



QUICK START GUIDE

Nozomi Networks

Guardian Community Edition

Notice

Legal notices

Publication Date

March 2020

Copyright

Copyright © 2013-2020, 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.

Table of Contents

Legal notices.....	iii
Chapter 1: Installation.....	7
Installing on Virtual Hardware.....	8
Setup Phase 1.....	10
Setup Phase 2.....	12
Additional settings.....	14
Chapter 2: Basics.....	15
Environment.....	16
Asset.....	16
Node.....	16
Session.....	17
Link.....	17
Query.....	18
Protocol.....	18
Chapter 3: User Interface Reference.....	19
Supported Web Browsers.....	20
Navigation header.....	20
Asset View.....	21
Network View.....	22
Queries.....	32
System.....	35
Queries.....	43
Chapter 4: Queries.....	47
Overview.....	48
Reference.....	49
Examples.....	58
Chapter 5: Maintenance.....	63
System Overview.....	64
Data Backup and Restore.....	65
Reboot and shutdown.....	66
Software Update and Rollback.....	67
Data Factory Reset.....	69
Support.....	69
Chapter 6: Scriptable protocols.....	71
Setup.....	72
Writing a scriptable protocol.....	73
API reference.....	77

Chapter 1

Installation

Topics:

- [Installing on Virtual Hardware](#)
- [Setup Phase 1](#)
- [Setup Phase 2](#)
- [Additional settings](#)

In this chapter you will receive the fundamental information necessary to get both Nozomi Networks Solution physical and virtual appliances up and running.

Further information on additional configuration is given in the [Configuration](#) chapter.

Maintenance tasks are described in the [Maintenance](#) chapter.

Installing on Virtual Hardware

Installation on Virtual Hardware has been tested on a variety of OVA-compatible environments. However, the current release of N2OS officially supports these hypervisors:

1. VMware ESXi 5.5 or newer
2. HyperV 2012 or newer

The minimum requirements for a Guardian Virtual Machine (VM) resources are:

- 4 vCPU running at 2 Ghz
- 4 GB of RAM
- 10 GB of minimum disk space, running on SSD or hybrid storage (100+ GB of disk recommended)
- 2 or more NICs (maximum number depends on hypervisor), one will be used for management and the 1 or more other for traffic monitoring

Ensure that all these resources are provided in healthy conditions. Overall hypervisor load must be under control and no ballooning should regularly occur on the Guardian VM, otherwise unexpected behavior of the system may be experienced, such as dropped packets or overall poor system performance.

Installing the Virtual Machine

In this section we will cover the installation of the Virtual Machine into the hypervisor. A running VM will be obtained, however further configuration enabling external access will be provided in subsequent sections.

In order to proceed you should be familiar with importing OVA Virtual Machine in your hypervisor environment. Should this not be the case, please refer to the manual or support service of your hypervisor.

1. Import the Virtual Machine into the hypervisor and configure resources according to the minimum requirements specified in the previous section.
2. After importing the VM, go to the hypervisor settings of the VM disk and set a desired size. Some hypervisors, for instance VMware ESX ≥ 6.0 , allow to change the disk size at this stage. With hypervisors that do not allow this operation, you must STOP HERE with this section and proceed with instructions contained in [Adding a secondary disk to Virtual Machine](#) on page 8.
3. Boot the VM. It will now boot into a valid N2OS environment.
4. Login as `admin`
You will be instantly logged in, no password is set by default.
5. Go to privileged mode with the command:

```
enable-me
```

You will now be able to perform changes into the system.

Adding a secondary disk to Virtual Machine

In this section we will cover how to add a bigger virtual data disk to the N2OS VM, in case the main disk could not be grown during the first import. In order to proceed you should be familiar with managing virtual disks in your hypervisor environment. Otherwise please refer to the manual or support service of your hypervisor.

1. Add a disk to the VM and restart it
2. In the VM console, use the following command to obtain the name of the disk devices:

```
sysctl kern.disks
```

3. Assuming `ada1` is the device disk added as secondary disk (note that `ada0` is the OS device), execute this command to move the data partition to it

```
data_move ada1
```


Adding a monitoring interface to the Virtual Machine

By default the VM has one management network interface and one monitoring interface. Depending on deployment needs, it may be useful to add more monitoring interfaces to the appliance. To add one or more interfaces, follow these steps:

1. If the VM is powered on, shut it down
2. Add one or more network interfaces from the hypervisor configuration
3. Power on the VM

The newly added interface(s) will be automatically recognized and used by the Guardian.

Setup Phase 1

We will now setup the very basic configuration needed to start using the Nozomi Networks Solution. After these steps the system will have the management interface setup and reachable as text console via SSH and as web console via HTTPS.

We assume that Nozomi Networks Solution has already been installed and ready to be configured for the first time. Depending on the case, a serial console must be used in this phase (for Physical Appliances) or the text hypervisor console (for Virtual Appliances).

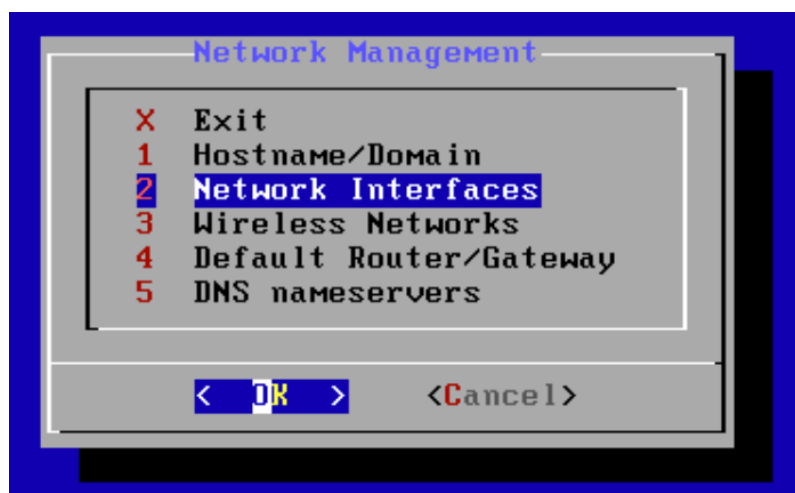
1. The console will display a prompt with the text "N2OS - login:". Type `admin` and then press [Enter]. In the Virtual Appliance, you will be instantly logged in, as no password is set by default. In Physical Appliances, `nozominetworks` is the default password.
2. Elevate the privileges with the command: `enable-me`
3. Now launch the initial configuration wizard with the command: `setup`

```
root@nozomi-ids:~ # setup
```

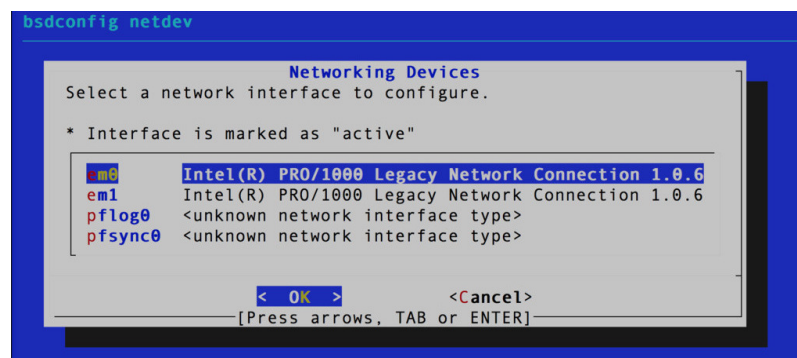
4. You will be prompted to choose the admin password first. Select a strong password as this will allow the admin user to access the appliance through SSH.

```
You must set the password for the 'admin' console user.
Changing local password for admin
You can now choose the new password.
A valid password should be a mix of upper and lower case letters,
digits and other characters. You can use an 8 character long
password with characters from at least 3 of these 4 classes, or
a 7 character long password containing characters from all the
classes. Characters that form a common pattern are discarded by
the check.
Alternatively, if noone else can see your terminal now, you can
pick this as your password: "motor.church!half".
Enter new password: 
```

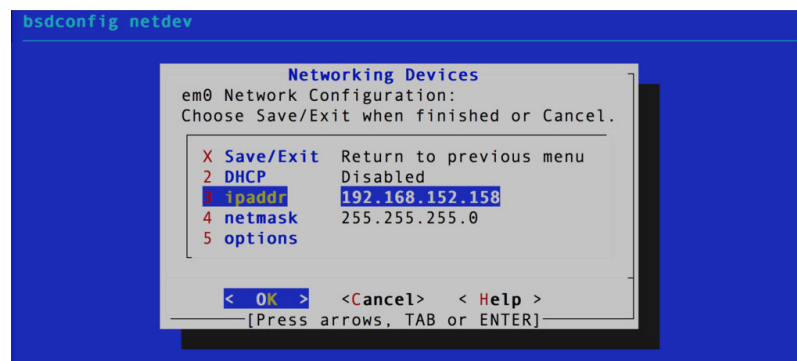
5. Secondly, you will need to setup the management interface IP address. Select the "2 Network Interfaces" menu in the dialog.



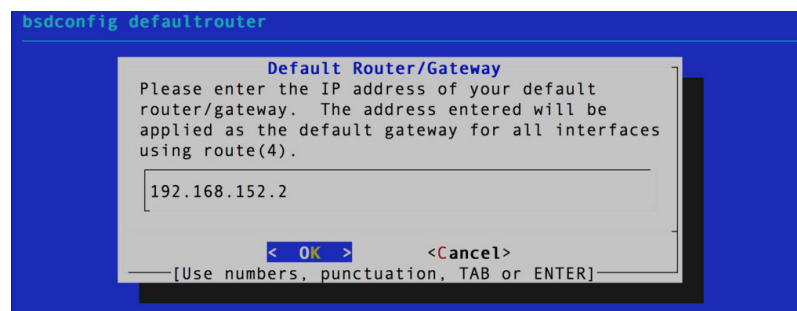
6. Now you will need to setup the management interface IP address. Depending on the appliance model, the management interface can be named `em0` or `mgmt`. Select it and press [Enter].



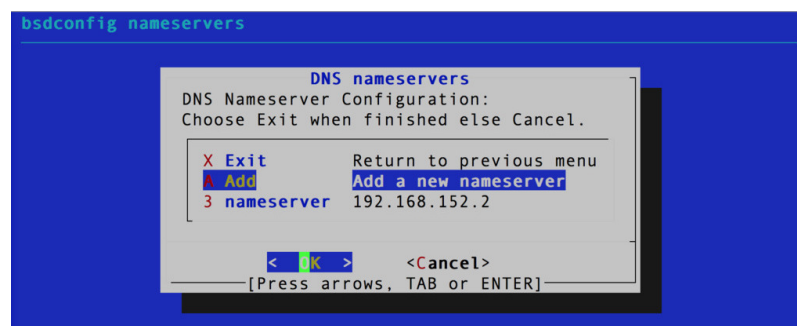
7. Edit the values for IP address (ipaddr) and Netmask (netmask). Enable DHCP to configure all automatically. Then move up to "X. Save/Exit" and press [Enter].



8. Now select "Default Router/Gateway" from the menu, and enter the IP address of the default gateway. Press [Tab] and then [Enter] to save and exit.



9. Now select "DNS nameservers" from the menu, and configure the IP addresses of DNS servers.

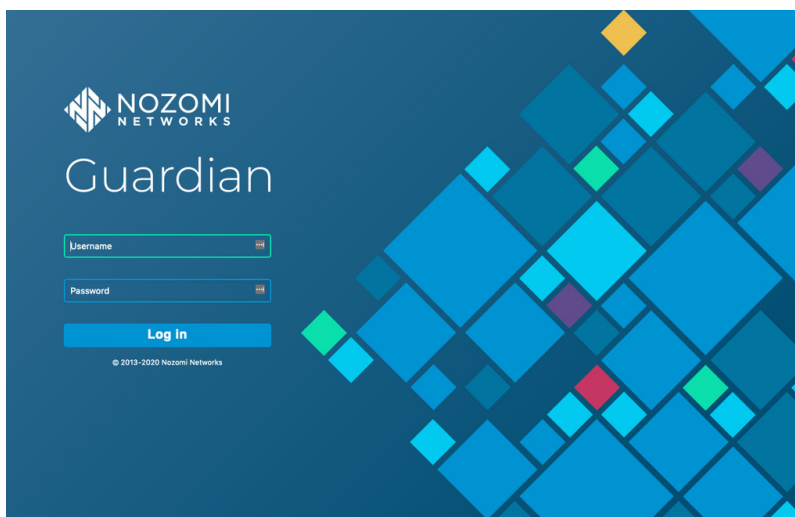


10. Move up to "X Exit" and press [Enter].
11. The basic networking setup is done; the remaining steps will be performed by opening the web console running on the management interface.

Setup Phase 2

This second phase of the setup will be performed with the web console. Before starting to use the web console, be sure to use one of the [supported web browsers](#).

The web console can be accessed pointing at `https://<appliance_ip>` where `<appliance_ip>` is the IP address assigned to the management interface. Please note that the product integrates self-signed SSL certificates to get started, so add an exception in your browser. Later in this chapter we will provide steps to import valid ones. You should now see the login screen:



Default username and password are admin / nozominetworks. For security reasons you will be prompted to change these credentials at first login.

Once logged in, the remaining steps of the setup can be completed. Go to **Administration > General** and change the host name.

Now fix date and time settings. Go to **Administration > Date and time**, and change the time zone, set the date and (optional) enable the NTP client.

Date settings

Timezone

Local: CET (UTC+02:00)

Save

Date (format ccyyymmddHHMM.ss)

Pick a date

Set as client

201907250932.16

Save

NTP

☐ Enabled

Servers

0.freebsd.pool.ntp.org,1.freebsd.pool.ntp.org

Save

The appliance is almost ready to be put into production: next step is to install a valid license.

License

In the `Administration > Updates & Licenses` page, you will need to set a new license. First copy the machine ID, then you can use it along with the Activation Code that you have received from Nozomi Networks to obtain a license key. Once you have a valid license key, paste it inside the text box. After confirmation, the appliance begins to monitor the configured network interfaces.

Updates & Licenses

Update service configuration

Base

Set new license

Current license - Full: monitoring and security

License status: ok
Licensee: Nozomi Engineering
License ID: nozomi id

Machine ID: 387d7a36-d6c6-4979-a4f6-4824a25d9ddc [Copy](#)

License key

Verify and apply

Figure 1: The License page

Additional settings

In this chapter some additional, non-mandatory settings of the system will be explained.

Install SSL certificates

In this section we will import a real SSL certificate into the appliance, needed to securely encrypt all traffic between client computers and the N2OS appliance over HTTPS.

The N2OS webserver that exposes the HTTPS interface is nginx. Please, be prepared with a certificate and a key file both compatible with NGINX and name them `https_nozomi.crt` and `https_nozomi.key`.

1. Upload the certificate and key file to the appliance with an SSH client in the `/data/tmp` folder. For example, given you have `https_nozomi.crt` and `https_nozomi.key` in the same folder, open a terminal, cd into it and then upload

```
scp https_nozomi.* admin@<appliance_ip>:/data/tmp
```

2. Log into the text-console, either directly or through SSH then elevate the privileges

```
enable-me
```

3. Execute the command `n2os-addtlscert`

```
n2os-addtlscert https_nozomi.crt https_nozomi.key
```

4. Now restart nginx by issuing the command

```
service nginx restart
```

5. Verify that the certificate is correctly loaded by pointing your browser to `https://<appliance_ip>/` and checking that the certificate is now recognized as valid.
6. We can safely save the new setup by issuing this command in the console

```
n2os-save
```

Now the imported SSL certificates are correctly working and will be applied also on next reboot.

