPI) to the decrypted communication. Depending on the specific protocol, different constraints and configuration steps are applicable.

Table 3. Protocol passive decryption

Protocol	References	Credential type	Enablement steps

dlms	IEC 62056	DLMS encryption key	1. In the **Credentials Manager**, enter the credentials.
			2. In the `/data/cfg/n2os.conf.user` file, add this line:
			`credentials enable dlms-cosem true`
			3. As a privileged user from a shell, run the command:
			`service n2osids stop`

iec104s	IEC 60870-5-7 IEC 62351-5 IEC 62351-3 TLS	RSA-signed certificate private key	See Configure IEC-62351-3 (on page 155).
r-goose	IEC 62351-6 IEC 62351-9 RFC 6407	R-GOOSE encryption key	In the **Credentials Manager**, enter the credentials. This happens through an integration to a IEC 62351-9 GDOI Key Distribution Center.

snmpv3	RFC 5590
	RFC 3414
		SNMP privacy password	1. In the **Credentials Manager**, enter the credentials.
			2. In the `/data/cfg/n2os.conf.user` file, add this line:
			`credentials enable snmpv3 true`
			3. As a privileged user from a shell, run the command:
			`service n2osids stop`

IEC 60870-5-7 / 62351-3/5

IEC 60870-5-7 / 62351-3/5 encrypted links

IEC TC57 (POWER SYSTEMS management and associated information exchange) develops the standards 60870 and 62351. IEC 60870 part 5 (by WG3) describes systems used for telecontrol. IEC 62351 (by WG15) handles the security of TC 57 series.

IEC TC57 WG15 recommends the combination of IEC 62351-3 and 5 to secure IEC 60870-5-104 links:

- IEC 62351-3 is a TLS profile to secure power systems related communication.
- IEC 62351-5 is an application security protocol applicable to IEC 60870-5-101, 104, and derivatives. Its implementation in terms of ASDUs (i.e., real encapsulation) is outlined in IEC 60870-5-7.

In order to decrypt IEC 62351-3 (TLS) traffic, you must meet these conditions:

- The private key for each TLS server (e.g. RTU, PLC) must be available; it is used to derive session keys.
- All the equipment where decryption is needed must operate using the **TLS_RSA_WITH_AES_128_CBC_SHA** (0x00002f) cipher suite. Often, this step is accomplished by forcing either the client or the server to confine itself to that specific cipher suite.

Configure IEC-62351-3

About this task

The following steps assume we're decoding the communication of a TLS server with the address 192.168.1.26.

Procedure

1. Upload the TLS server's private key to **/data/cfg**. The file name must match the server's address. In our case, the file must be named **192.168.1.26.key**.
   Your key should be similar to the following:

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQDIi4LBdRJTwGQBF91dRzYFPNb6ZsHDp9tuucB2vYkRfWXR/FLG
zFsEA+ws5BcKC/JxTy6Ud3IJL3dkPMLHEpIWvNHOEAWJ0wRXe4TlKD6vZH91Bu4X
yY/hi9W4MM4CD/WB/s/+QXnCGwXXsitXbNDnB6mbpNRKWrFCXrREFpO9eQIDAQAB
AoGALcymrPXGnLfoXtOLa30S2jldw0vqXnucQnpLeRKqFgN0WHNumBYdhxo6OrpQ
hKvbz4LM2X053t0nQA4cUYhtMjoJDZk6uaj5e2lX5qDDCWqb5RsDiL+UcaMwA9wN
Dtu/zo49GdMKnPBx4Gcf8bTEnGs7JyGJ9aQMzxUGUMFSgH0CQQD84Bg6vwPrWUY9
bigXQgUtggpmJ1VUVcIaZ8QNZbqwjB7C60fy06cii6kHs4Mi8Cc50UUjfinC3VSI
SjeVmdaLAkEAywXh7FOXtzWWpFU89gCLZh+/7cB8hDlmlg5FfDbZU7Nxq1ERGCoH
930GMnVaHhXjmye/ORYaVS6LnEUbeezAiwJBAJKTwveqaTcuVEyfYZW5qw6p5iU4
f3mXDCAKLdYmNYTIc7xAbUAP9z1K/vt7znOeCN62swDTzJkreihV/Mo2lekCQQDG
P8xJdadk5BNiDUol/oohA0fr5law10xxoyX/EaBOt5975wGGUyTMa22twCJr1nwX
ekykdgzouTbjNiiKesHZAkBXp4B1OnVSWAaYw88P61mr5QGl3oChZYQx15r1i9GI
LTWmnCbyQu1Nm8df8wNdMgfvqWgbrlfJJqSIJtZhk5+U
-----END RSA PRIVATE KEY-----
```

2. In Guardian's Features Control Panel, enable link events; this provides visibility to the TLS decoded handshakes; for example:

192.168.0.104	192.168.1.26	tcp/19998	TCP_SYN:FIRST	3	52792	19998
192.168.0.104	192.168.1.26	tcp/19998	UP	4	52792	19998
192.168.0.104	192.168.1.26	iec104s	SSLv2_CLIENT_HELLO(TLS-1.0)	5	52792	19998
192.168.1.26	192.168.0.104	iec104s	TLS_SERVER_HELLO(TLS-1.0)	6	19998	52792
192.168.1.26	192.168.0.104	iec104s	TLS_SERVER_HELLO(TLS_RSA_WITH_AES_128_CBC_SHA)	7	19998	52792
192.168.1.26	192.168.0.104	iec104s	TLS_HANDSHAKE_CERTIFICATE	8	19998	52792
192.168.0.104	192.168.1.26	iec104s	TLS_HANDSHAKE_CERTIFICATE	9	52792	19998
192.168.0.104	192.168.1.26	iec104s	TLS_HANDSHAKE_CLIENT_KEY_EXCHANGE	10	52792	19998
192.168.0.104	192.168.1.26	iec104s	TLS_CHANGE_CIPHER_SPEC(client)	11	52792	19998
192.168.1.26	192.168.0.104	iec104s	TLS_CHANGE_CIPHER_SPEC(server)	12	19998	52792
192.168.0.104	192.168.1.26	iec104s	DOWN	13	52792	19998

3. Specify the key file's location by defining it in the **CLI**. To continue our example, we would use the following string:

```
conf.user configure probe protocol iec104s tls private_key
 192.168.1.26 /data/cfg/192.168.1.26.key
```

4. Repeat these steps for each applicable TLS server key.
5. Run the following command in a shell console:

```
service n2osids stop
```

Chapter 19. Trace



Configure trace

Trace size and timeout

A trace is a sequence of packets saved to the disk in the PCAP file format. The number of packets in a trace is fixed, this way when a trace of N packets is triggered Guardian starts to write to disk the N/2 packets that were sniffed before the trace was triggered, after that it tries to save another N/2 packets and then finalize the write operation, at this point the trace can be downloaded. To avoid a trace being pending for too much time there is also a timeout, when the time expires the trace is saved also if the desired number of packets has not been reached.

Trace files are stored in directory `/data/traces`, which employs disk based storage. In order to improve performance though, in machines with larger memory configurations this directory is backed by RAM based storage.

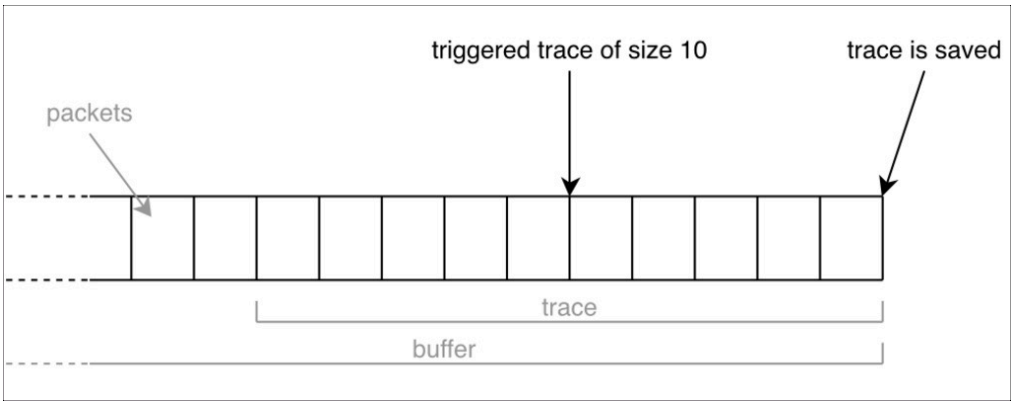


Figure 4. A schematic illustration of the trace saving process

Set max trace packets

Product	Guardian
Syntax	`conf.user configure trace trace_size <size>`
Description	The maximum number of packets that will be stored in the trace file.
Parameters	`size`: Default value 5000
Where	**CLI**
To apply	It is applied automatically

Set trace request timeout

Product	Guardian
Syntax	`conf.user configure trace trace_request_timeout <seconds>`

Description	The time in seconds after which the trace will be finalized also if the `trace_size` parameter is not fulfilled
Parameters	`seconds`: Default value 60
Where	**CLI**
To apply	It is applied automatically

Set max pcaps to retain

Product	Guardian
Syntax	`conf.user configure trace max_pcaps_to_retain <value>`
Description	The maximum number of PCAP files to keep on disk, when this number is exceeded the oldest traces will be deleted. Both automatic alert traces and user-requested traces are included. This is a runtime machine setting used for self protection prevailing on the retention settings as described in the Configuring retention section
Parameters	`value`: Default value 100000
Where	**CLI**
To apply	It is applied automatically

Set minimum free disk percentage

Product	Guardian
Syntax	`conf.user configure trace min_disk_free <percent>`
Description	The minimum percentage of disk free under which the oldest traces will be deleted. If the traces directory is memory backed, this configuration cannot be overridden and the default value will always be used.
Parameters	`percent`: Default value 10 if traces storage is disk backed, 5 if memory backed. Enter without `%` sign
Where	**CLI**
To apply	It is applied automatically

Set maximum occupied space

Product	Guardian

Syntax	`conf.user configure retention trace_request occupied_space <max_occupied_bytes>`
Description	The maximum traces occupation on disk in bytes. If the traces directory is memory backed, this configuration cannot be overridden and the default value will always be used.
Parameters	`max_occupied_bytes`: Default value is half of disk size if traces storage is disk backed, 95% of available space if memory backed
Where	**CLI**
To apply	It is applied automatically

Configure continuous trace

Set max continuous trace occupation in bytes

Product	Guardian
Syntax	`conf.user configure continuous_trace max_bytes_per_trace <size>`
Description	The maximum size in bytes for a continuous trace file.
Parameters	`size`: Default value 100000000
Where	**CLI**
To apply	It is applied automatically

Set max pcaps to retain

Product	Guardian
Syntax	`conf.user configure continuous_trace max_pcaps_to_retain <value>`
Description	The maximum number of PCAP files to keep on disk, when this number is exceeded the oldest traces will be deleted. This is a runtime machine setting used for self protection prevailing on the retention settings as described in the Configuring retention section
Parameters	`value`: Default value 100000
Where	**CLI**
To apply	It is applied automatically

Set minimum free disk percentage

Product	Guardian
Syntax	`conf.user configure continuous_trace min_disk_free <percent>`
Description	The minimum percentage of disk free under which the oldest continuous traces will be deleted
Parameters	`percent`: Default value 10, enter without `%` sign
Where	**CLI**

To apply	It is applied automatically

Set maximum occupied space

Product	Guardian
Syntax	`conf.user configure retention continuous_trace <occupied_space>`
Description	The maximum continuous traces occupation on disk in bytes
Parameters	`occupied_space`: Default value is half of disk size
Where	**CLI**
To apply	It is applied automatically



Chapter 20. Time Machine



Configure Time Machine

In this section we will configure the Nozomi Networks Solution Time Machine functionality.

Set snapshot interval

Products	CMC, Guardian
Syntax	`conf.user configure tm snap interval <interval_seconds>`
Description	Set the desired interval between snapshots, in seconds.
Parameters	`interval_seconds`: The amount of seconds between snapshots (default: 3600, minimum: 3600)
Where	**CLI**
To apply	`service n2osjobs stop`

Enable or disable automatic snapshot for each alert

Product	Guardian
Syntax	`conf.user configure tm snap on_alert [true|false]`
Description	It can enable (option true) or disable (option false) the possibility to take a snapshot on alert. By default snapshots are taken only for VI alerts; it is possible to explicitly set the alerts that will trigger automatic snapshots via `on_alert_trigger`
Where	**CLI**
To apply	`service n2osjobs stop`

Configure how alerts trigger automatic snapshots

Product	Guardian
Syntax	`conf.user configure tm snap on_alert_trigger <json_value>`

Description	Configures the alert triggers for automatic snapshots. The JSON object must have the following attribute:
	`type_ids` - A JSON array of the alert type IDs that will trigger an automatic snapshot. These type IDs may be literals or wildcarded ones (the asterisk can be used to match any substring).
	For example, the following command will configure the system to automatically take a snapshot whenever a VI:NEW-NODE or VI:NEW-LINK alert occurs:
	`conf.user configure tm snap on_alert_trigger {"type_ids": ["VI:NEW-NODE", "VI:NEW-LINK"]}`
	As a second example, the command below will configure the system to take a snapshot on all VI or SIGN alerts:
	`conf.user configure tm snap on_alert_trigger {"type_ids": ["VI:*", "SIGN:*"]}`

Parameters	`json_value`: A JSON object describing the alert automatic snapshot triggers (default: `{"type_ids": ["VI:*"]}`)
Where	**CLI**
To apply	`service n2osjobs stop`

Set maximum number of network elements allowed in a diff

Product	Guardian
Syntax	`conf.user configure tm diff max_results_network_elements <num_elements>`
Description	When comparing time machine snapshots that are too different, it is possible to overtax the system resources (memory, CPU). By setting a limit on the number of network elements that are allowed to be reported in a diff, the system is protected by such effects. When this threshold is crossed, the diff job is aborted and the appropriate error message is shown to the user.
Parameters	`num_elements`: Maximum number of network elements that may be reported by a diff (default: 10000)
Where	**CLI**
To apply	It is applied automatically

Chapter 21. Retention



Configure retention

Retention of historical data is controlled for each persisted entity by a configuration entry. Modify it to extend or reduce the default retention.

By default, the CMC retains 500,000 alerts. Note that retaining large numbers of alerts can impair performance. We recommend limiting the number of alerts generated rather than retaining more data. If you want to retain more alerts, we recommend an iterative approach of incrementally increasing this value and evaluating the system's performance. In some cases, you may want to send alerts to a different system using our data integration features instead of retaining the alerts in the sensor.

Alerts retention

Products	CMC, Guardian
Syntax	`conf.user configure retention alert rows <rows_to_retain>`
Description	Set the amount of alerts to retain. **NOTE**: When an alert is deleted, the related trace file is deleted too.
Parameters	`rows_to_retain`: The number of rows to keep (default: 500000)
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Alerts advanced retention

Products	CMC, Guardian
Syntax	`conf.user configure retention alert.out_of_security_profile rows <rows_to_retain>`
Description	Set the amount of alerts out of security profile to retain. By default, this feature is disabled. **NOTE**: • This retention has a higher priority than `retention alert rows <rows_to_retain>` and will be executed before it. • When an alert is deleted, the related trace file is deleted too.
Parameters	`rows_to_retain`: The number of rows to keep (disabled by default)
Where	**CLI**
To apply	It is applied automatically

Note	You can also change this configuration from the Web UI.

Trace retention size

Product	Guardian
Syntax	`conf.user configure retention trace_request occupied_space <max_occupied_bytes>`
Description	The maximum traces occupation on disk in bytes. If the traces directory is memory backed, this configuration cannot be overridden and the default value will always be used.
Parameters	`max_occupied_bytes`: Default value is half of disk size if traces storage is disk backed, 95% of available space if memory backed
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Trace retention rows

Product	Guardian
Syntax	`conf.user configure retention trace_request rows <rows_to_retain>`
Description	Set the amount of traces to retain.
Parameters	`rows_to_retain`: The number of rows to keep (default: 10000)
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Trace advanced retention

Products	CMC, Guardian
Syntax	`conf.user configure retention trace_request.<generation_cause> rows <rows_to_retain>`

Description	Set the amount of traces retained considering their generation cause. By default, these options are disabled. **NOTE**: This retention has a higher priority than `retention trace rows <rows_to_retain>` and will be executed before it. Moreover, These advanced retention options depend on each other, thus they must be configured all together or none.
Parameters	• `generation_cause`: Can be any of: • by_alerts_high: traces generated by high risk alerts • by_alerts_medium: traces generated by medium risk alerts • by_alerts_low: traces generated by low risk alerts • by_user_request: traces generated by a request from the user • `rows_to_retain`: The number of rows to keep
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

For example, we can configure the trace retention with the following command:

```
conf.user configure retention trace_request 10000
```

and also set up the advanced retention with:

```
conf.user configure retention trace_request 10000

 conf.user configure retention trace_request.by_alerts_high 5000
 conf.user configure retention trace_request.by_alerts_medium 1000
 conf.user configure retention trace_request.by_alerts_low 1000
 conf.user configure retention trace_request.by_user_request 3000
```

Continuous trace retention size

Product	Guardian
Syntax	`conf.user configure retention continuous_trace occupied_space <max_occupied_bytes>`
Description	Set max occupation in bytes for continuous traces
Parameters	`max_occupied_bytes`: the number of bytes to keep (default: half of disk size)
Where	**CLI**
To apply	In a shell console execute: `service n2ostrace stop`

Note	You can also change this configuration from the Web UI.

Continuous trace retention rows

Product	Guardian
Syntax	`conf.user configure retention continuous_trace rows <rows_to_retain>`
Description	Set the amount of continuous traces to retain
Parameters	`rows_to_retain`: the number of rows to keep (default: 10000)
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Link events retention

Product	Guardian
Syntax	`conf.user configure retention link_event rows <rows_to_retain>`
Description	Set the amount of link events to retain
Parameters	`rows_to_retain`: The number of rows to keep (default: 2500000)
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Captured urls retention

Product	Guardian
Syntax	`conf.user configure retention captured_urls rows <rows_to_retain>`
Description	Set the amount of captured "urls" (http queries, dns queries, etc) to retain
Parameters	`rows_to_retain`: The number of rows to keep (default: 10000)
Where	**CLI**
To apply	It is applied automatically

Note	You can also change this configuration from the Web UI.

Variable history retention

Product	Guardian
Syntax	`conf.user configure retention variable_history rows <rows_to_retain>`
Description	Set the amount of variable historical values to retain
Parameters	`rows_to_retain`: The number of rows to keep (default: 1000000)
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Node CVE retention

Product	Guardian
Syntax	`conf.user configure retention node_cve rows <rows_to_retain>`
Description	Set the maximum amount of node_cve entries to retain
Parameters	`rows_to_retain`: The number of rows to keep (default: 100000)
Where	**CLI**
To apply	In a shell console execute: `service n2osva stop`
Note	You can also change this configuration from the Web UI.

Node CPE retention

Product	Guardian
Syntax	`conf.user configure retention node_cpe rows max_age <days>`
Description	Configure automatic deletion of old node CPEs. If a CPE is not observed in the network for the configured number of days, it is scheduled for deletion. Set `days` to 0 to disable the time limit.
Parameters	`days`: The number of days to keep (default: 0)
Where	**CLI**

To apply	In a shell console execute: `service n2osva stop`

Uploaded traces retention

Product	Guardian
Syntax	`conf.user configure retention input_pcap rows <files_to_retain>`
Description	Set the amount of PCAP files to retain
Parameters	`files_to_retain`: The number of files to keep (default: 10)
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

File quarantine retention

Product	Guardian
Syntax	`conf.user configure retention quarantine number_of_files <files_to_retain>`
Description	Set the number of files to retain. When a new file is added to a sensor, Nozomi deletes the oldest quarantined file if the file exceeds this limit and the sensor needs to free disk space.
Parameters	`files_to_retain`: The number of files to keep (default: 50)
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Milestone bundle retention

Products	CMC, Guardian
Syntax	`conf.user configure retention milestone number_of_files <bundles_to_retain>`
Description	Set the number of milestone update bundles to keep on disk. When a new bundle becomes available (downloaded from upstream or manually uploaded), older bundles beyond this limit are removed.

Parameters	`bundles_to_retain`: The number of bundles to keep (default: 0, no bundles are retained). The maximum depends on the number of upgrade milestones available to the appliance.
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.



Chapter 22. Bandwidth throttling



Configure bandwidth throttling

It is possible to limit the bandwidth that a sensor's management port has at its disposal (so for access and updates) by specifying the maximum amount of allowed traffic.

Limit traffic shaping bandwidth

Product	Guardian
Syntax	`conf.user configure system traffic_shaping bandwidth <max_bandwidth>`
Description	Set the maximum outbound bandwidth that the sensor's management interface can use. Inbound data is still unlimited. By default this works on all the outbound traffic and you can exclude some IPs. Instead if you use 'system traffic_shaping include ' it will limit only the outbound traffic of the inclusions and everything else will be unlimited.
Parameters	`max_bandwidth`: the bandwidth limit. The following units are supported: b, Kb, Mb, Gb. When no unit is specified, b is intended by default (i.e. bits per second). When setting a limit in decimal notation make sure you add the all the leading zeros, and the unit (e.g., write 0.015Mb, not .015Mb). (default: no limitation).
Where	**CLI**
To apply	Update rules with `n2os-firewall-update`. On a fresh installation a reboot is necessary.

For example, we can set a limit of two megabits with the following configuration command:

```
conf.user configure system traffic_shaping bandwidth 2Mb
```

Note that this command affects only the sensor on which it is executed, its effects are not propagated to other sensors.
It is possible to exclude from the limitation of the bandwidth specific hostnames or IPs.

Exclude IP from traffic shaping

Product	Guardian
Syntax	`conf.user configure system traffic_shaping exclude <ip>`
Description	Set the IP to exclude from the limitation. This command and 'system traffic_shaping include' are mutually exclusive.
Parameters	`ip`: the IP to exclude. It can be a single IP, a class of IPS or an hostname (e.g. 192.168.12.34 or 192.168.0.0/16 or `vantage.nozominetworks.io`).

Where	**CLI**
To apply	Update rules with `n2os-firewall-update`. On a fresh installation a reboot is necessary.

For example, we can exclude an IP with the following configuration command:

```
conf.user configure system traffic_shaping exclude 192.168.12.34
```

Note that this command affects only the sensor on which it is executed, its effects are not propagated to other sensors.
Or to include in the limitation of the bandwidth only specific hostnames or IPs.

Include IP to traffic shaping

Product	Guardian
Syntax	`conf.user configure system traffic_shaping include <ip>`
Description	Set the IP to include in the limitation. This command and 'system traffic_shaping exclude' are mutually exclusive.
Parameters	`ip`: the IP to include. It can be a single IP, a class of IPS or an hostname (e.g. 192.168.12.34 or 192.168.0.0/16 or `vantage.nozominetworks.io`).
Where	**CLI**
To apply	Update rules with `n2os-firewall-update`. On a fresh installation a reboot is necessary.

For example, we can include an IP with the following configuration command:

```
conf.user configure system traffic_shaping include 192.168.12.34
```

Note that this command affects only the sensor on which it is executed, its effects are not propagated to other sensors.
Inclusions and exclusions are mutually exclusive and you cannot use both simultaneously.

Chapter 23. Synchronization



Configure synchronization

In this section we will configure the synchronization between sensors at different levels.

Set the global synchronization interval (notification message)

Product	CMC
Syntax	`conf.user configure cmc sync interval <interval_seconds>`
Description	Set the desired global synchronization interval for the in-scope sensor. Configuration is defined on the parent sensor; synchronization starts at child sensors and flows upstream. Each and every sync takes place following a notification message sent by the child sensor, stating that the child sensor is ready to synchronize data to its parent. The notification messages act as global synchronization settings, working together with the following settings as well. **Note**: In a multi-level deployment (e.g., one with root CMC, local CMC, and Guardian), the setting must be applied at each parent level (e.g., at the root CMC as well as at the local CMC).
Parameters	`interval_seconds`: the number of seconds between status notifications (default: 60)
Where	**CLI**
To apply	It is applied automatically

Set the DB synchronization interval

Products	CMC, Guardian
Syntax	`conf.user configure cmc sync_db_interval <interval_seconds>`
Description	Set the desired interval between DB synchronizations for the in-scope sensor. Configuration is done on the parent sensor; synchronization starts at child sensors and flows upstream. The setting applies to each DB element subject to synchronization (e.g., Alerts, Assets, Audit logs, and Health logs). As the interval expires, the DB entries are synchronized at the next notification message. **Note**: In a multi-level deployment (e.g., one with root CMC, local CMC, and Guardian), if the setting is applicable, it must be applied at each parent level (e.g., at the root CMC as well as at the local CMC).
Parameters	• `interval_seconds`: the number of seconds between DB synchronizations (default: 60). This parameter only makes sense when set higher than the global synchronization interval.
Where	**CLI**

To apply	It is applied automatically

Set the filesystem synchronization interval

Product	CMC
Syntax	`conf.user configure cmc sync_fs_interval <interval_seconds>`
Description	Set the desired interval between filesystem synchronizations for the sensor in scope, from its child sensors. The setting applies to each filesystem element subject to synchronization (e.g., nodes, links, and variables). As the interval expires, the filesystem entries are synchronized at the next notification message. In case the CMC is All-In-One, this interval will be used as default value for the Set the configurations merge interval (on page 188). **Note**: In a multi-level deployment (e.g., one with root CMC, local CMC, and Guardian), if the setting is applicable, it must be applied at each parent level (e.g., at the root CMC as well as at the local CMC).
Parameters	• `interval_seconds`: the number of seconds between filesystem synchronizations (default: 10800 [3 hours] if the CMC is multi-context, 720 (12 minutes) if the CMC is All-In-One). This parameter only makes sense when set higher than the global synchronization interval.
Where	**CLI**
To apply	It is applied automatically

Set the binary files synchronization interval

Product	CMC
Syntax	`conf.user configure cmc sync_binary_files_interval <interval_seconds>`
Description	Set the desired interval between binary files synchronizations for the sensor in scope, from its child sensors. The setting applies to each binary file element subject to synchronization (e.g., PDF reports). As the interval expires, the binary file entries are synchronized at the next notification message. **Note**: In a multi-level deployment (e.g., one with root CMC, local CMC, and Guardian), if the setting is applicable, it must be applied at each parent level (e.g., at the root CMC as well as at the local CMC).
Parameters	`interval_seconds`: the number of seconds between binary files synchronizations (default: 60). This parameter only makes sense when set higher than the global synchronization interval.

Where	**CLI**
To apply	It is applied automatically

Set the rows to be sent at every DB synchronization for each DB element

Products	CMC, Guardian
Syntax	`conf.user configure cmc sync record_per_loop <number_of_record_per_loop>`
Description	The system allows the user to customize the synchronization, in particular the number of records to be sent at each phase. A synchronization phase is composed of 50 steps for each DB element, every one sending number_of_record_per_loop rows, which means that the system sends, by default, 2500 rows every time.
Parameters	`number_of_record_per_loop`: the number of DB rows sent per single request (default: 50)
Where	**CLI**
To apply	It is applied automatically

Synchronize only visible alerts

Products	CMC, Guardian
Syntax	`conf.user configure cmc sync send_only_visible_alert [true|false]`
Description	Set whether to synchronize all alerts from the child sensors to the in-scope parent sensor (false), or to synchronize only visible alerts (as defined in the Security Profile) (true). Default: false. **Note**: In a multi-level deployment (e.g., one with root CMC, local CMC, and Guardian), if the setting is applicable, it must be applied at each parent level (e.g., at the root CMC as well as at the local CMC).
Where	**CLI**
To apply	It is applied automatically

Set the alert rules execution policy

Product	CMC

Syntax	`conf.user configure alerts execution_policy alert_rules [upstream_only|upstream_prevails|local_prevails]`
Description	Set the desired execution policy for the alert rules. **Note**: In a multi-level deployment (e.g., one with root CMC, local CMC, and Guardian), if the setting is applicable, it must be applied at each parent level (e.g., at the root CMC as well as at the local CMC).
Where	**CLI**
To apply	It is applied automatically
Note	You can also change this configuration from the Web UI.

Set the configurations merge interval

Product	CMC
Syntax	`conf.user configure cmc merge interval <interval_seconds>`
Description	Periodically, CMC All-In-One merge all the filesystem elements received by the connected Guardians. This operation strictly depends on the filesystem synchronization. It is possible to define a custom interval, however it is suggested to specify a similar value as the one set for the filesystem synchronization. **Note**: In a multi-level deployment (e.g., one with root CMC, local CMC, and Guardian), if the setting is applicable, it must be applied at each parent level (e.g., at the root CMC as well as at the local CMC).
Parameters	`interval_seconds`: the number of seconds between two merging actions (default: it follows the value of the Set the filesystem synchronization interval (on page 186).
Where	**CLI**
To apply	It is applied automatically

Enable the PostgreSQL advisory locks for assets synchronization

Product	CMC
Syntax	`conf.user configure cmc save_assets_with_advisory_lock [true|false]`

Description	Enable this feature to avoid potential database deadlocks on assets. This option shall be applied on mid-level CMC if the bulk asset synchronization is not enabled. **Note**: This option applies only on the CMC it is configured on. It is enabled by default.
Where	**CLI**
To apply	It is applied automatically

Enable content modifications on a Guardian or CMC to be local only

Products	CMC, Guardian
Syntax	`conf.user configure threat_intelligence local_contents enable [true|false]`
Description	Enable this feature to be able to modify contents on a machine even when it has an upstream connection. All changes will be local only, and changes from upstream will not be pushed down. If the feature is afterwards disabled, all the changes from the upstream will be propagated, and overwrite all the modifications done when the feature was enabled.