Chapter

2

Basics

Topics:

- [*Environment*](#)
- [*Asset*](#)
- [*Node*](#)
- [*Session*](#)
- [*Link*](#)
- [*Query*](#)
- [*Protocol*](#)

In the chapter you will get introduced to some basic concepts of the Nozomi Networks Solution and some recurring graphical interface controls will be explained.

You must have mastered these concepts in order to understand how to properly use and configure the N2OS system.

Environment

The Nozomi Networks Solution **Environment** is the real time representation of the network monitored by the Guardian, providing a synthetic view of all the assets, all the network nodes and the communications between them.

Asset View

In the Asset View section are displayed all your assets, intended as single discrete endpoints. In this section it is easy to visualize, find and drill down on asset information such as hardware and software versions.

For more details see [Asset View](#) on page 21

Network View

In the Network View section are contained all the generic network information which are not related to the SCADA side of some protocols like the list of nodes, the connection between nodes and the topology.

For more details see [Network View](#) on page 22

Process View

In the Process View section are contained all the SCADA specific information like the SCADA slaves list, the slave variables with their history of values and other related information, a section with the analysis on the variables values and some variables related statistics.

Asset

An **asset** in the Environment represents an actor in the network communication and, depending on the nodes and components involved, it can be something ranging from a simple personal computer to an OT device.

All the assets are listed in the `Environment > Asset View > List` section and can also be viewed in a more graphical way in the `Environment > Asset View > Diagram` section which aggregates the assets in different levels.

Page 1 of 7,165 entries

Export Confirmed MACs only ☒ Live ☒ 8 selected

ACTIONS	NAME	TYPE	OS/FIRMWARE	IP	MAC ADDRESS	MAC
	MACH-666666	switch	Firmware: 09.0.06		[multiple]	Hirschman
	ACMEIncHQ_SW1	switch	Firmware: h.10.38		[multiple]	ProCurve N
	172.16.66.53	computer	Windows XP SP3	172.16.66.53		
	ACMEIncHQ_SW2	switch	Firmware: V05.01.03		[multiple]	Siemens Ar
	192.168.162.22	computer	Windows XP SP3	192.168.162.22		
	Modicon M340 BMX P34 2020	PLC	Firmware: 2.90	172.16.0.157		
	Modicon M340 BMX P34 2020	PLC	Firmware: 2.90	172.16.1.174		

Figure 2: An example list of assets

Node

A **node** in the Environment represents an actor in the network communication and, depending on the protocols involved, it can be something ranging from a simple personal computer to an RTU or a PLC.

All the nodes in the Environment are listed in the `Environment > Network View > Nodes` section or can be viewed in a more graphical way in the `Environment > Network View > Graph` section.

When a node is involved in a communication using SCADA protocols it can be a master or a slave. SCADA slaves can be analyzed in detail in the `Environment > Process View` section.

Network view								Nodes	Links	Sessions	Graph	Traffic
Page 1 of 8,185 entries / sorted by roles: asc x								Export Live 9 selected ▾				
ACTIONS	ADDRESS	LABEL	ROLES	MAC ADDRESS	SENT BYTES	RECEIVED BYTES	# LINKS	PROTOCOLS				
	172.16.0.1		dns_server	c4:6e:1f:92:ed:d8	8.7 KB	0.0 B	3	dns				
	192.168.1.1		dns_server	c4:6e:1f:92:ed:d8	16.6 KB	0.0 B	3	dns				
	10.1.1.1	HISTORIAN-01	historian	d8:9d:b9:00:17:ec	2.1 KB	377.9 KB	7	browser, pi-connect				
	172.16.0.253		master	00:04:23:e0:04:1c	3.0 MB	1.0 MB	26	dns, modbus, pi-connect, smb, vnc				
	192.168.1.2		master	09:00:09:00:01:12	1.7 MB	869.0 KB	50	dns, iec104, pi-connect, smb, vnc				
	192.168.1.11		master	18:66:da:00:01:11	1.6 MB	570.6 KB	41	dns, iec104, pi-connect, smb, vnc				
	172.16.1.253		master	00:04:23:e0:04:1c	3.0 MB	1.0 MB	26	dns, modbus, pi-connect, smb, vnc				
	192.168.162.22		master	09:00:09:00:01:12	304.6 KB	361.8 KB	12	dns, ethernetip, pi-connect, smb				
	172.16.0.101		master	10:c3:7b:4c:83:17	1.7 MB	716.2 KB	4	dce-rpc, dns, opc, smb, vnc				
	00:60:78:00:6a:10		other	00:60:78:00:6a:10	240.0 B	120.0 B	0	-				
	10.4.1.32		other	b4:a3:82:02:66:00	54.5 KB	2.6 KB	1	rtsp				
	ff:ff:ff:ff:ff:ff		other	ff:ff:ff:ff:ff:ff	0.0 B	282.0 B	0	-				
	ec:74:ba:66:66:60	MACH-666666	other	ec:74:ba:66:66:60	0.0 B	0.0 B	0	-				
	00:16:b9:49:b6:7d	ACMEIncHQ_SW1	other	00:16:b9:49:b6:7d	356.0 B	0.0 B	1	lldp				
	00:16:b9:49:b6:40	ACMEIncHQ_SW1	other	00:16:b9:49:b6:40	188.0 B	0.0 B	0	-				
	10.2.1.255		other	ff:ff:ff:ff:ff:ff	0.0 B	12.5 KB	1	browser				
	172.16.66.53		other	18:a9:05:24:d8:b5	788.0 B	3.8 KB	6	smb				
	00:1b:1b:c0:c8:62	ACMEIncHQ_SW2	other	00:1b:1b:c0:c8:62	0.0 B	0.0 B	0	-				
	01:80:c2:00:00:0e		other	01:80:c2:00:00:0e	0.0 B	3.1 KB	6	lldp				
	00:04:23:e0:04:1c		other	00:04:23:e0:04:1c	240.0 B	120.0 B	0	-				
	10.4.1.36		other	b4:a3:82:06:48:aa	54.5 KB	2.6 KB	1	rtsp				
	10.5.1.253		other	c4:2f:90:a8:c7:79	18.4 KB	381.7 KB	7	rtsp				
	00:50:56:f6:3c:50		other	00:50:56:f6:3c:50	42.0 B	0.0 B	0	-				
	00:0c:29:1d:f2:4a		other	00:0c:29:1d:f2:4a	42.0 B	42.0 B	0	-				
	192.168.162.255		other	ff:ff:ff:ff:ff:ff	0.0 B	777.0 B	1	browser				

Figure 3: An example list of network nodes

Session

A **session** is a semi-permanent interactive information interchange between two or more communicating nodes.

A session is set up or established at a certain point in time, and then turned down at some later point. An established communication session may involve more than one message in each direction.

The Nozomi Networks Solution shows the status of a session depending on the transport protocol, for example a TCP session can be in the SYN or SYN-ACK status before being OPEN.

When a session is closed it will be retained for a certain amount of time and can still be queried to perform subsequent analysis.

All the sessions are listed in the Environment > Network View > Sessions.

Page 1 of 12 293 entries

ExportLive13 selected













ACTIONS	STATUS	FROM	TO	TRANSPORT PROTOCOL	FROM PORT	TO PORT	PROTOCOL	THROUGHPUT	TRANSFERRED BYTES	TRANSFER
 	ACTIVE	10.51.253	10.41.32	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp
 	ACTIVE	10.51.253	10.41.34	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp
 	ACTIVE	10.51.253	10.41.30	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp
 	ACTIVE	10.51.253	10.41.31	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp
 	ACTIVE	10.51.253	10.41.35	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp
 	ACTIVE	10.51.253	10.41.36	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp

Figure 4: An example list of network sessions

Link

A **link** in the Environment represents the communication between two nodes using a specific protocol.

All the links are listed in the Environment > Network View > Link section and can be viewed in a more graphical way in the Environment > Network View > Graph section.

Page 1 of 8,186 entries

Export Live 11 selected

























ACTIONS	FROM	TO	PROTOCOL	LAST ACTIVITY	# ALERTS	THROUGHPUT	TRANSFERRED BYTES	TRANSFERRED PAC
   	00:16:b9:49:b6:7d	01:80:c2:00:00:0e	lldp	2017-02-16 13:57:47.709	0	0.0 b/s	178.0 B	1 pp
   	172.16.66.53	192.168.162.53	smb	2016-11-22 03:00:09.105	0	0.0 b/s	394.0 B	1 pp
   	192.168.162.22	192.168.1.29	ethernetip	2017-01-05 18:13:55.348	0	0.0 b/s	25.4 KB	240 pp
   	192.168.162.22	192.168.1.32	ethernetip	2017-01-05 18:13:55.348	0	0.0 b/s	25.4 KB	240 pp
   	192.168.162.22	192.168.1.33	ethernetip	2017-01-05 18:13:55.348	0	0.0 b/s	25.4 KB	240 pp
   	192.168.162.22	192.168.1.31	ethernetip	2017-01-05 18:13:55.348	0	0.0 b/s	25.4 KB	240 pp

Figure 5: An example list of network links

Query

The N2QL (Nozomi Networks Query Language) syntax is inspired by the most common Linux and Unix terminal scripting languages: the query is a concatenation of single commands separated by the `|` symbol in which the output of a command is the input of the next command. In this way it is possible to create complex data processing by composing several simple operations.

The following example is a query that lists all nodes ordered by received_bytes (in descending order):

```
nodes | sort received.bytes desc
```

For a reference of the graphical user interface or how you can create/edit queries go to the [Query - User interface reference](#)

For a full reference of commands, data sources, and examples of the query language go to the [Query - complete reference](#)

Protocol

In the Environment a link can communicate with one or more protocols. A **protocol** can be recognized by the system simply by the transport layer and the port or by a deep inspection of its application layer packets.

SCADA protocols mapping

All SCADA protocols are recognized by deep packet inspection and for each of them there is a mapping that brings protocol specific concepts to the more generic and flexible Environment Variable model.

As an example of such mappings, consider the following table:

Protocol	RTU ID	Name
Modbus	Unit identifier	(r dr c di)<register address>
IEC 104	Common address	<ioa>-<high byte>-<low byte>
Siemens S7 (Timer or Counter area)	Fixed to 1	(C T)<address>
Siemens S7 (DB or DI area)	Fixed to 1	(DB DI)<db number>.<type>_<byte position>.<bitposition>
Siemens S7 (other areas)	Fixed to 1	(P I Q M L).<type>_<byte position>.<bitposition>
Beckhoff ADS	<AMSNetId Target><AMSPort Target>	<Index Group>/<Index Offset>
and more...		

Chapter

3

User Interface Reference

Topics:

- [*Supported Web Browsers*](#)
- [*Navigation header*](#)
- [*Asset View*](#)
- [*Network View*](#)
- [*Queries*](#)
- [*System*](#)
- [*Queries*](#)

In this chapter we will describe every aspect of the graphical user interface. For each view of the GUI we attached a screenshot with a reference explaining the meaning and the behavior of each interface control.

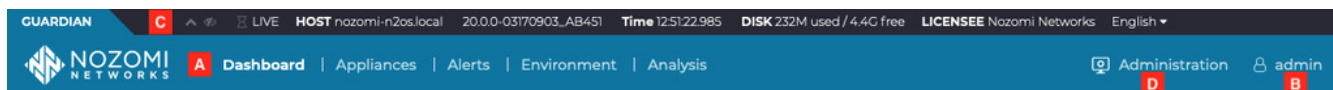
Supported Web Browsers

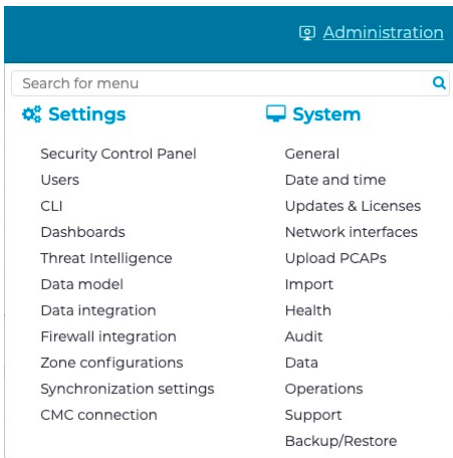
To have the best experience with the Nozomi Networks Solution web console be sure to use one of the following web browsers:

- [Google Chrome](#) version 48 and later
- [Chromium](#) version 48 and later
- [Safari](#) version 9.0 and later (for macOS)
- [Firefox](#) version 49 and later
- Microsoft Internet Explorer version 11
- Microsoft Edge version 12 and later

Navigation header






The navigation bar is always present on the top of the Nozomi Networks Solution user interface. It enables the user to navigate through the pages and it also displays some useful information about the status of the system.



A	The sections of the Nozomi Networks Solution; by clicking on them you will change the page
B	The user menu; by clicking on it you can logout or access the Other actions page
C	<p>The sub navigation bar with:</p> <ul style="list-style-type: none"> • the collapse button: click on it to reduce the height of the navigation bar • the monitoring mode button: click on it to disable the auto logout • the time machine status: it is either LIVE, if the displayed data are realtime, or a timestamp when a time machine snapshot is loaded • the hostname • the N2OS version • the NTP offset • disk statistics, i.e., the used space and the available space • the license information • the language switcher
D	<p>The button that shows the administration menu.</p>  <p>Figure 6: The administration menu</p>

Asset View

Page 1 of 7,165 entries

Export  Confirmed MACs only  Live   8 selected 















ACTIONS	NAME	TYPE	OS/FIRMWARE	IP	MAC ADDRESS	MAC
 	MACH-666666	switch	Firmware: 09.0.06		[multiple]	Hirschman
 	ACMEIncHQ_SW1	switch	Firmware: h.10.38		[multiple]	ProCurve N
 	172.16.66.53	computer	Windows XP SP3	172.16.66.53		
 	ACMEIncHQ_SW2	switch	Firmware: V05.01.03		[multiple]	Siemens At
 	192.168.162.22	computer	Windows XP SP3	192.168.162.22		
 	Modicon M340 BMX P34 2020	PLC	Firmware: 2.90	172.16.0.157		
 	Modicon M340 BMX P34 2020	PLC	Firmware: 2.90	172.16.1.174		

Figure 7: The Assets table

In this page are listed all the Assets using a [table](#). By clicking on an Asset link it is possible to view a popup with some additional details about the asset.

ControlLogix 1756-ENBT/A

IP: 192.168.1.29
 Roles: [slave](#)
 Firmware version: 18.002
 Serial number: 00112232
 Type: PLC

MAC address: 00:0a:dc:85:12:02
 Product name: ControlLogix 1756-ENBT/A
 Vendor: Rockwell Automation/Allen-Bradley
 MAC vendor: RuggedCom inc.

Overview Sessions (0 active) Alerts (0 high, 0 med.) Patches (0 missing) Vulnerabilities (2 high, 15 med.)

Network Stats

Received	13.4 KB	Retransmission	Links
Sent	20.0 KB	0.000%	1
First seen	2017-01-05 18:13	0.0 B in last 30'	active
Last seen	2017-01-05 18:13		

Network Location

Zone	Subnet	Vlan
Undefined	-	-

Protocols

Protocol	Last activity	Inbound	Outbound
ethernetip	2017-01-05 18:13	-	-
		active sessions	active sessions

Learning status

Node is **not** learned
 Asset intelligence **active**

Security Updated on: 2020-03-17

Vulnerabilities **15** Antivirus **-**

Hardware components

- 1756-L61/B LOGIX5561 (Address 0)
 - device_type: Programmable Logic Contrc
 - firmware_version: 20.055
 - serial_number: 00112232
- 1756-RM2/A REDUNDANCY MODULE (Address 1)
 - device_type: Redundancy Module
 - firmware_version: 20.004
 - serial_number: 00010202
- 1756-ENBT/A (Address 3)
 - device_type: Communications Adapter
 - firmware_version: 18.002
 - serial_number: 00112232

Figure 8: The Asset details popup

Network View





































































































Network Nodes

Network view

NodesLinksSessionsGraphTraffic

Page 1 of 8, 185 entries / sorted by roles: asc

ExportLive9 selected

ACTIONS	ADDRESS	LABEL	ROLES	MAC ADDRESS	SENT BYTES	RECEIVED BYTES	# LINKS	PROTOCOLS
   	172.16.0.1		dns_server	c4:6e:1f:92:ed:d8	8.7 KB	0.0 B	3	dns
   	192.168.1.1		dns_server	c4:6e:1f:92:ed:d8	16.6 KB	0.0 B	3	dns
   	10.1.1.1	HISTORIAN-01	historian	d8:9d:b9:00:17:ec	2.1 KB	377.9 KB	7	browser, pi-connect
   	172.16.0.253		master	00:04:23:e0:04:3c	3.0 MB	1.0 MB	26	dns, modbus, pi-connect, smb, vnc
   	192.168.1.12		master	09:00:09:00:01:12	1.7 MB	869.0 KB	50	dns, iec104, pi-connect, smb, vnc
   	192.168.1.11		master	18:66:da:00:01:11	1.6 MB	570.6 KB	41	dns, iec104, pi-connect, smb, vnc
   	172.16.1.253		master	00:04:23:e0:04:3c	3.0 MB	1.0 MB	26	dns, modbus, pi-connect, smb, vnc
   	192.168.162.22		master	09:00:09:00:01:12	304.6 KB	361.8 KB	12	dns, ethernetip, pi-connect, smb
   	172.16.0.101		master	10:c3:7b:4c:83:17	1.7 MB	716.2 KB	4	dce-rpc, dns, opc, smb, vnc
   	00:60:78:00:6a:10		other	00:60:78:00:6a:10	240.0 B	120.0 B	0	-
   	10.4.1.32		other	b4:a3:82:02:66:00	54.5 KB	2.6 KB	1	rtsp
   	ff:ff:ff:ff:ff:ff		other	ff:ff:ff:ff:ff:ff	0.0 B	282.0 B	0	-
   	ec:74:ba:66:66:60	MACH-666666	other	ec:74:ba:66:66:60	0.0 B	0.0 B	0	-
   	00:16:b9:49:b6:7d	ACMEIncHQ_SW1	other	00:16:b9:49:b6:7d	356.0 B	0.0 B	1	lldp
   	00:16:b9:49:b6:40	ACMEIncHQ_SW1	other	00:16:b9:49:b6:40	188.0 B	0.0 B	0	-
   	10.2.1.255		other	ff:ff:ff:ff:ff:ff	0.0 B	12.5 KB	1	browser
   	172.16.66.53		other	18:a9:05:24:d8:b5	788.0 B	3.8 KB	6	smb
   	00:1b:bce:c8:62	ACMEIncHQ_SW2	other	00:1b:bce:c8:62	0.0 B	0.0 B	0	-
   	01:80:c2:00:00:0e		other	01:80:c2:00:00:0e	0.0 B	3.1 KB	6	lldp
   	00:04:23:e0:04:3c		other	00:04:23:e0:04:3c	240.0 B	120.0 B	0	-
   	10.4.1.36		other	b4:a3:82:06:48:aa	54.5 KB	2.6 KB	1	rtsp
   	10.5.1.253		other	c4:2f:90:a8:cf:79	18.4 KB	381.7 KB	7	rtsp
   	00:50:56:f6:3c:50		other	00:50:56:f6:3c:50	42.0 B	0.0 B	0	-
   	00:0c:29:1d:f2:4a		other	00:0c:29:1d:f2:4a	42.0 B	42.0 B	0	-
   	192.168.162.255		other	ff:ff:ff:ff:ff:ff	0.0 B	777.0 B	1	browser

1 2 3 4 5 6 7 8

Figure 9: The Nodes table

This page shows all the nodes in the Environment.

In addition to the node information there is an Actions column which enables the user to gain more information about a node, here is an explanation:

In this page are listed all the Nodes using a [table](#). By clicking on an IP link it is possible to view a popup with some additional details about the node.

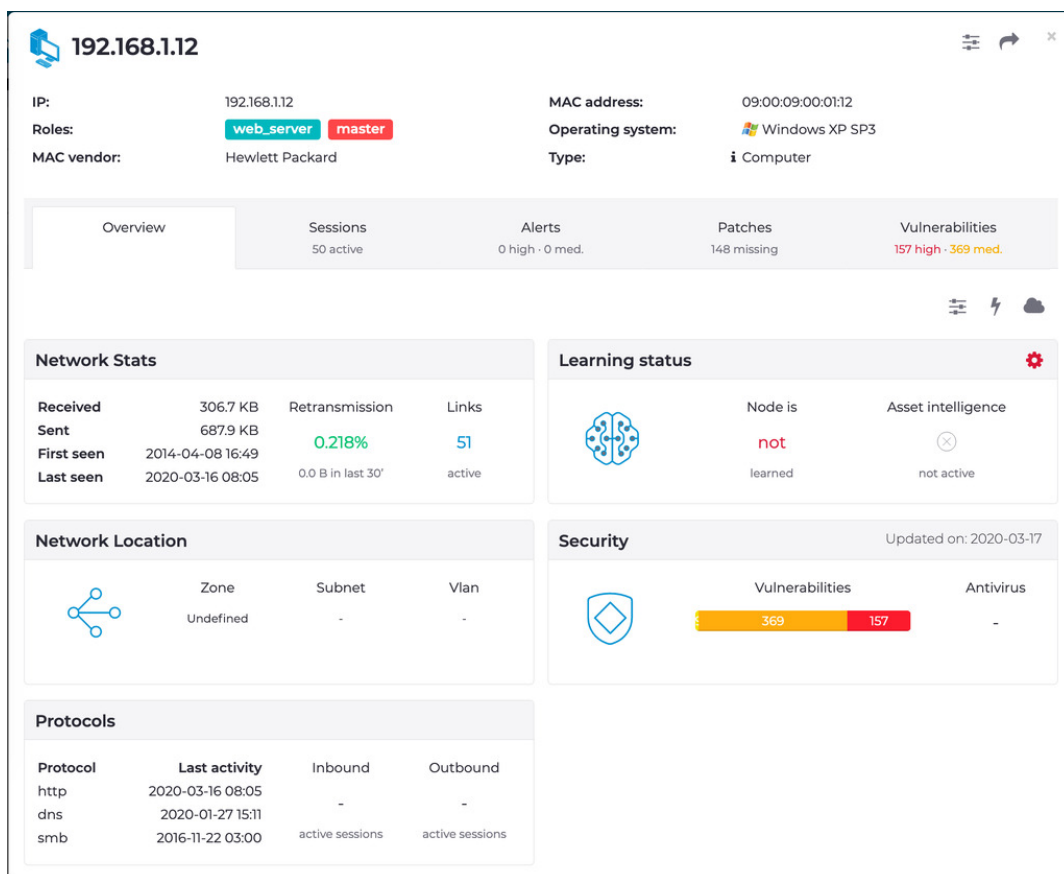


Figure 10: The Node details popup

Network Links

Page 1 of 8,186 entries

Export Live 11 selected

ACTIONS	FROM	TO	PROTOCOL	LAST ACTIVITY	# ALERTS	THROUGHPUT	TRANSFERRED BYTES	TRANSFERRED PACKETS
	00:16:b9:49:b6:7d	01:80:c2:00:00:0e	lldp	2017-02-16 13:57:47.709	0	0.0 b/s	178.0 B	1 pp
	172.16.66.53	192.168.162.53	smb	2016-11-22 03:00:09.105	0	0.0 b/s	394.0 B	1 pp
	192.168.162.22	192.168.1.29	ethernetip	2017-01-05 18:13:55.348	0	0.0 b/s	25.4 KB	240 pp
	192.168.162.22	192.168.1.32	ethernetip	2017-01-05 18:13:55.348	0	0.0 b/s	25.4 KB	240 pp
	192.168.162.22	192.168.1.33	ethernetip	2017-01-05 18:13:55.348	0	0.0 b/s	25.4 KB	240 pp
	192.168.162.22	192.168.1.31	ethernetip	2017-01-05 18:13:55.348	0	0.0 b/s	25.4 KB	240 pp

Figure 11: The Links table

This page shows all the links in the Environment.

In addition to the link information there is an Actions column which enable the user to gain more information about a link, here is an explanation:

Network Sessions

Page 1 of 12,293 entries

Export Live 13 selected

ACTIONS	STATUS	FROM	TO	TRANSPORT PROTOCOL	FROM PORT	TO PORT	PROTOCOL	THROUGHPUT	TRANSFERRED BYTES	TRANSFERRED PACKETS
	ACTIVE	10.51.253	10.4.132	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp
	ACTIVE	10.51.253	10.4.134	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp
	ACTIVE	10.51.253	10.4.130	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp
	ACTIVE	10.51.253	10.4.131	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp
	ACTIVE	10.51.253	10.4.135	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp
	ACTIVE	10.51.253	10.4.136	tcp	51183	554	rtsp	0.0 b/s	79.0 KB	76 pp

Figure 12: The Sessions table

In this page are listed all the [Sessions](#) using a [table](#). By clicking on the From or To node ids additional details about the involved Nodes are displayed. The buttons in the Actions column enable the user to ask or to see the traces and to navigate through the UI. In the other columns there are fine-grained information about each session, like the source and destination ports, the number of transferred packets or bytes, etc.

Network Graph

The network graph page gives a visual overview of the network. In the graph, every vertex represents a network node, while every edge represents one or multiple links between nodes. Edges and vertexes are annotated to give information about the identification of the node, the protocols used in the communications between two nodes, and more. The contents of the graph can be filtered using different criteria in order to obtain a clearer representation, or to evidence specific aspects.

The position of the nodes in the graph is determined by either a specific layout or a dynamic automatic adjustment algorithm that looks for minimal overlap and best readability of the items.

In order to better visualize the desired nodes/links the user can move and zoom the graph using the mouse.

Move	To move the graph click somewhere, not on a node, and start dragging
Zoom (mode 1)	with the mouse inside the window, turn the mouse wheel up and down to zoom in and out (scrolling). The zoom will be centered on the mouse position
Zoom (mode 2)	Drag in vertical direction while keeping pressed the 'z' key. The zoom will be centered on the position where started the mouse dragging

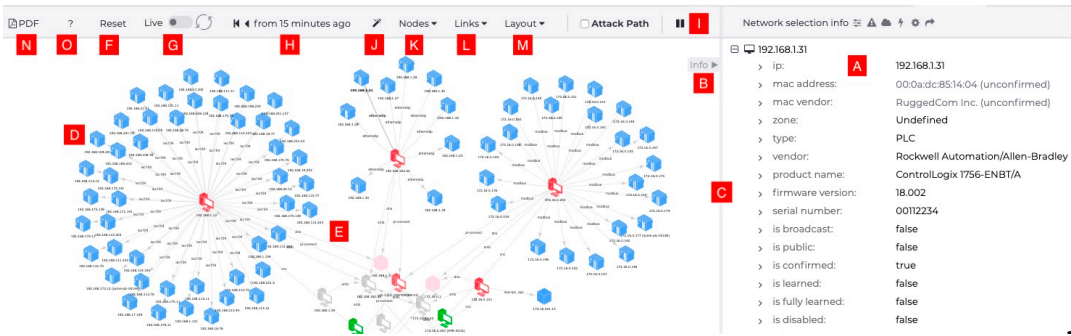


Figure 13: The Environment Network Graph showing info for the selected node

A	The information pane contains the details about the selected item, that is either a node or a link
B	The button to toggle the information pane
C	Drag this vertical line with the mouse to resize the information pane
D	A node
E	A link
F	The button to reset all the customizations and reload the data
G	The button to update the data; it keeps the current customizations
H	The button to filter by activity time
I	The button to toggle the dynamic adjustment motion of the items
J	The magic wand button will open a wizard to help the user to filter the graph and view only the desired information. It contains some solutions to reduce the size of a big graph.
K	The button that configures the appearance of the nodes.

L	The button that configures the appearance of the links.
M	The button that allows to select a graph layout.
N	The button that exports a PDF report containing the graph. Notice that the graph is exported as it is currently shown on the page.
O	The ? button is explained below.

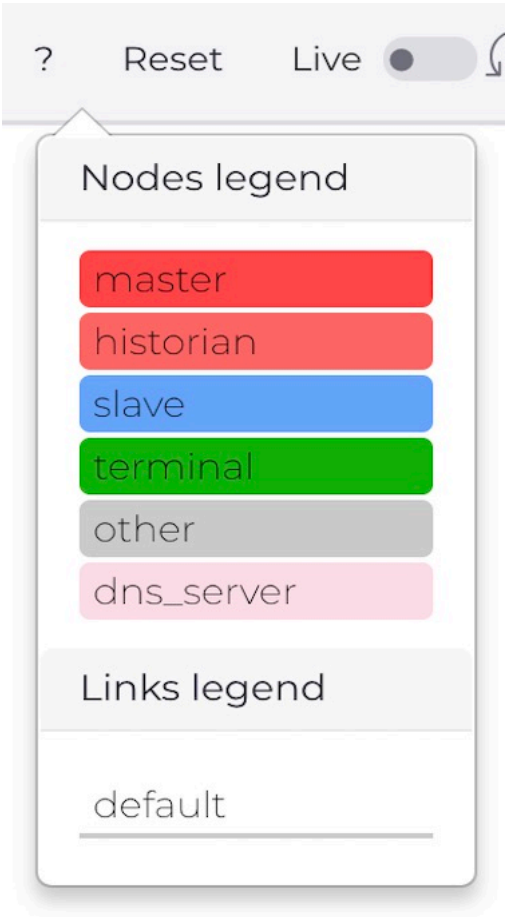


Figure 14: Clicking on the ? button will show the legend for link and nodes. The content of the legend is aware of the selected perspectives

"Magic wand" options

The wizard help the user with several hints to improve the performance of the graph. Settings annotated with an orange exclamation mark are considered suboptimal. Green thumbs annotate options whose settings are considered helpful.

Graph is big, you can filter to go faster

!

Use Google Chrome/Chromium for better performance

!

✓

Show broadcast

Hide broadcast nodes to display a simpler graph

!

Only with confirmed data

Show only links with confirmed data to display a simpler graph

👍

✓

Only confirmed nodes

Show only confirmed nodes to display a simpler graph

!