Where	**CLI**
To apply	It is applied automatically

Enable Smart Polling entities synchronization

Products	CMC, Guardian
Syntax	`cmc sync-force-default smart_polling [true|false]`
Description	Enable this feature to allow synchronization of Smart Polling configuration data from sensors connected upstream through a data diode. When disabled (default), Smart Polling data will not be synchronized in this scenario. In non-data diode architectures, synchronization of Smart Polling data can be enabled upstream as needed.
Where	**CLI**
To apply	In a shell console execute: `service n2osjobs stop`

Enable Smart Polling node points synchronization

Products	CMC, Guardian

Syntax	`cmc sync-force-default sync_node_points [true|false]`
Description	Enable this feature to allow synchronization of node point data gathered via Smart Polling from sensors connected upstream through a data diode. When disabled (default), Smart Polling data will not be synchronized in this scenario. In non-data diode architectures, synchronization of Smart Polling data can be enabled upstream as needed.
Where	**CLI**
To apply	In a shell console execute: `service n2osjobs stop`

Allow appliances automatically on first notification

Products	CMC, Guardian
Syntax	`conf.user configure allow_on_first_notify <appliance_type> [true|false]`
Description	Specifies whether appliances of a certain type are automatically allowed upon receiving their first notification. By default, appliances are disallowed. Enable this setting to automatically allow appliances when their initial notification is received.
Parameters	`appliance_type`: `arc` | `remote_collector` | `cmc` | `guardian`
Where	**CLI**
To apply	It is applied automatically

Troubleshooting

In N2OS v25.6.0, and later, you can use command-line interface (CLI) commands to print the contents, statistics, and configuration parameters of the commit logs. This information helps monitor synchronization status and diagnose potential issues. Outputs are provided in JavaScript Object Notation (JSON) format for easier parsing and analysis.

Print the contents of the commit log

Product	CMC, Guardian
Syntax	`ids commit_log_entries [{"limit": <number>}]`
Output	Prints, in *JavaScript Object Notation (JSON)* format, the current contents of the commit log. Each commit description has the following layout: • `operation`: is the name of the committed operation: ◦ `NOP`: no operation, used internally for commit log maintenance operations ◦ `CREATE`: create a node ◦ `SET_SUBNET`: set the subnet of a node ◦ `SET_BROADCAST_IP`: set the broadcast address of a node ◦ `RENAME`: rename a node ◦ `REMOVE`: remove a node • `cid`: also known as commit ID, is an integer value uniquely identifying a commit • `timestamp`: date and time, in ISO-8601 format, at which the operation was committed • `diff`: changes from the previous commit. It provides three fields: `added`, `updated`, and `removed`. They describe nodes that were respectively added, updated, or removed. The field `uid` (unique ID) is always part of the node description. The fields `id` (node ID) and `subnet` are also present in this description if the corresponding information is available
Parameters	`{"limit": <number>}` where `limit` is an integer value specifying the number of most recent commits to display.
Where	**CLI**

Print commit log statistics

Product	CMC, Guardian
Syntax	`ids commit_log_stats`

Output	Prints, in *JSON* format, a list of counter names and their corresponding values. These counters help understand if the parameters governing the behavior of the commit log are correctly tuned: • `commit_log_size`: is the current number of entries in the commit log • `commit_log_changes`: keeps track of the total number of commits and node snapshots. This quantity is used by the garbage collector to control the growth of the commit log • `commit_log_stale_entries`: keeps track of the number of commits that have not been processed by any synchronization context after a timeout expired • `commit_log_compactions_since_1h`: is the number of compactions that were performed by the garbage collector during the last hour • `commit_log_prunings_since_1h`: is the number of pruning operations that were performed during the last hour. A pruning operation occurs when attempting to add a snapshot to a node that has already reached its maximum allowed number of snapshots
Parameters	None
Where	**CLI**

Print the current values of the commit log configuration parameters

Product	CMC, Guardian
Syntax	`ids commit_log_configuration`

Output	Prints, in *JSON* format, the current value of each commit log configuration parameter: • `enable_consistent_synchronization`: is set to `true` if synchronization between *Central Management Console (CMC)*s and Guardians via the `consume` command is done consistently based on information from the commit log. If it is set to `false`, synchronization leverages information from the nodes of the system, which is the default behavior for all releases prior to v25.6.0. From v25.6.0 onwards, this parameter is set to `true` by default • `commit_log_max_append_tries`: specifies the maximum number of attempts that should be performed to append a entry to the commit log • `max_snapshots_per_node`: is the maximum allowed number of snapshots, for example, versioned data instances, per node • `max_changes_to_nodes_ratio`: is the maximum allowed ratio of commits and nodes snapshots to nodes. This parameter is used by the garbage collector to decide whether or not to compact the commit log • `commit_log_sync_timeout`: is the amount of time, expressed in minutes, after which a commit never accessed by synchronization since it was appended becomes a candidate for eviction by the garbage collector
Parameters	None
Where	**CLI**



Chapter 24. Slow updates



Configure slow updates

In this section we show how to configure a sensor for receiving firmware updates from an upstream sensor at a determined speed. This option is suitable for those scenarios where a limited bandwidth is available, and a normal firmware update procedure would result in a timeout. For example, one may want to configure a remote collector constrained by a 50Kbps bandwidth to receive the updates at 24Kbps. This configuration will prevent the saturation of the communication channel and thus it will allow to send data traffic while receiving an update.

Enable or disable slow update

Products	CMC, Guardian, Remote Collector
Syntax	`software_update_slow_mode [true|false]`
Description	This is a global switch that enables (true) or disables (false) the feature. When the feature is disabled all the other switches are ignored.
Where	**CLI**
To apply	It is applied automatically

Set the transfer chunk size

Products	CMC, Guardian, Remote Collector
Syntax	`software_update_slow_mode_chunk_size <size_in_bytes>`
Description	The update bundle is split into multiple fixed-sized chunks. Chunks are individually transmitted, verified and reassembled. In case of a failed delivery, only invalid chunks are retransmitted. Essentially, big chunks are more suitable for high-speed networks, while smaller chunks are to be preferred for more effective bandwidth limitation.
Parameters	`size_in_bytes`: the chunk size in bytes. Values are normalized to stay in the range [128, 10485760]. Default value is 4096.
Where	**CLI**
To apply	It is applied automatically

Set the transfer speed

Products	CMC, Guardian, Remote Collector
Syntax	`software_update_slow_mode_max_speed <speed_in_bps>`

Description	Sets the maximum allowed speed for update transfer.
Parameters	`speed_in_bps`: The maximum allowed speed in bytes per second. Values lower than 1024 are normalized to 1024. The default value is 4096. Notice that small chunks add some slight overhead, so generally the transfer speed will remain consistently below the declared limit.
Where	**CLI**
To apply	It is applied automatically

Chapter 25. Session hijacking



Configure session hijacking protection

Web management interface protects itself from session hijacking attacks binding web session to ip addresses and browser configurations. When it detects differences on these parameters it automatically destroys the session and records the error in the audit log. This feature is enabled by default and it can be disabled using this configuration:

Disable session hijacking protection

Products	CMC, Guardian
Syntax	`conf.user configure ui session protection [true|false]`
Description	Enable (option true, default behavior) or disable (option false) session hijacking protection.
Where	**CLI**
To apply	It is applied automatically

When closing sessions the web management interface will record in the audit log this error text