Exclude tangled nodes

Tangled nodes will be excluded from graph, they can be reincluded by removing their IDs from the nodes options

!

Protocols

SCADA

Choose a protocol ▼

OK

Show broadcast	Broadcast addresses are not actual network nodes in that no asset is bound to a broadcast address. They are used to represent communications performed by a node towards an entire subnet. Removing broadcast nodes reduces the complexity of a graph.
Only with confirmed data	Unconfirmed links can be hidden easily to reduce the complexity of an entangled graph.
Only confirmed nodes	Unconfirmed nodes can be hidden to reduce the size of a large graph.
Exclude tangled nodes	Nodes whose connections cause the node to be too complex can be removed to improve the readability of the graph.
Protocols	Nodes and edges can be filtered so to show only those items participating in communications involving one of the selected protocols. By clicking on "SCADA", all SCADA protocols are selected.

Nodes options

NodesLinks

⏮ from the beginning ✎ Nodes ▾

Perspective:

Roles ▾

Roles:

Choose a role ▾

Exclude IDs:

192.168.1.1,192.168.1.2,...

ID filter: ☐ Exact match

10.197.2,192.168.1.2,...

Display:

ID (label) ▾

Group by

Group by ▾

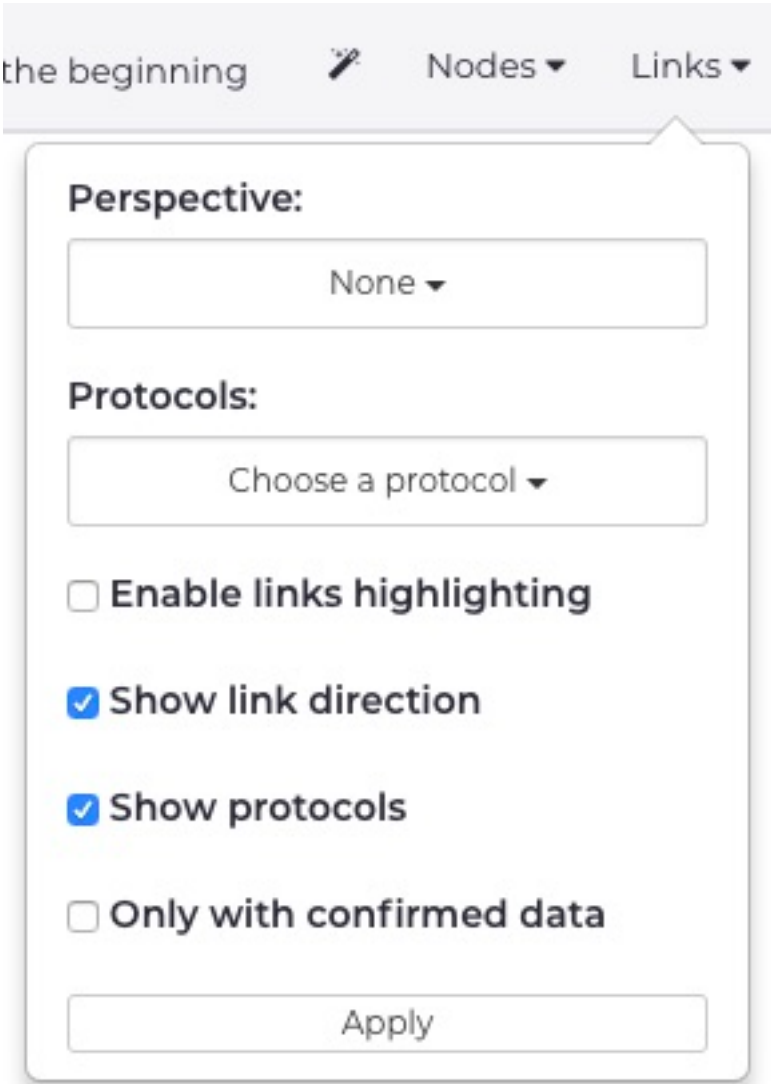
☐ Show broadcast

☒ Only confirmed nodes

Apply

Perspective	Change the color of the nodes according to a predefined criterion
Roles	Allow you to filter the graph by node roles
Exclude IDs	Remove the specified IDs from the graph view; it is possible to specify more IDs separated by comma
ID filter	The graph can be filtered by one or more ID addresses, separated by comma
ID filter exact match	If checked, the ID filter will let the graph show only the nodes with exactly the specified ID(s) and not with a "start with" criterion
Display	Choose the label formatting of the nodes
Group by	Nodes with the chosen property (i.e. zone, subnet, etc) are assigned to the same group, then the way in which the group is displayed depends on the option chosen in the <code>Layout Options</code> . With <code>Standard</code> layout each group is shown collapsed as a single node, while with <code>Grouped</code> layout all the nodes belonging to the same group are placed inside a circle
Show broadcast	If checked, it includes in the graph all the nodes with a broadcast IP
Only confirmed nodes	If checked, it shows only the nodes that exchanged some data in both directions while communicating

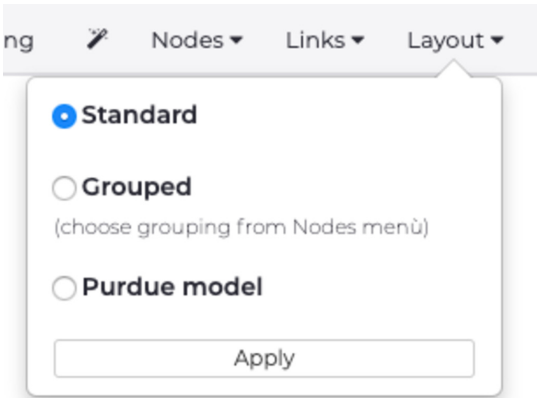
Links options



Perspective	Change the color of the links according to a predefined criterion
Protocols	Allows the ability to filter the graph by link protocols
Enable links highlighting	If checked, links will become bolder in reaction to mouse movements making the link easier to select (may affect performance)
Show protocols	If checked, every link will show its protocols
Only with confirmed data	If checked, it shows only the links which exchanged some data in both directions

Layout options

The layout define the way in which the nodes and links are shown in the graph.



Standard	<p>It is the default layout and the kind of visualization depends on <code>Group_by</code> property:</p> <ul style="list-style-type: none">• <u><code>Group_by</code> not defined</u>: All the nodes and links are shown• <u><code>Group_by</code> defined</u>: All the nodes belonging to the same groups are collpsed into a single node
Grouped	<p>The nodes are grouped according to the criteria defined in <code>Group_by</code>, and the graph is visualized as following</p> <ul style="list-style-type: none">• <u><code>Group_by</code> not defined</u>: All the nodes and links are shown• <u><code>Group_by</code> defined</u>: All the nodes belonging to the same group are shown and are placed inside a circle that represent the group, links between nodes belonging to the same group are shown, while links between nodes of different groups are replaced by links between groups represented as lines that connects the circles
Purdue model	<p>Places the nodes in separate groups according to their level. This allows to distinguish the different levels and isolate potential problems due to communications that cross two or more level boundaries.</p>

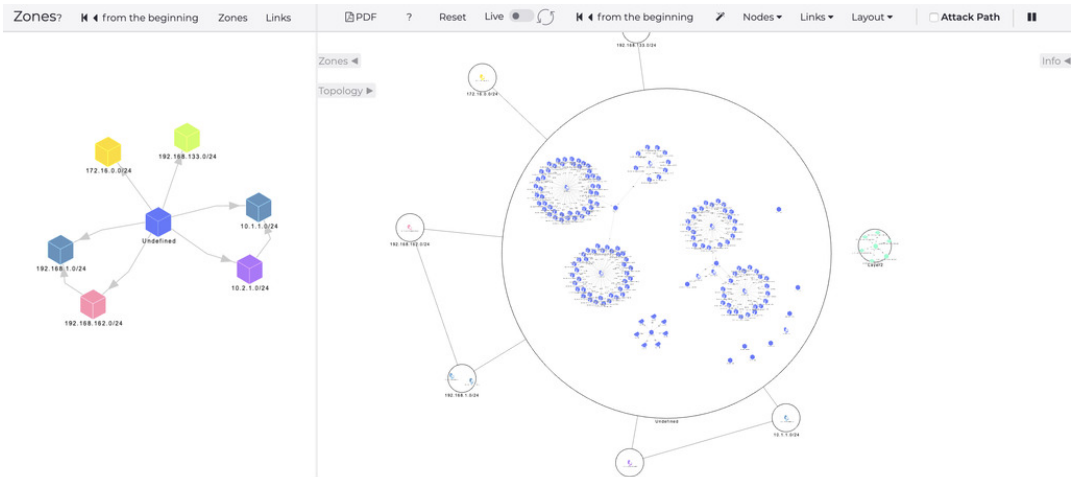


Figure 15: The Environment Graph with the zones pane opened with the `Group_by=Zones`, `Layout = Grouped` and zone perspective.

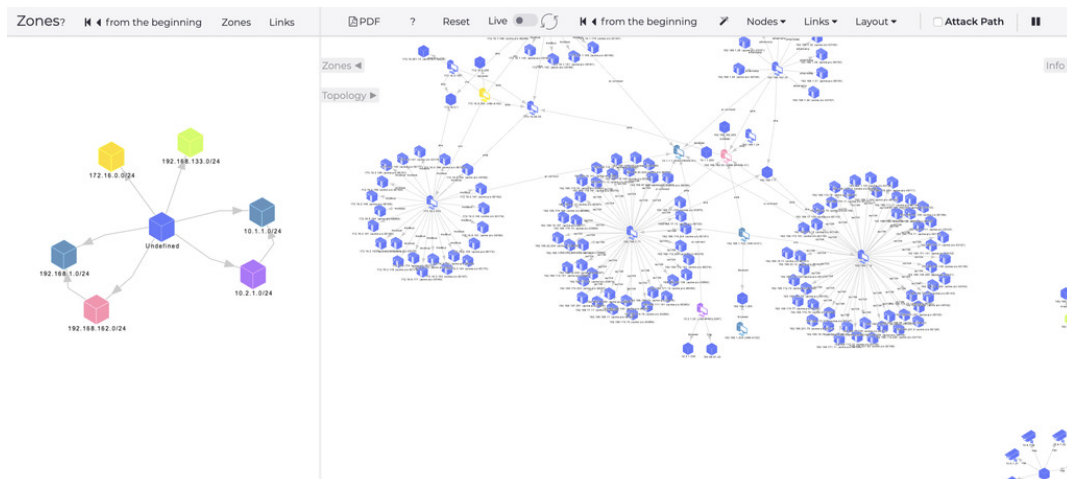


Figure 16: The Environment Graph with the zones pane opened and the zones perspective active to highlight the zone of origin of each node.

The zones pane offers the ability to filter the graph by clicking on a zone or on a link between two zones. The zones graph also has a legend and shares some of the nodes and links options. Clicking on a node or link in the zone pane will show some additional information about the zone or the links between the zones. See [the basic configuration rules](#) to customize Zones.

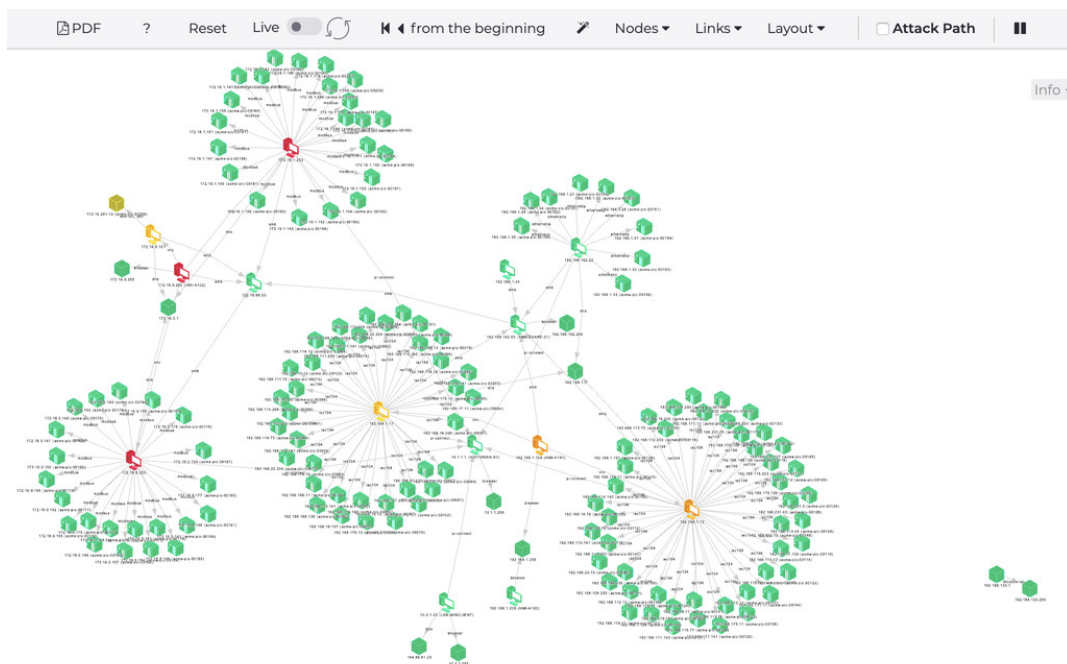


Figure 17: The Environment Graph with the transferred bytes node perspective highlighting the high traffic usage of the master nodes

Traffic

The Traffic tab in the Environment > Network View page shows some useful charts about throughput, protocols and opened TCP connections.

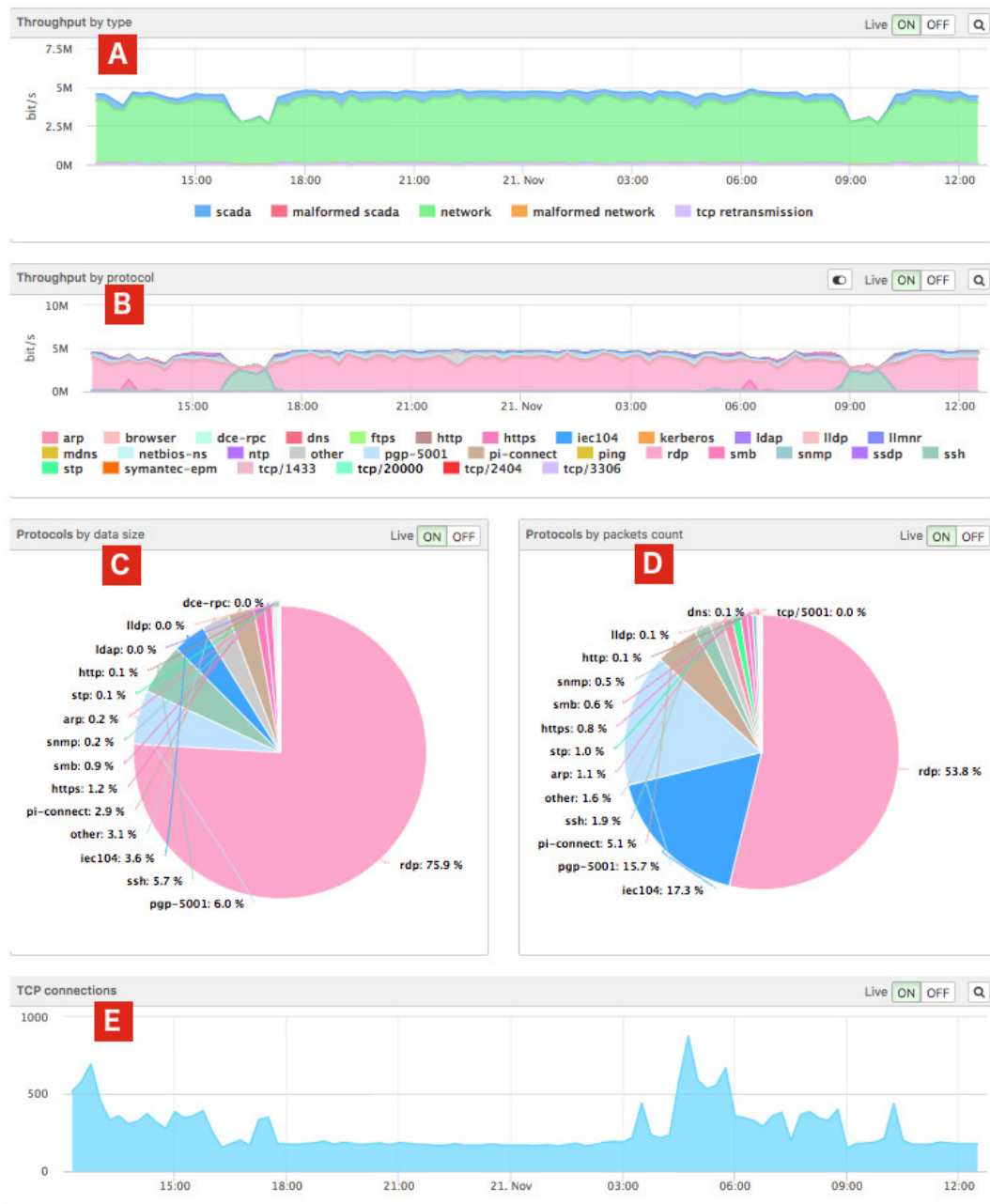


Figure 18: The traffic charts

An explanation of the sections

A	The throughput chart showing traffic divided in macro categories
B	The throughput chart showing traffic for each protocol
C	A pie chart showing the proportions of packets sent by protocol
D	A pie chart showing the proportions of traffic generated by protocol
E	The number of opened TCP connections

Queries

All the data sources of the Nozomi Networks Solution can be queried using N2QL (Nozomi Networks Query Language) from the query page ([Analysis > Queries](#)). In that page, you can also see all the queries that are already saved in the running installation.

You can choose between Standard (currently offered as beta feature) and Expert, the first allows for an easier experience, useful if you want to quickly have a look at your data, the second allows for more complex queries but requires more expertise.

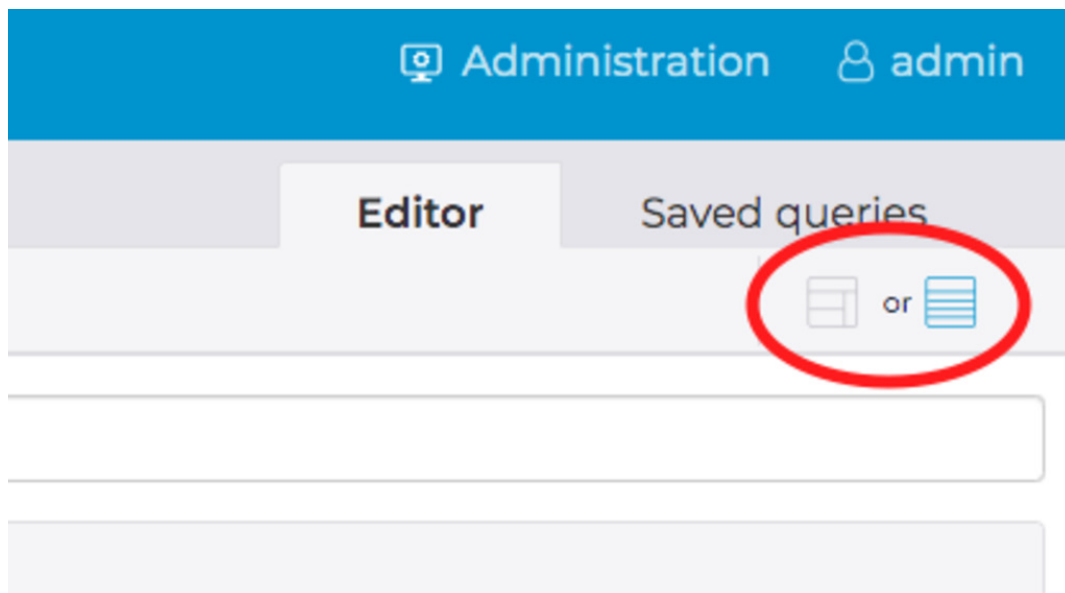


Figure 19: Choose between Standard and Expert

Go to [Queries](#) on page 47 to get a complete reference of query commands and data sources.

Query builder

The Query builder enables the user to easily create and execute queries on the observed system. To do so just click through the different options.

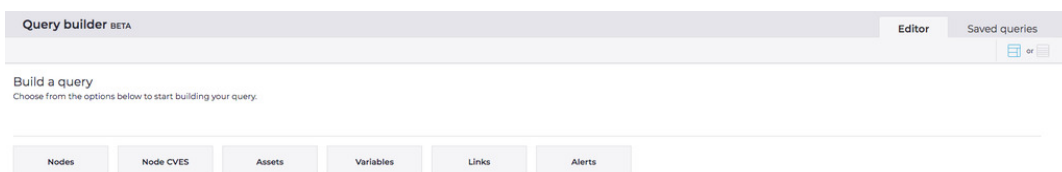


Figure 20: The Query builder

While you build your query the available options change to reflect your choices, guiding you through the process.

Query builder BETA Editor Saved queries

Build a query
Choose from the options below to start building your query:

nodes 42 select appliance_host,created_at,id sort created_at desc

Group by: Group results by column
Head: Returns the first n results
Join: Merges two records into one
Pie chart: Show as a pie chart
Select: Show selected columns
Sort: Sort results by column
Where: Filter results by condition

Result

appliance_host	created_at	id
LAB-sg-upload-PCAPXXX-vm-master	2018-10-25 11:34:28.760	172.16.0.55
LAB-sg-upload-PCAPXXX-vm-master	2018-10-25 11:32:26.285	10.5.1.253
LAB-sg-upload-PCAPXXX-vm-master	2018-10-25 11:32:26.285	10.4.1.31
LAB-sg-upload-PCAPXXX-vm-master	2018-10-25 11:32:26.285	10.4.1.30
LAB-sg-upload-PCAPXXX-vm-master	2018-10-25 11:32:26.285	10.4.1.33

Figure 21: The Query builder during a query

Query Editor

The Query Editor enables the user to execute queries on the observed system. To execute a query just type the query text in the field and press the enter key on the keyboard.

Queries Editor Saved queries

Enter your query

Example queries (click on a query to fill the text box)

- Show a pie chart with the proportion between learned and not learned nodes
nodes | group_by is_learned | pie is_learned count
- Show an histogram with received and sent bytes of the first ten nodes by received bytes
nodes | sort received.bytes desc | head | column ip sent.bytes received.bytes
- Show the first ten most TCP retransmitting iec104 links
links | where protocol == iec104 | sort tcp_retransmission.bytes desc | head
- Show a pie chart with the proportions of the alert types
alerts | group_by type_id | sort count desc | pie type_id count
- Show a pie chart with the average risk by alert type
alerts | group_by type_id avg risk | sort avg desc | pie type_id avg
- Show the top ten requested variables
variables | sort request_count desc | head
- Draw a network graph with only the http links, set the node labels to the mac address vendor, coloring the nodes with a 'zones' perspective and the links with a 'transferred bytes' perspective
nodes | where_link protocol == http | graph node_label:mac_vendor node_perspective:zones link_perspective:transferred_bytes

Figure 22: The Query Editor. Some sample queries are displayed at the beginning, clicking on them will trigger the execution

After the execution, the result will be displayed like in the figure below. If the user has enough privileges (i.e. it belongs to a group with admin privileges), by clicking on the floppy icon on the right, the query will be saved and displayed in the [Saved Queries](#) section, otherwise the button is disabled. To save a query, you must specify a description and a group. The query results can be exported by clicking on the **Export** button, and by choosing between the **Excel** or the **CSV** format. The corresponding file will be produced in background (to facilitate the production of queries with large amount of data) and it can be retrieved through the **Exports List** submenu, once ready. When an export is downloaded it is automatically removed from the filesystem.

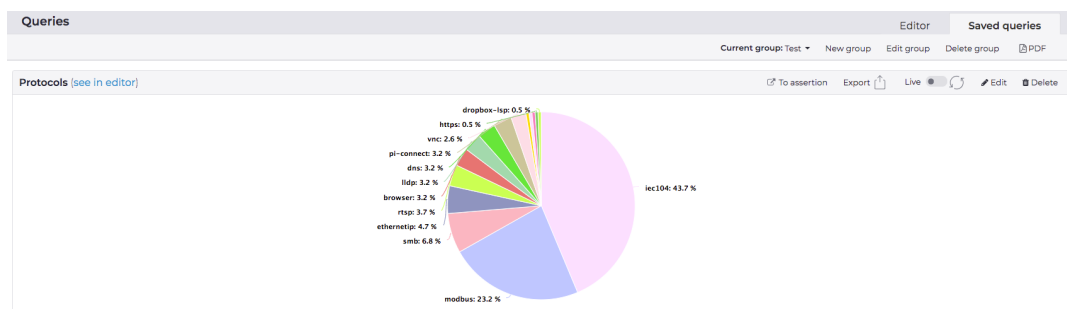


Figure 23: The Query Editor during a query

Saved Queries

When a query is saved, it will be displayed in the Saved Queries section. Here, by using the group selector, it is possible to change the current group and to restrict the view to the queries of the chosen group.

Query groups, a simple but powerful method to organize the queries, can be created, renamed and deleted only by admin users. When a group is deleted, all the queries contained in it will be eliminated.

By clicking on the pen icon, it is possible to change the description and/or the group of a query. By clicking on the trash icon, the saved query will be deleted. As for the saving actions, the user requires admin privileges in order to perform such operations.

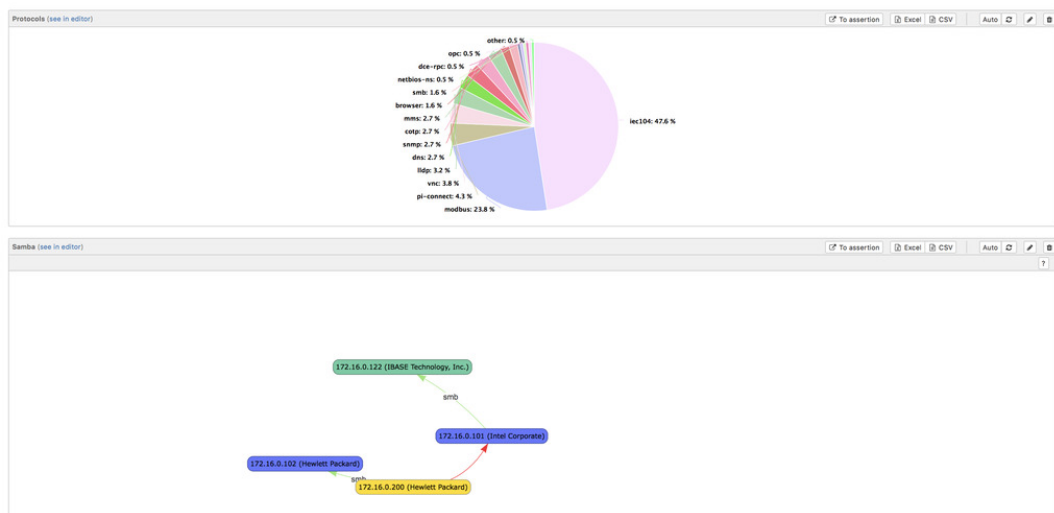
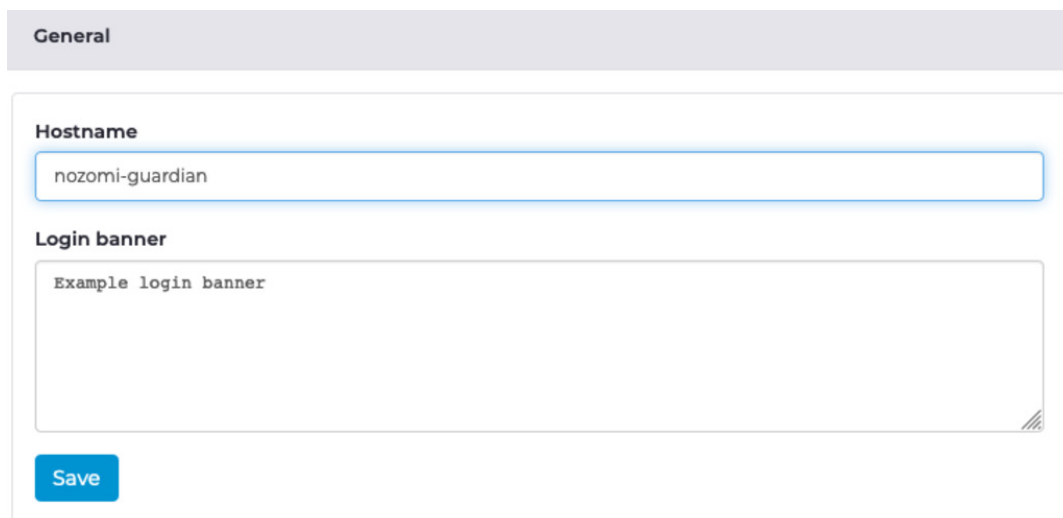


Figure 24: The Saved Queries

System

General

In the **Administration > General** page it is possible to change the hostname of the Appliance and to specify a login banner. The login banner is optional and, when set, it is shown in the login page and at the beginning of all SSH connections.



The screenshot shows the 'General' configuration page. At the top is a tab labeled 'General'. Below it, there are two input fields. The first is labeled 'Hostname' and contains the text 'nozomi-guardian'. The second is labeled 'Login banner' and contains the text 'Example login banner'. Below these fields is a blue 'Save' button.