```
Session hijacking detected, closing session
```

and the details of the affected session.



Chapter 26. Passwords



Configure passwords

This topic describes N2OS password parameters and their default values. Modifying the default password requires that you change the configuration of the sensor.

Set maximum attempts

Product
Guardian

```
conf.user configure password_policy maximum_attempts <attempts>
```

Description
Set the maximum number of attempts for inserting a password, before the system locks.

Parameters
`attempts`: The maximum number of attempts allowed (default: 3)

Where
CLI

To apply
It is applied automatically.

Enable pbkdf2 sha512 password hashing algorithm

Product
Guardian

```
conf.user configure password_policy hashing_pbkdf2 [true|false]
```

Description
Enable or disable the pbkdf2 sha512 password hashing algorithm (default: false).

Where
CLI

To apply
It is applied automatically.

Set lock time

Product
Guardian

```
conf.user configure password_policy lock_time <minutes>
```

Description
Set the maximum number of attempts for inserting a password, before the system locks.

Parameters
`minutes`: The number of minutes for which a user account is locked out after having failed to login for the maximum attempts (default: 5)

Where
CLI

To apply
It is applied automatically.

Set history

Product
Guardian

```
conf.user configure password_policy history <number>
```

Description
Set the number of historical passwords that are required to be unique (default: 3).

Parameters
`number`: The number of unique passwords to be used.

Where
CLI

To apply
It is applied automatically.

Set password digits

Product
Guardian

```
conf.user configure password_policy digit <number>
```

Description
Sets the minimum number of digits that are required to be contained in a password (default: 1).

Parameters
`number`: The number of digits required.

Where
CLI

To apply
It is applied automatically.

Set password lower

Product
Guardian

```
conf.user configure password_policy lower <number>
```

Description
Sets the minimum number of lowercase characters that are required to be contained in a password (default: 1).

Parameters
`number`: The number of lowercase characters required.

Where
CLI

To apply
It is applied automatically.

Set password upper

Product
Guardian

```
conf.user configure password_policy upper <number>
```

Description
Sets the minimum number of uppercase characters that are required to be contained in a password (default: 1).

Parameters
`number`: The number of uppercase characters required.

Where
CLI

To apply
It is applied automatically.

Set password symbol

Product
Guardian

```
conf.user configure password_policy symbol <number>
```

Description
Sets the minimum number of symbol characters that are required to be contained in a password (default: 0).

Parameters
`number`: The number of symbol characters required.

Where
CLI

To apply
It is applied automatically.

Set password min length

Product
Guardian

```
conf.user configure password_policy min_password_length <number>
```

Description
Sets the minimum length required for a password (default: 12).

Parameters
`number`: The minimum length.

Where
CLI

To apply
It is applied automatically.

Set password max length

Product
Guardian

```
conf.user configure password_policy max_password_length <number>
```

Description
Sets the maximum length required for a password (default: 128).

Parameters
`number`: The maximum length.

Where
CLI

To apply
It is applied automatically.

Enable expire for inactive users

Product
Guardian

```
conf.user configure password_policy inactive_user_expire_enable [true|
false]
```

Description
Enable or disable the expiration for inactive users (default: false).

Parameters
`number`: The maximum length.

Where
CLI

To apply
It is applied automatically.

Set user lifetime

Product
Guardian

```
conf.user configure password_policy inactive_user_lifetime <number>
```

Description
Sets the required inactive days after which a user is forced as disabled (default: 60).

Parameters
`number`: The number of days.

Where
CLI

To apply
It is applied automatically.

Enable expiration for admin users

Product
Guardian

```
conf.user configure password_policy admin_can_expire [true|false]
```

Description
Enable or disable the expiration for inactive admin users (default: false).

Where
CLI

To apply
It is applied automatically.

Enable password expiration

Product
Guardian

```
conf.user configure password_policy password_expire_enable [true|false]
```

Description
Enable or disable the expiration for passwords (default: false).

Where
CLI

To apply
It is applied automatically.

Set password lifetime

Product
Guardian

```
conf.user configure password_policy password_lifetime <number>
```

Description
Sets the required number of days after which a password change is enforced (default: 90).

Parameters
`number`: The number of days.

Where
CLI

To apply
It is applied automatically.

Chapter 27. Sandbox



Sandbox archive processing

A description of sandbox archive processing.

When a sandbox file contains an archive, the archive is uncompressed and each extracted file is processed to check the presence of malware using the YARA rules and indicators. The process is repeated recursively for each extracted file to eventually check for malware in nested archives. The 'archive' configurations commands listed below permits to control how this unpacking and checking process is performed. Since the unpacking and checking can consume significant resources, the tuning of these parameters can be important when a large number of potentially large files is present in the sandbox.

Set the number of workers

```
conf.user configure sandbox dispatcher number_of_workers <value>
```

Product
Guardian

Description
Service will start one dispatcher application and as many worker applications as requested. By default, Sandbox will instead start in standalone mode.

Parameters
`value`: Maximum value of 8.

Where
CLI

To apply
It is applied automatically.

Set the size of the sandbox tmpfs partition

```
conf.user configure sandbox tmpfs sandbox <value>
```

Product
Guardian

Description
The tmpfs in-memory partition should be big enough to host two times the maximum number of retained files.

Parameters
`value`: Size in MB of the partition. Defaults to 400MB.

Where
CLI

To apply
It is applied upon the reboot of the sensor.

Set the size of the sandbox tmpfs temporary partition

```
conf.user configure sandbox tmpfs tmp_sandbox <value>
```

Product
Guardian

Description
For high throughputs, the tmpfs in-memory partition should be big enough to host all the files that can fit in the sandbox tmpfs partition described in the previous command.

Parameters
`value`: Size in MB of the partition. Defaults to 100MB.

Where
CLI

To apply
It is applied upon the reboot of the sensor.

Set the size of the sandbox tmpfs pipes partition

```
conf.user configure sandbox tmpfs pipes <value>
```

Product
Guardian

Description
10MB should be allocated for every worker that has been configured.

Parameters
`value`: Size in MB of the partition. Defaults to 10MB.

Where
CLI

To apply
It is applied upon the reboot of the sensor.

Set sandbox strategies for vulnerability assessment

Product
Guardian

```
conf.user configure sandbox strategies <json_value>
```

Description
The schema for the configuration options is:

```
{"enabled_strategies": ["yara_rules", "stix_indicators"]}
```

Parameters
`json_value`: Define which strategies will be used to analyze files.

Where
CLI

To apply
It is applied automatically.

Set the minimum percentage of free disk

Product
Guardian

```
conf.user configure sandbox min_disk_free <value>
```

Description
Minimum free disk percentage that should be observed in the tmpfs /var/sandbox folder, where `n2os_ids` will write the captured files.

Parameters
`value`: Default value 10

Where
CLI

To apply
It is applied automatically.

Set the maximum number of files to retain

Product
Guardian