Figure 25: The hostname and login banner input fields

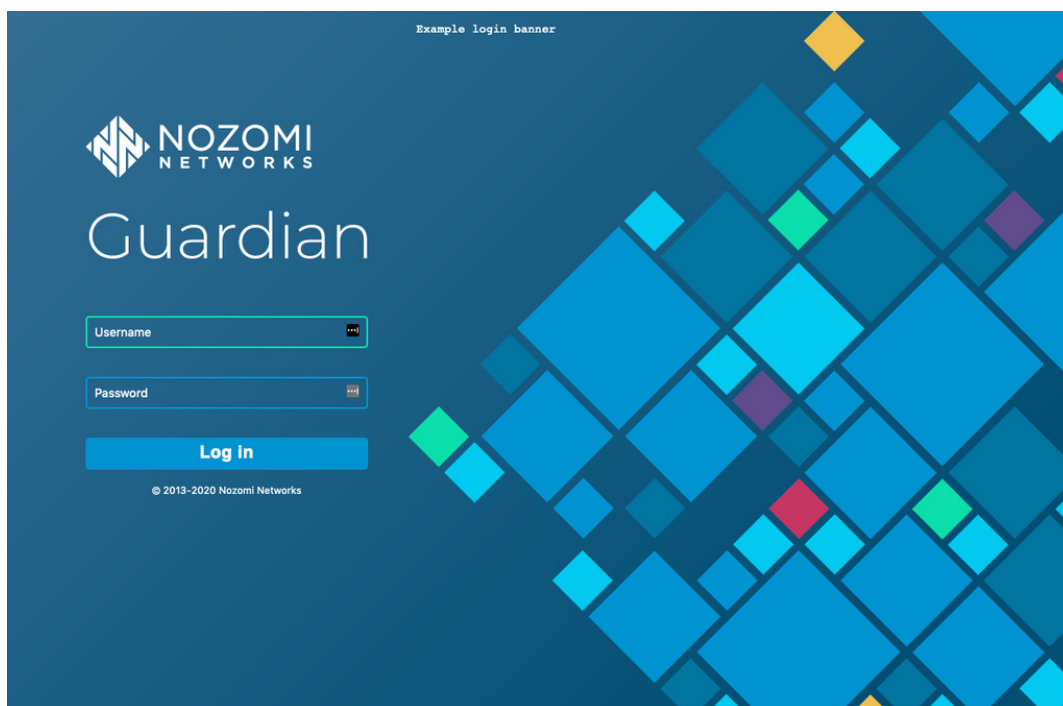


Figure 26: An example of login banner

Date and time

Date settings

Timezone

Local: CET (UTC+02:00) ▼

Save

Date (format ccyymmddHHMM.ss)

Pick a date

Set as client

201907250932.16

Save

NTP

☐ Enabled

Servers

0.freebsd.pool.ntp.org,1.freebsd.pool.ntp.org

Save

Figure 27: Date and time configuration panel

From the date and time page you can:

- change the timezone of the appliance
- change the current time of the appliance (you can use the **Pick a date** or **Set as client** buttons to set a date in a simple way)
- enable or disable the time synchronization to a NTP server by writing a list of comma-separated server addresses

Network interfaces

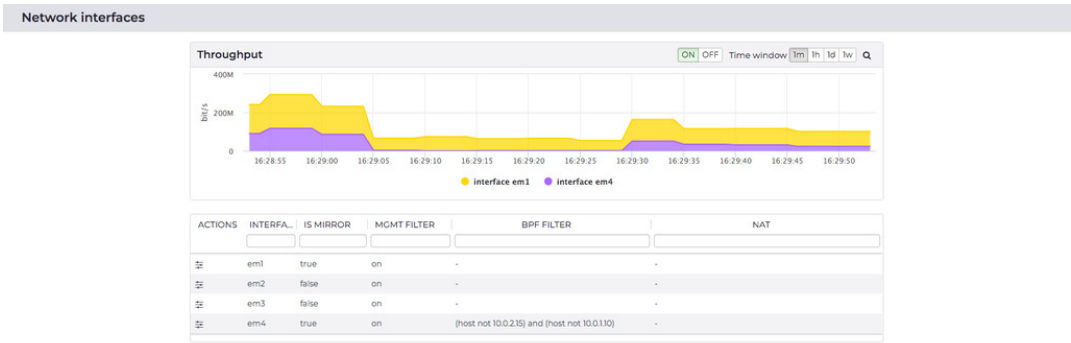


Figure 28: Network interfaces list

Actions	With the configuration button, you are able to define/modify the NAT rule to be applied to the current interface.
Interface	The interface name
Is mirror	It is true if the interface is likely receiving mirror traffic and not only broadcast.
Mgmt filter	When on the traffic of the appliance is filtered out. It is on by default. To change the value see the specific configuration rule in Basic configuration rules .
BPF filter	The BPF filter applied to the sniffed traffic.
NAT	The NAT rule applied to the current interface.

In this form you can set the NAT configuration and the BPF filter.

Configure interface

Interface
em4

NAT

Original subnet
e.g. 192.168.0.0

Translated subnet
e.g. 10.1.0.0

CIDR mask
e.g. /16

▪ This rule allow the rewriting of source and destination IPs of packets sniffed on this interface.
 ▪ e.g. to translate 192.168.1.100 in 10.1.1.100 you have to configure the rule: 192.168.0.0 10.1.0.0 /16

BPF filter

BPF filter editor

Manual insertion of a custom filter expression ☐

(host not 10.0.2.15) and (host not 10.0.1.10)

Save **Cancel**

Figure 29: Interface configuration form

In the NAT part you may configure the original subnet, the destination subnet and the CIDR mask for the NAT rule.

In the BPF filter part you may configure the filter to apply to this interface. There are two ways to configure the filter, via a visual editor or manually. By clicking on the "BPF Filter editor" the following visual editor appears. It is possible to edit the most common filters.

BPF filter editor

This editor supports the most common BPF syntax. If you wish to enter an advanced filter please use the "Manual insertion" from the previous window.

NOT AND OR **Add rule** **Add group**

NOT host 10.0.2.15 **Add rule** **Add group** **Delete**

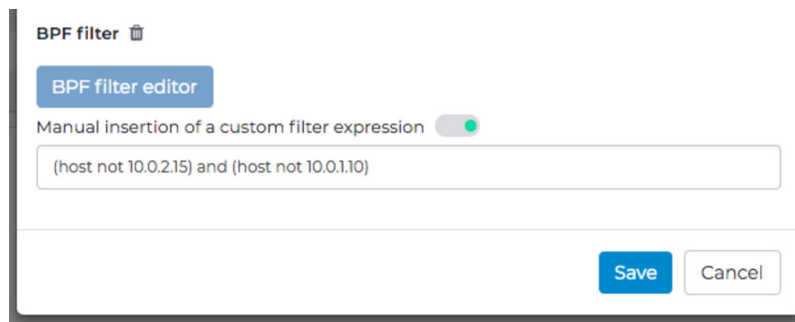
NOT host 10.0.1.10 **Add rule** **Add group** **Delete**

Current BPF Filter
(not (host 10.0.2.15)) and (not (host 10.0.1.10))


Ok **Cancel**

Figure 30: BPF filter editor

More complex filters may be inserted manually in the input box by clicking on the toggle.



The image shows a user interface for editing a BPF filter. At the top, it says "BPF filter" with a trash icon. Below that is a blue button labeled "BPF filter editor". Underneath the button is the text "Manual insertion of a custom filter expression" followed by a toggle switch that is currently turned on (green). Below the toggle is a text input field containing the expression "(host not 10.0.2.15) and (host not 10.0.1.10)". At the bottom right of the interface are two buttons: a blue "Save" button and a white "Cancel" button with a grey border.

BPF filter 

BPF filter editor

Manual insertion of a custom filter expression ☒

(host not 10.0.2.15) and (host not 10.0.1.10)

Save Cancel

Figure 31: Manual insertion of a BPF filter

Upload PCAPs

In the **Administration > Upload PCAPs** page you can play a PCAP file into Guardian, the appliance ingests the traffic as if it came through the network.

Upload PCAPs

Options:

☒ Use PCAP timestamps

☐ Delete data before play

☐ Auto play PCAP after upload

Drop a PCAP here or click to upload

Supported formats: PCAP and PCAPNG - Maximum file size: 2G

Last uploaded PCAPs

Page 1 of 1, 0 entries

Live ☐

6 selected

ACTIONS	LAST UPLOADED TI...	LAST PLAYED TIME	FILENAME	NOTE	USERNAME
No PCAP uploaded yet					

On top, there are flags that you can use to customize the behaviour of the upload/play action.

Use PCAP timestamps	Check this if you want to use the time captured in the PCAP file. Otherwise, the current time is used.
Delete data before play	Check this option if you want to delete all data in the Appliance before running the play action.
Auto play PCAP after upload	With this flag enabled, the PCAP is played immediately after the upload.

On every single PCAP file uploaded there are some available actions as shown below.

Last uploaded PCAPs

Page 1 of 1, 1 entries / filtered by filename: base x

Live ☐

6 selected

ACTIONS	LAST UPLOADED TI...	LAST PLAYED TI...	FILENAME	NOTE	USERNAME
	20:44:26.716	20:44:39.134	base.pcap	-	admin

Replay PCAP	With this action you can replay the PCAP.
Edit note	If you need to share some note about the uploaded PCAP.
Delete from the list	Erase the PCAP file from the Appliance, no Environment data will be affected.

Note: By default, the Appliance has a retention of 10 PCAP files. To configure this value see [Configuring retention](#)

Health

All the sections described below are available for admin user. Additionally, access is granted to all users with Health permission.

Performance

In this tab there are three charts showing, respectively, the CPU, RAM and disk usage over time.

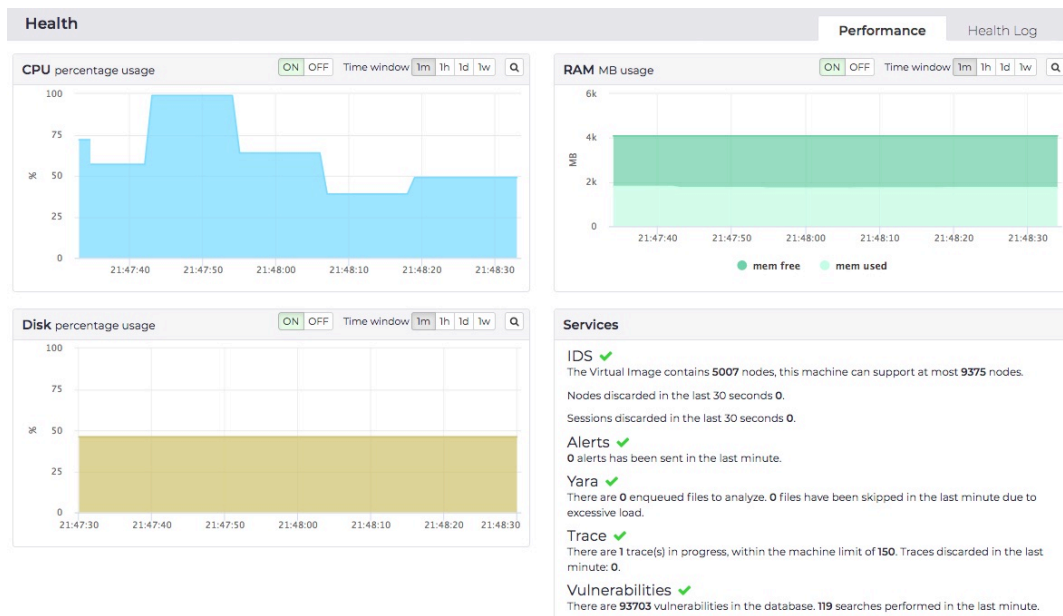


Figure 32: The performance charts

Health Log

If there are any kind of performance issues on the appliance, here you can see the history of those problems with a small description as the number of packet, the session or nodes discarded in the last 30 seconds.

Health		Performance	Health Log
Page 1 of 30,729 entries		Export	Live
TIME	DESCRIPTION		
2018-12-17 12:23:58.343	100% cpu usage		
2018-10-21 22:57:01.761	100% cpu usage		
2018-09-26 03:41:05.379	99% disk space used		
2018-09-26 03:31:01.216	99% disk space used 99% cpu usage		
2018-09-25 22:28:55.947	98% disk space used		
2018-09-25 22:13:49.672	98% disk space used 100% cpu usage		
2018-09-25 17:46:58.364	97% disk space used		

Figure 33: The health log table

Audit

In the Administration > Audit page are listed all relevant actions made by users, starting from Login/Logout action to all the configuration operations, such as learn/delete of objects in Environment.

All the recorded actions are related to the IP and the username of the user who made the action and, as seen in the other Nozomi Networks Solution tables, you can easily filter and sort on this data.

Audit				
Page 1 of 1, 8 entries / filtered by time: ["to":1545162024913,"from":1545123087186] x				
Live 5 selected				
ACTIONS	TIME	IP	USERNAME	EVENT
<div> <div> <div></div> <div></div> <div></div> </div> <div> <div></div> <div></div> <div></div> </div> </div>				
Q	20:40:11.935	10.41.128.12	admin	Configured rules, rules: ["vi zones delete LocalSCADA"]
Q	20:40:09.884	10.41.128.12	admin	Configured rules, rules: ["vi zones delete VPN_RSA"]
Q	20:38:17.832	10.41.128.12	admin	Configured rules, rules: ["vi zones add 10.197.163.0/24 LocalSCADA", "vi zones setlevel 2 LocalSCADA"]
Q	20:37:39.528	10.41.128.12	admin	Configured rules, rules: ["vi zones add 10.195.0.0/16 VPN_RSA", "vi zones setlevel VPN_RSA"]
Q	20:14:06.957	10.41.128.12	admin	User signed in
Q	17:01:07.736	10.41.132.167	admin	User signed in
Q	14:22:31.552	10.41.132.165	admin	User signed in
Q	12:13:31.254	10.41.132.165	admin	User signed in

Figure 34: The audit table

Reset Data

In the **Administration > Data** page it is possible to selectively reset several kinds of data used by the Nozomi Networks Solution.

Environment	All learned nodes, links and variables
Network Data	All the history of network data visible in the charts
Process Data	All the history of process data visible in the charts like the variables history
Assets Data	All the information related to asset (e.g. software and hardware version)
Alerts	The raised alerts
Traces	The traces both generated by the alert and by a user request
Time machine	The snapshots saved by the time machine
Queries	The queries and query groups saved by each user
Assertions	The assertions saved by each user

In addition to the usual buttons for selecting and deselecting all the checkboxes, **All** and **None**, there is also a **Only data** button that selects everything but traces, queries and assertions.

Data

Reset Restore

Reset different kind of data for all the users

All Only data None

☐ Environment
Reset network nodes, links status and variables

☐ Network Data
Reset link event history, network charts data and captured urls

☐ Process Data
Reset the variables history

☐ Assets Data
Reset the data related to assets

☐ Alerts
Reset the alerts

☐ Traces
Reset the traces, both requested by users and generated by alerts

☐ Time machine
Delete all the snapshots of the time machine

☐ Queries
Delete all the queries and query groups

☐ Assertions
Delete all the assertions

Reset

Figure 35: The reset data form

Queries

All the data sources of the Nozomi Networks Solution can be queried using N2QL (Nozomi Networks Query Language) from the query page ([Analysis > Queries](#)). In that page, you can also see all the queries that are already saved in the running installation.

You can choose between Standard (currently offered as beta feature) and Expert, the first allows for an easier experience, useful if you want to quickly have a look at your data, the second allows for more complex queries but requires more expertise.

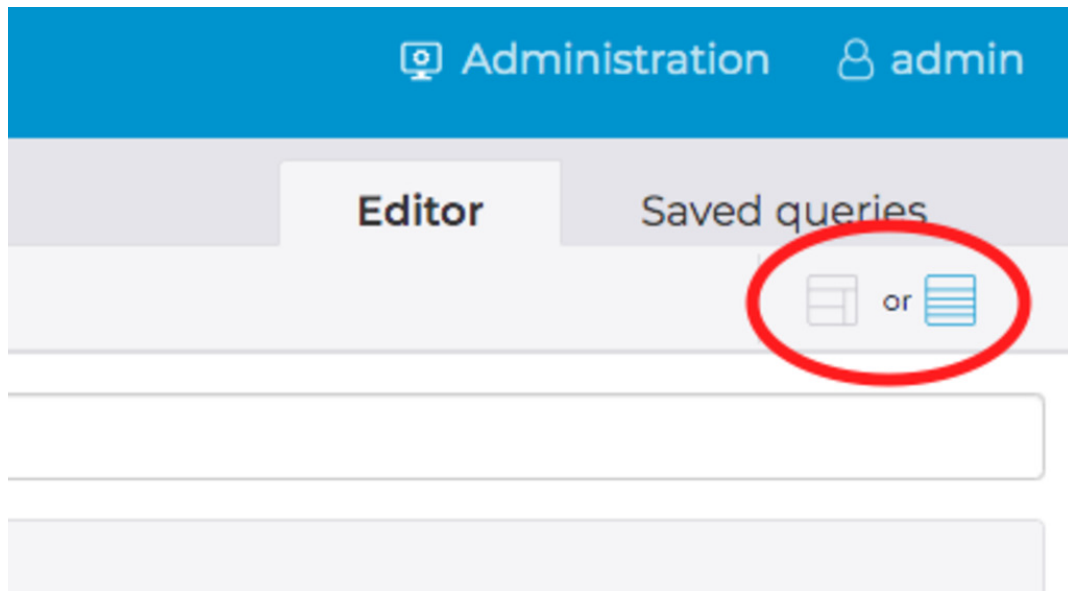


Figure 36: Choose between Standard and Expert

Go to [Queries](#) on page 47 to get a complete reference of query commands and data sources.

Query builder

The Query builder enables the user to easily create and execute queries on the observed system. To do so just click through the different options.

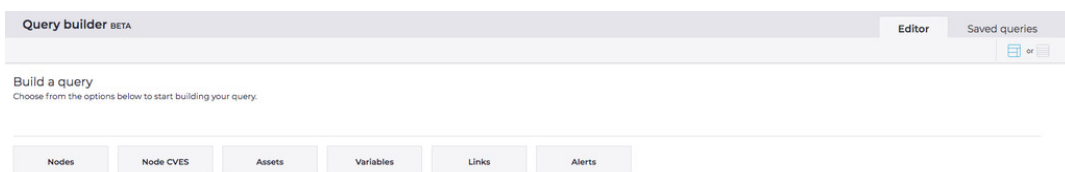


Figure 37: The Query builder

While you build your query the available options change to reflect your choices, guiding you through the process.

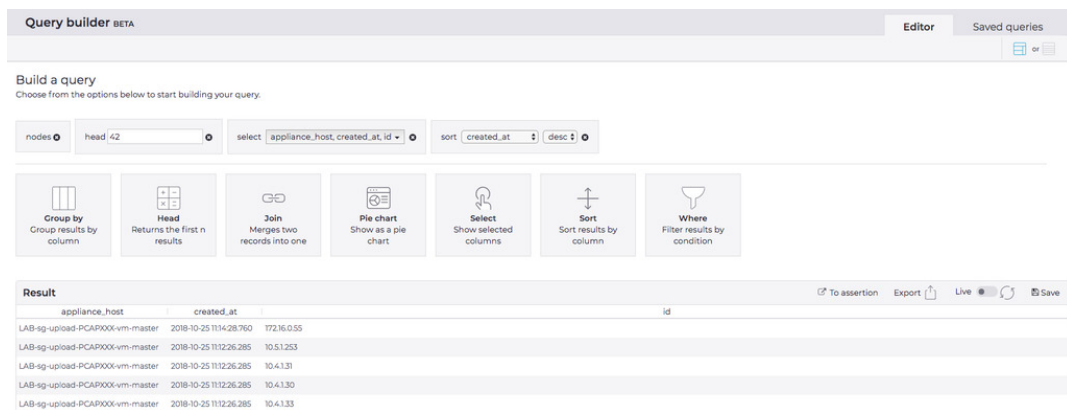


Figure 38: The Query builder during a query

Query Editor

The Query Editor enables the user to execute queries on the observed system. To execute a query just type the query text in the field and press the enter key on the keyboard.

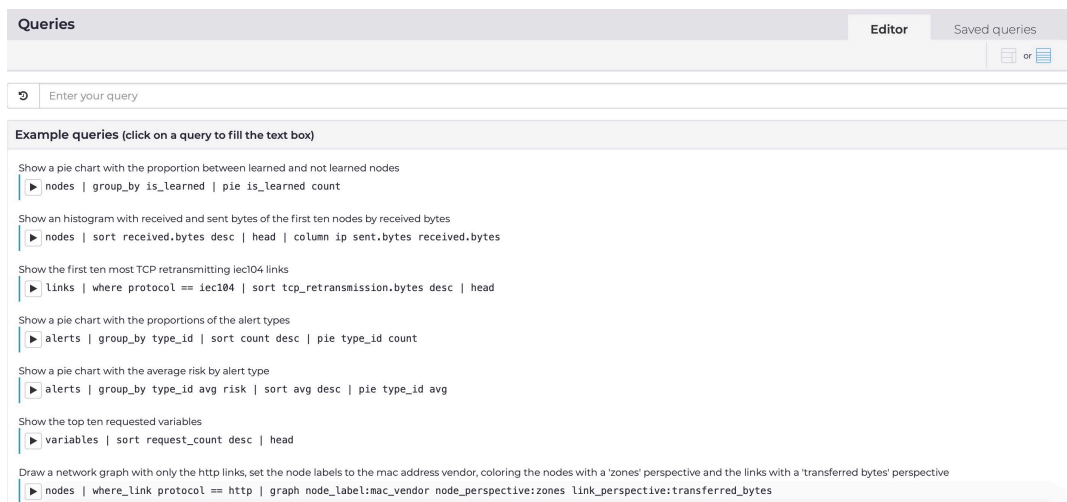


Figure 39: The Query Editor. Some sample queries are displayed at the beginning, clicking on them will trigger the execution

After the execution, the result will be displayed like in the figure below. If the user has enough privileges (i.e. it belongs to a group with admin privileges), by clicking on the floppy icon on the right, the query will be saved and displayed in the [Saved Queries](#) section, otherwise the button is disabled. To save a query, you must specify a description and a group. The query results can be exported by clicking on the **Export** button, and by choosing between the **Excel** or the **CSV** format. The corresponding file will be produced in background (to facilitate the production of queries with large amount of data) and it can be retrieved through the **Exports List** submenu, once ready. When an export is downloaded it is automatically removed from the filesystem.

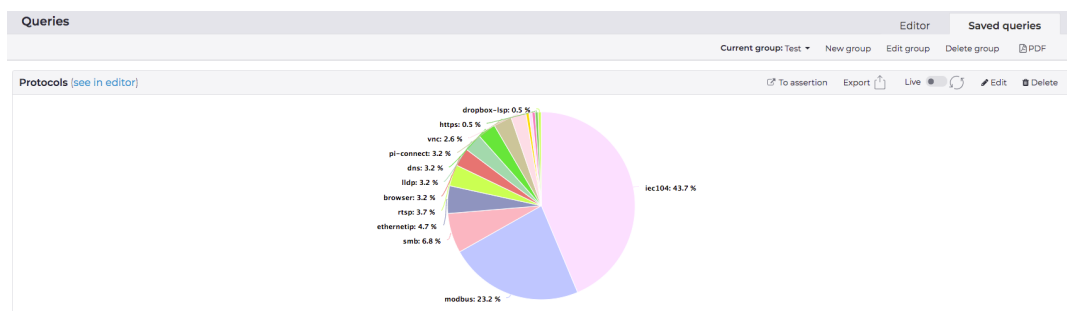


Figure 40: The Query Editor during a query

Saved Queries

When a query is saved, it will be displayed in the Saved Queries section. Here, by using the group selector, it is possible to change the current group and to restrict the view to the queries of the chosen group.

Query groups, a simple but powerful method to organize the queries, can be created, renamed and deleted only by admin users. When a group is deleted, all the queries contained in it will be eliminated.

By clicking on the pen icon, it is possible to change the description and/or the group of a query. By clicking on the trash icon, the saved query will be deleted. As for the saving actions, the user requires admin privileges in order to perform such operations.

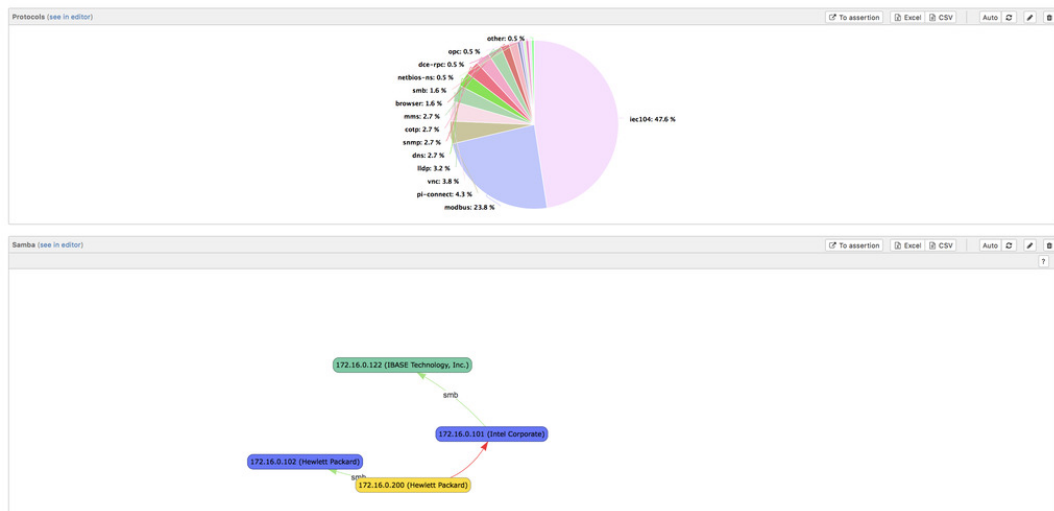


Figure 41: The Saved Queries

Chapter 4

Queries

Topics:

- [Overview](#)
- [Reference](#)
- [Examples](#)

In this chapter are listed all the [data sources](#), [commands](#) and [functions](#) which can be used in N2QL (Nozomi Networks Query Language).

Overview

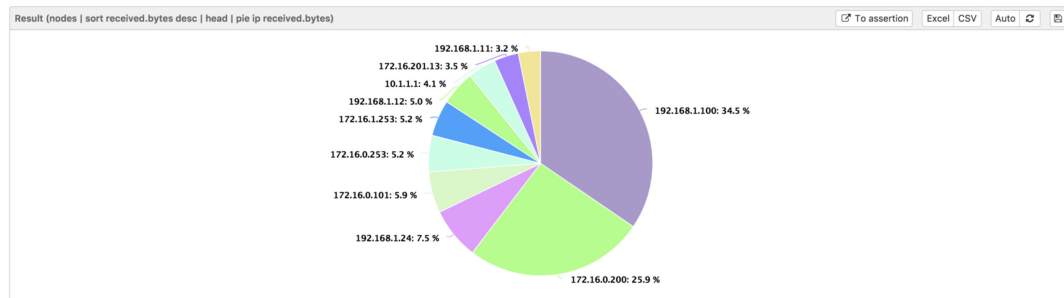
Each query must start by calling a [data source](#), for example:

```
nodes | sort received.bytes desc | head
```

will show in a table the first 10 nodes which received the most bytes.

By adding the `pie` [command](#) at the end it is possible to display the result as a pie chart where each slice has the node ip as label and the `received.bytes` field as data:

```
nodes | sort received.bytes desc | head | pie ip received.bytes
```



Sometimes query commands are not enough to achieve the desired result. As a consequence, the query syntax supports [functions](#). Functions allow you to apply calculations on the fields and to use the result as a new temporary field.

For example, the query:

```
nodes | sort sum(sent.bytes,received.bytes) desc | column ip  
sum(sent.bytes,received.bytes)
```

uses the `sum` function to `sort` on the aggregated parameters and to produce a chart with the columns representing the sum of the sent and received bytes.

Reference

Data sources

These are the available data sources with which you can start a query:

help	Show this list of data sources
alerts	All the alerts raised
assertions	All the assertions saved by the users
assets	All the assets identified in the system
captured_urls	All the urls captured from network protocols
function_codes	All the function codes
links	The links in the system, each link has a one-to-one association with a protocol
link_events	The link events saved for each link, for instance channel up/down events, protocol-specific parameters, etc.
nodes	The nodes in the system
node_cpes	All the CPEs (hardware, operating system and software versions) detected on nodes
node_cpe_changes	CPEs (hardware, operating system and software versions) changes collected over time
node_cves	All vulnerabilities detected on node's CPEs
sessions	All currently live network sessions
variables	The SCADA variables of the slaves
variable_history	The history of the variable values
variable_history_month	The history of the variable values within the month specified
zones	The zone nodes
zone_links	The zone links

Commands

Here is the complete list of commands:

Syntax	<code>select <field1> <field2> ... <fieldN></code>
Parameters	<ul style="list-style-type: none"> the list of field(s) to output
Description	The select command takes all the input items and outputs them with only the selected fields

Syntax	<code>exclude <field1> <field2> ... <fieldN></code>
Parameters	<ul style="list-style-type: none"> the list of field(s) to remove from the output
Description	The exclude command takes all the input items and outputs them without the specified field(s)

Syntax	<code>where <field> <== != < > <= >= include? start_with? end_with? in_subnet?> <value></code>
Parameters	<ul style="list-style-type: none"> field: the name of the field to which the operator will be applied operator value: the value used for the comparison. It can be a number, a string or a list (using JSON syntax), the query engine will understand the semantics
Description	The where command will send to the output only the items which fulfill the specified criterion, many clauses can be concatenated using the boolean OR operator
Example	<ul style="list-style-type: none"> <code>nodes where roles include? master OR zone == office</code> <code>nodes where ip in_subnet? 192.168.1.0/24</code>

Syntax	<code>sort <field> [asc desc]</code>
Parameters	<ul style="list-style-type: none"> field: the field used for sorting asc desc: the sorting direction
Description	The sort command will sort all the items according to the field and the direction specified, it automatically understands if the field is a number or a string

Syntax	<code>group_by <field> [[avg sum] [field2]]</code>
Parameters	<ul style="list-style-type: none"> field: the field used for grouping avg sum: if specified, the relative operation will be applied on field2
Description	The group_by command will output a grouping of the items using the field value. By default the output will be the count of the occurrences of distinct values. If an operator and a field2 are specified, the output will be the average or the sum of the field2 values

Syntax	<code>head [count]</code>
Parameters	<ul style="list-style-type: none"> count: the number of items to output
Description	The head command will take the first count items, if count is not specified the default is 10

Syntax	<code>uniq</code>
Parameters	
Description	The <code>uniq</code> command will remove from the output the duplicated items

Syntax	<code>expand <field></code>
Parameters	<ul style="list-style-type: none"> <code>field</code>: the field containing the list of values to be expanded
Description	The <code>expand</code> command will take the list of values contained in <code>field</code> and for each of them it will duplicate the original item substituting the original <code>field</code> value with the current value of the iteration

Syntax	<code>sub <field></code>
Parameters	<ul style="list-style-type: none"> <code>field</code>: the field containing the list of objects
Description	The <code>sub</code> command will output the items contained in <code>field</code>

Syntax	<code>count</code>
Parameters	
Description	The <code>count</code> command outputs the number of items