```
conf.user configure sandbox max_files_to_retain <value>
```

Description
Maximum number of files to retain in the tmpfs /var/sandbox folder, where n2os_ids will write the captured files. The actual number may be up to two times higher under heavy loading in order to improve the performance of the n2os_sandbox process.

Parameters
`value`: Default value 250

Where
CLI

To apply
It is applied automatically.

Set the size of the asynchronous processing queues

Product
Guardian

```
conf.user configure sandbox queues queue_length <value>
```

Description
Length of the asynchronous queues processing and analysing the captured files. This number is applied to both the standalone, dispatcher and worker applications. This figure should be increased to the number of files which are expected to be handled by the application every 250ms. After the application of this setting, also the size of the tmpfs folder partitions needs to be carefully tuned as described in the corresponding commands.

Parameters
`value`: Default value 200

Where
CLI

To apply
It is applied automatically.

Set the interval for stale file analysis

Product
Guardian

```
conf.user configure sandbox dispatcher files_timeout <value>
```

Description
Stale files which have not been analyzed and cleaned by the worker applications under severe loading are garbage collected by default every hour. This is an exceptional situation which should not even occur under heavy loading, but this setting is provided as an additional protection mechanism.

Parameters
`value`: Default value 3600 seconds

Where
CLI

To apply
It is applied automatically.

Set the maximum number of extracted files

Product
Guardian

```
conf.user configure archive max_number_of_files <value>
```

Description
Defines the maximum number of files that can be extracted and processed from an archive file. If the maximum number of extracted files is reached for a file, then additional files nested inside it are neither expanded nor processed.

Parameters
`value`: Default value 100

Where
CLI

To apply
It is applied automatically.

Set the maximum level of nested extracted files

Product
Guardian

```
conf.user configure archive max_levels <value>
```

Description
Defines the maximum level of nested archives that are extracted and processed from an archive file. Files nested in archives at a level deeper than the specified one are neither extracted nor processed.

Parameters
`value`: Default value 3

Where
CLI

To apply
It is applied automatically.

Set the maximum size of a single extracted file

Product
Guardian

```
conf.user configure archive max_single_size <size_in_bytes>
```

Description
Defines the maximum size in bytes of a file that can be extracted and processed from a compressed archive file. If a file is larger than the limit it is neither extracted nor processed.

Parameters
`size_in_bytes`: Default value 200000000 (200M)

Where
CLI

To apply
It is applied automatically.

Set the maximum overall size of extracted files

Product
Guardian

```
conf.user configure archive max_overall_size <size_in_bytes>
```

Description
Defines the overall maximum size of the files that can be extracted from a single compressed archive file. If the overall size of the files extracted from a file exceed this limit, the remaining files in the archive are neither extracted nor processed.

Parameters
`size_in_bytes`: Default value 400000000 (400M)

Where
CLI

To apply
It is applied automatically.

Set the maximum time to be spent for each file extraction

Product
Guardian

```
conf.user configure archive max_wait_msec <time_in_millisecs>
```

Description
Defines the maximum time to be spent during a file extraction from a compressed archive file. If the spent time exceeds the limit, the extraction is aborted.

Parameters
`time_in_millisecs`: Default value 10000 millisecs

Where
CLI

To apply
It is applied automatically.

Define maximum recursion depth allowed for VBA extraction

Product
Guardian

```
conf.user configure archive vba_macro_extraction max_recursion <value>
```

Description
VBA archives unzipping could require a high number of recursive steps to complete full recursion: this could lead to possible performance issues due to high number of memory allocations. This parameter limits the number of recursions allowed to unzip a single file.

Parameters
`value`: Default value 100

Where
CLI

To apply
It is applied automatically.

Enable or disable the adaptive algorithm for file unzipping

Product
Guardian

```
conf.user configure archive auto_switch_off <flag>
```

Description
Unzipping is a very expensive process which is automatically disabled when Sandbox is under heavy loading. Instead of discarding files, Sandbox will disable unzipping for some files and process only the unzipped file with STIX and YARA indicators.

Parameters
`flag`: Default to true. false to disable the feature

Where
CLI

To apply
It is applied automatically.

Configuring handling of sandboxed zipped files

Product
Guardian

```
conf.user configure sandbox unzipping <json_value>
```

Description
The json object can have the following attributes: * modes - array of unzipping modes which should be enabled. By default all of them are enabled and are executed in the described order. Possible values are: fast, for fast unzipping, macro, for macro extraction and analysis, upx, for upx decompression, full, for extensive and advanced archive decompression. An empty array can be used to completely disable the unzipping functionalities of Sandbox.

```
conf.user configure sandbox unzipping {"modes": ["macro", "upx",
"full"]}
```

Parameters
`json_value`: A json object to configure how zipped files are handled by Guardian

Where
CLI

To apply
It is applied automatically.

Configure high throughput protection

```
conf.user configure sandbox extraction <json_value>
```

Product
Guardian

Description
This option lets you define which files have to be analysed by Sandbox. The `enable_...` suboptions enable only the files that satisfy the specific criteria, while the `disable_...` suboptions disable only the files that match the specific criteria and enable all the others. In the case that several criteria are specified they are applied with an AND sequence (i.e. all of them have to be satisfied). The user should note in the following that advertised file extensions are considered. If an attacker hides behind a *joint photographic experts group (JPEG)* file extension a malicious executable, there is no way for Sandbox to understand that the file is an executable without performing an in-depth analysis on the file itself. For this reason, we highly discourage the use of the file extension attribute in the *JSON* below. Protocols, zones and node criteria are instead encouraged, when even the auto switch off adapative algorithm cannot provide a sufficient protection against high throughputs.

The json object can have the following attributes:

- `enabled_protocols` - only files extracted from these protocols will be analysed
- `disabled_protocols` - files extracted from these protocols will be excluded from the analysis
- `enabled_file_extensions` - only files extracted with these advertised extensions will be analysed
- `disabled_file_extensions` - files extracted with these advertised extensions will be excluded from the analysis
- `enabled_zones` - only files extracted from the specified zones (both source and destination) will be analysed.
- `disabled_zones` - files extracted from the specified zones (both source and destination) will be excluded from the analysis.
- `enabled_src_zones` - only files extracted from the specified source zones will be analysed.
- `disabled_src_zones` - files extracted from the specified source zones will be excluded from the analysis.
- `enabled_dst_zones` - only files extracted from the specified destination zones will be analysed.
- `disabled_dst_zones` - files extracted from the specified destination zones will be excluded from the analysis.
- `enabled_node_types` - only files extracted from nodes of the specified types will be analysed (both source and destination).
- `disabled_node_types` - files extracted from nodes of the specified types will be excluded from the analysis (both source and destination).
- `enabled_src_node_types` - only files extracted from packets with source nodes of the specified types will be analysed.
- `disabled_src_node_types` - files extracted from packets with source nodes of the specified types will be excluded from the analysis.
- `enabled_dst_node_types` - only files extracted from packets with destination nodes of the specified types will be analysed.
- `disabled_dst_node_types` - files extracted from packets with destination nodes of the specified types will be excluded from the analysis.
- `enabled_node_ids` - only files extracted from nodes with the specified ids will be analysed (both source and destination).
- `disabled_node_ids` - files extracted from nodes with the specified ids will be excluded from the analysis (both source and destination).
- `enabled_src_node_ids` - only files extracted from packets with source nodes contained in the specified ids will be analysed.
- `disabled_src_node_ids` - files extracted from packets with source nodes contained in the specified ids will be excluded from the analysis.
- `enabled_dst_node_ids` - only files extracted from packets with destination nodes contained in the specified ids will be analysed.
- `disabled_dst_node_ids` - files extracted from packets with destination nodes contained in the specified ids will be excluded from the analysis.

```
conf.user configure sandbox extraction {"enabled_protocols": ["http"]}
```

Parameters

`json_value`: A *JSON* object to configure which files are not analysed by Sandbox.

Where
CLI

To apply
It is applied automatically.



Chapter 28. Miscellaneous



Configure SSH key update interval

```
conf.user configure ssh_key_update interval <seconds>
```

Product
Guardian

Description
See **Add an SSH key for an admin user**, in the **Maintenance Guide**.

Parameters
`seconds`: Number of seconds between propagations.

Where to apply
CLI

How to apply
It is applied automatically.

Configure paranoid mode for user login authentication

```
conf.user configure authentication paranoid_mode [true|false]
```

Product
Guardian

Description
Paranoid mode in authentication is enabled by default. It is used to control the disclosure of information about the existence of a user during the login authentication process and to normalize the login response time. When this setting is disabled, after several failed login attempts, the user is warned with a message about the remaining attempts before the account gets locked. As a consequence, the user information can be leaked. However, when this setting is enabled, there is no warning message and thus no potential for user information leak.

Where to apply
CLI

How to apply
In a shell console execute: `service webserver stop`

Configure identity provider

```
conf.user configure cmc identity_provider_url <url>
```

Product
CMC

Description
Expose identity provider endpoint.

Parameters
`url`: The of the identity provider.

Where to apply
CLI

How to apply
In a shell console execute: `service webserver stop`



Chapter 29. N2OS Maintenance



Hardware information

Show CPU information

Shows the CPU model, number of cores, frequency, and architecture of the sensor.

Product	CMC, Guardian, Remote Collector
Syntax	`n2os-hwinfo cpu --show [--json]`
Description	Shows the CPU model, number of cores, frequency, and architecture.
Parameters	`--json`: Returns the output in JSON format (must be used with `--show`)
Where	The text console (access directly or through secure shell (SSH)). The user must have root privileges.
To apply	Enter the command and press enter.
Platform availability	Available on on-premises appliances and container installations. Depending on the platform, some values may not be available and are shown as `-` (for example, the CPU frequency).

Show disk information

Shows information about the disks detected on the system.

Product	CMC, Guardian, Remote Collector
Syntax	`n2os-hwinfo disk --show [--verbose] [--json]`
Description	Shows information about the detected disks. With `--verbose`, shows full SMART details and ZFS counters (table format only).
Parameters	`--verbose`: Shows full SMART details and ZFS counters (table format only). `--json`: Returns the output in JSON format (must be used with `--show`)
Where	The text console (access directly or through secure shell (SSH)). The user must have root privileges.
To apply	Enter the command and press enter.
Platform availability	Available only on on-premises appliances. In container installations, the command returns no disk information.

Show RAM information

Shows information about the memory of the sensor.

Product	CMC, Guardian, Remote Collector
Syntax	`n2os-hwinfo ram --show [--json]`
Description	Shows information about the memory of the sensor.
Parameters	`--json`: Returns the output in JSON format (must be used with `--show`)
Where	The text console (access directly or through secure shell (SSH)). The user must have root privileges.
To apply	Enter the command and press enter.
Platform availability	Available on on-premises appliances and container installations. On on-premises appliances, the command shows the installed memory modules. In container installations, it shows the total memory available to the container and the configured limit, if any.

Show motherboard information

Shows information about the motherboard of the sensor, including BIOS/UEFI, PCI, and ACPI characteristics.

Product	CMC, Guardian, Remote Collector
Syntax	`n2os-hwinfo mobo --show [--json]`
Description	Shows information about the motherboard, including BIOS/UEFI, PCI, and ACPI characteristics.
Parameters	`--json`: Returns the output in JSON format (must be used with `--show`)
Where	The text console (access directly or through secure shell (SSH)). The user must have root privileges.
To apply	Enter the command and press enter.
Platform availability	Available on on-premises appliances and container installations. In container installations, motherboard characteristics are not available and are shown as `No data available`

Show network interface information

Shows the network cards detected on the system, with their location, type, driver, number of ports, and name.

Product	CMC, Guardian, Remote Collector
Syntax	`n2os-netconf cards --show [--all] [--json]`
Description	Shows the network cards detected on the system with their location, type, driver, number of ports, and name. With `--all`, lists all network cards detected on the system, not only slot-based ones.
Parameters	`--all`: With `--show`, lists all network cards detected on the system, not only slot-based ones. `--json`: Returns the output in JSON format (must be used with `--show`)
Where	The text console (access directly or through secure shell (SSH)).
To apply	Enter the command and press enter.
Platform availability	On on-premises appliances, `--show` lists the installed expansion cards. In container installations, it shows no expansion card data. With `--all`, both platforms list all detected network interfaces, including virtual ones in container installations.

Show PSU information

Shows information about the PSU of the sensor.

Product	CMC, Guardian
Syntax	`n2os-hwinfo psu --show`
Description	Shows information about the PSU of the sensor.
Parameters	`--show`
Where	The text console (access directly or through secure shell (SSH)). The user must have root privileges.
To apply	Enter the command and press enter.
Platform availability	Available on the Nozomi Networks sensor NS20.

Glossary



Berkeley Packet Filter

The BPF is a technology that is used in some computer operating systems for programs that need to analyze network traffic. A BPF provides a raw interface to data link layers, permitting raw link-layer packets to be sent and received.

Central Management Console

The Central Management Console (CMC) is a Nozomi Networks product that has been designed to support complex deployments that cannot be addressed with a single sensor. A central design principle behind the CMC is the unified experience, that lets you access information in a similar way as on the sensor.

Command-line interface

A command-line processor uses a command-line interface (CLI) as text input commands. It lets you invoke executables and provide information for the actions that you want them to do. It also lets you set parameters for the environment.

cURL

cURL is a command-line tool and library used to transfer data to or from a server. It supports many protocols, including HTTP, HTTPS, FTP, and more. In API contexts, curl is commonly used to test and interact with endpoints by sending requests and receiving responses.

Industrial Control Systems

An ICS is an electronic control system and related instrumentation that is used to control industrial processes.

IPv6

Internet Protocol version 6 is the most recent version of the Internet Protocol, designed to replace IPv4. It uses 128-bit numerical addresses to identify devices, allowing for a vastly larger address space.

JavaScript Object Notation

JSON is an open standard file format for data interchange. It uses human-readable text to store and transmit data objects, which consist of attribute–value pairs and arrays.

Joint Photographic Experts Group

JPEG, or JPG, is a method of lossy compression that is used for digital images. The degree of compression can be adjusted, allowing a selectable tradeoff between storage size and image quality.

Secure Shell

A cryptographic network protocol that let you operate network services securely over an unsecured network. It is commonly used for command-line execution and remote login applications.

Universal Serial Bus

Universal Serial Bus (USB) is a standard that sets specifications for protocols, connectors, and cables for communication and connection between computers and peripheral devices.

Variable

In the context of control systems, a variable can refer to process values that change over time. These can be temperature, speed, pressure etc.

Zettabyte File System

A file system and logical volume manager that combines storage pooling with data integrity features. N2OS uses ZFS for the `/data` file system.