Syntax	<code>pie <label_field> <value_field></code>
Parameters	<ul style="list-style-type: none"> <code>label_field</code>: the field used for each slice label <code>value_field</code>: the field used for the value of the slice, must be a numeric field
Description	The <code>pie</code> command will output a pie chart according to the specified parameters

Syntax	<code>column <label_field> <value_field ...></code>
Parameters	<ul style="list-style-type: none"> <code>label_field</code>: the field used for each column label <code>value_field</code>: one or more field used for the values of the columns
Description	The <code>column</code> command will output an histogram, for each label a group of columns is displayed with the value from the specified <code>value_field(s)</code>

Syntax	<code>history <count_field> <time_field></code>
Parameters	<ul style="list-style-type: none"> <code>count_field</code>: the field used to draw the Y value <code>time_field</code>: the field used to draw the X points of the time series
Description	The <code>history</code> command will draw a chart representing an historic series of values

Syntax	<code>distance <id_field> <distance_field></code>
Parameters	<ul style="list-style-type: none"> <code>id_field</code>: the field used to identify the data <code>distance_field</code>: the field on which the distances are calculated
Description	The <code>distance</code> command will calculate a series of distances from the original series. Each distance value is calculated as the difference between a value and its subsequent occurrence

Syntax	<code>bucket <field> <range></code>
Parameters	<ul style="list-style-type: none"> • <code>field</code>: the field on which the buckets are calculated • <code>range</code>: the range of tolerance in which values are grouped
Description	The bucket command will group data in different buckets, different records will be put in the same bucket when the values fall in the same multiple of <code><range></code>

Syntax	<code>join <other_source> <field> <other_source_field></code>
Parameters	<ul style="list-style-type: none"> • <code>other_source</code>: the name of the other data source • <code>field</code>: the field of the original source used to match the object to join • <code>other_source_field</code>: the field of the other data source used to match the object to join
Description	The join command will take two records and will join them in one record when <code><field></code> and <code><other_source_field></code> have the same value

Syntax	<code>gauge <field> [min] [max]</code>
Parameters	<ul style="list-style-type: none"> • <code>field</code>: the value to draw • <code>min</code>: the minimum value to put on the gauge scale • <code>max</code>: the maximum value to put on the gauge scale
Description	The gauge command will take a value and represent it in a graphical way

Syntax	<code>value <field></code>
Parameters	<ul style="list-style-type: none"> • <code>field</code>: the value to draw
Description	The value command will take a value and represent it in a textual way

Syntax	<code>reduce <field> [sum avg]</code>
Parameters	<ul style="list-style-type: none"> • <code>field</code>: the field on which the reduction will be performed • <code>sum</code> or <code>avg</code>: the reduce operation to perform, it is <code>sum</code> if not specified
Description	The reduce command will take a series of values and calculate a single value

Nodes-specific commands reference

Syntax	<code>where_node <field> < == != < > <= >= include? exclude? start_with? end_with? > <value></code>
Parameters	<ul style="list-style-type: none"> • <code>field</code>: the name of the field to which the operator will be applied • <code>operator</code> • <code>value</code>: the value used for the comparison. It can be a number, a string or a list (using JSON syntax), the query engine will understand the semantics
Description	The <code>where_node</code> command will send to the output only the items which fulfill the specified criterion, many clauses can be concatenated using the boolean OR operator. Compared to the generic <code>where</code> command, the adjacent nodes are also included in the output.

Syntax	<code>where_link <field> < == != < > <= >= include? exclude? start_with? end_with? > <value></code>
Parameters	<ul style="list-style-type: none"> • <code>field</code>: the name of the field to which the operator will be applied • <code>operator</code> • <code>value</code>: the value used for the comparison. It can be a number, a string or a list (using JSON syntax) the query engine will understand the semantics
Description	The <code>where_link</code> command will send to the output only the nodes which are connected by a link fulfilling the specified criterion. Many clauses can be concatenated using the boolean <code>OR</code> operator.

Syntax	<code>graph [node_label:<node_field>] [node_perspective:<perspective_name>] [link_perspective:<perspective_name>]</code>
Parameters	<ul style="list-style-type: none"> • <code>node_label</code>: add a label to the node, the label will be the content of the specified node field • <code>node_perspective</code>: apply the specified node perspective to the resulting graph. Valid node perspective values are: <ul style="list-style-type: none"> • <code>roles</code> • <code>zones</code> • <code>transferred_bytes</code> • <code>not_learned</code> • <code>public_nodes</code> • <code>reputation</code> • <code>appliance_host</code> • <code>link_perspective</code>: apply the specified link perspective to the resulting graph. Valid link perspectives are: <ul style="list-style-type: none"> • <code>transferred_bytes</code> • <code>tcp_firewalled</code> • <code>tcp_handshaked_connections</code> • <code>tcp_connection_attempts</code> • <code>tcp_retransmitted_bytes</code> • <code>throughput</code> • <code>interzones</code> • <code>not_learned</code>
Description	The <code>graph</code> command renders a network graph by taking some nodes as input.

Link Events-specific commands reference

Syntax	<code>availability</code>
Parameters	
Description	The <code>availability</code> command computes the percentage of time a link is UP. The computation is based on the link events UP and DOWN that are seen for the link.

Syntax	<code>availability_history <range></code>
Parameters	<ul style="list-style-type: none"> • <code>range</code>: the temporal window in milliseconds to use to group the link events

Description	The <code>availability_history</code> command computes the percentage of time a link is UP by grouping the link events into many buckets. Each bucket will include the events of the temporal window specified by the <code>range</code> parameter.
--------------------	---

Syntax	<code>availability_history_month <months_back> <range></code>
---------------	---

Parameters	<ul style="list-style-type: none">• <code>months_back</code>: number of months to go back in regards to the current month to group the link events• <code>range</code>: the temporal window in seconds to use to group the link events
-------------------	---

Description	The <code>availability_history</code> command computes the percentage of time a link is UP by grouping the link events into many buckets. Each bucket will include the events of the temporal window specified by the <code>range</code> and <code>months</code> parameters.
--------------------	--

Functions

Here is the complete list of functions:

Syntax	<code>sum(<field>, ...)</code>
Parameters	<ul style="list-style-type: none"> a list of fields to sum
Description	The sum function returns the sum of the fields passed as arguments
Warning	Only available for nodes, links, variables and function_codes

Syntax	<code>color(<field>)</code>
Parameters	<ul style="list-style-type: none"> field: the field on which to calculate the color
Description	The color function generates a color in the rgb hex format from a value
Warning	Only available for nodes, links, variables and function_codes

Syntax	<code>date(<time>)</code>
Parameters	<ul style="list-style-type: none"> time defined as unix epoch
Description	The date function returns a date from a raw time

Syntax	<code>dist(<field1>, <field2>)</code>
Parameters	<ul style="list-style-type: none"> the two fields to subtract
Description	The dist function returns the distance between field1 and field2

Syntax	<code>abs(<field>)</code>
Parameters	<ul style="list-style-type: none"> the field on which to calculate the absolute value
Description	The abs function returns the absolute value of the field

Syntax	<code>div(<field1>, <field2>)</code>
Parameters	<ul style="list-style-type: none"> field1 and field2: the two field to divide
Description	The div function will calculate the division field1/field2

Syntax	<code>coalesce(<field1>, <field2>, ...)</code>
Parameters	<ul style="list-style-type: none"> a list of fields or string literals in the format "<chars>"
Description	The coalesce function will output the first value that is not null

Syntax	<code>concat(<field1>, <field2>, ...)</code>
Parameters	<ul style="list-style-type: none"> a list of fields or string literals in the format "<chars>"
Description	The concat function will output the concatenation of the input fields or values

Syntax	<code>round(<field>, [precision])</code>
Parameters	<ul style="list-style-type: none"> field: the numeric field to round precision: the number of decimal places

Description	The round function takes a number and output the rounded value
--------------------	--

Syntax	<code>split(<field>,<splitter>,<index>)</code>
Parameters	<ul style="list-style-type: none"> field: the field to split splitter: the character used to separate the string and produce the tokens index: the 0 based index of the token to output
Description	The split function takes a string, separates it and outputs the token at the <index> position

Syntax	<code>is_recent(<time_field>)</code>
Parameters	<ul style="list-style-type: none"> time_field: the field representing a time
Description	The is_recent function takes a time field and returns true if the time is not farther than 30 minutes

Syntax	<code>seconds_ago(<time_field>)</code>
Parameters	<ul style="list-style-type: none"> time_field: the field representing a time
Description	The seconds_ago function returns the amount of seconds passed between the current time and the time field value

Syntax	<code>minutes_ago(<time_field>)</code>
Parameters	<ul style="list-style-type: none"> time_field: the field representing a time
Description	The minutes_ago function returns the amount of minutes passed between the current time and the time field value

Syntax	<code>hours_ago(<time_field>)</code>
Parameters	<ul style="list-style-type: none"> time_field: the field representing a time
Description	The hours_ago function returns the amount of hours passed between the current time and the time field value

Syntax	<code>days_ago(<time_field>)</code>
Parameters	<ul style="list-style-type: none"> time_field: the field representing a time
Description	The days_ago function returns the amount of days passed between the current time and the time field value

Syntax	<code>to_time(<time_field>)</code>
Parameters	<ul style="list-style-type: none"> time_field: the field representing a time
Description	The to_time function takes a string representing a date and time in ISO 8601 format and returns the corresponding UNIX timestamp in milliseconds

Syntax	<code>bitwise_and(<numeric_field>,<mask>)</code>
Parameters	<ul style="list-style-type: none"> numeric_field: the numeric field on which apply the mask mask: a number that will be interpreted as a bit mask

Description The bitwise_and function calculates the bitwise & operator between the numeric_field and the mask entered by the user

Examples

Creating a pie chart

In this example we will create a pie chart to understand the MAC vendor distribution in our network. We choose `nodes` as our query source and we start to group the nodes by `mac_vendor`:

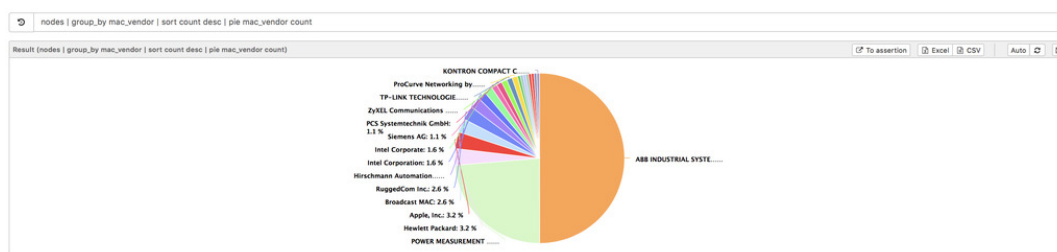
```
nodes | group_by mac_vendor
```

We can see the list of the vendors in our network associated with the occurrences count. To better understand our data we can use the `sort` command, so the query becomes:

```
nodes | group_by mac_vendor | sort count desc
```

In the last step we use the `pie` command to draw the chart with the `mac_vendor` as a label and the count as the value.

```
nodes | group_by mac_vendor | sort count desc | pie mac_vendor count
```



Creating a column chart

In this example we will create a column chart with the top nodes by traffic. We start by getting the `nodes` and selecting the `id`, `sent.bytes`, `received.bytes` and the `sum` of `sent.bytes` and `received.bytes`. To calculate the sum we use the `sum` function, the query is:

```
nodes | select id sent.bytes received.bytes sum(sent.bytes,received.bytes)
```

If we execute the previous query we notice that the sum field has a very long name, we can rename it to be more comfortable with the next commands:

```
nodes | select id sent.bytes received.bytes  
sum(sent.bytes,received.bytes)->sum
```

To obtain the top nodes by traffic we sort and take the first 10:

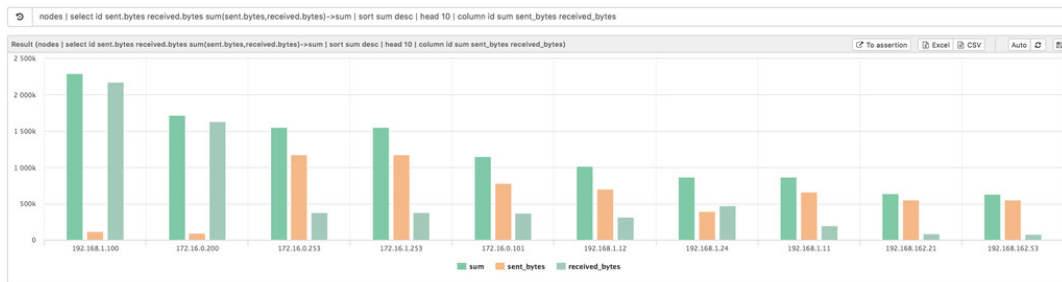
```
nodes | select id sent.bytes received.bytes  
sum(sent.bytes,received.bytes)->sum | sort sum desc | head 10
```

Finally we use the `column` command to display the data in a graphical way:

```
nodes | select id sent.bytes received.bytes  
sum(sent.bytes,received.bytes)->sum | sort sum desc | head 10 | column id  
sum sent_bytes received_bytes
```

Note: you can access an inner field of a complex type with the dot syntax, in the example the dot syntax is used on the fields `sent` and `received` to access their `bytes` sub field.

Note: after accessing a field with the dot syntax it will gain a new name to avoid ambiguity, the dot is replaced by an underscore. In the example `sent.bytes` become `sent_bytes`.



Using where with multiple conditions in OR

With this query we want to get all the nodes with a specific role, in particular we want all the nodes which are web servers or DNS server.

With the `where` command it is possible to achieve this by writing many conditions separated by `OR`.

Note: the `roles` field contains a list of values, thus we used the `include?` operator to check if a value was contained in the list.

```
nodes | where roles include? web_server OR roles include? dns_server |
select id roles
```

id	roles
192.168.1.1	["dns_server"]
172.16.0.1	["dns_server"]

Using bucket and history

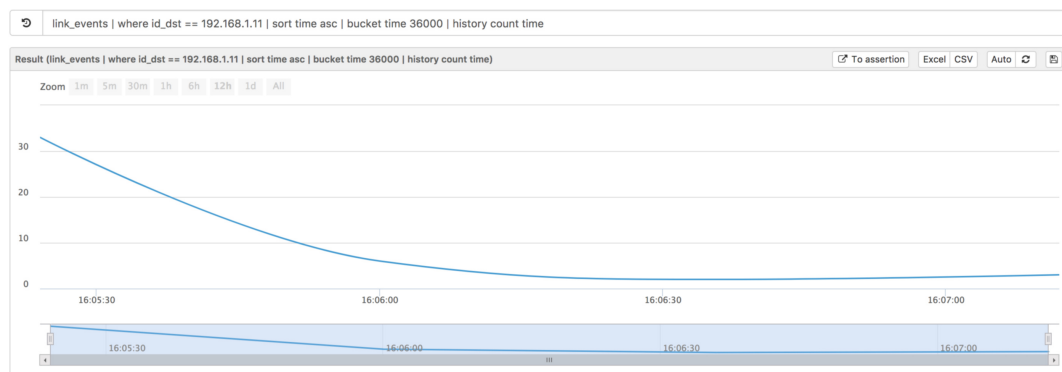
In this example we are going to calculate the distribution of link events towards an IP address. We start by filtering all the `link_events` with `id_dst` equal to 192.168.1.11.

After this we sort by time, this is a very important step because `bucket` and `history` depend on how the data are sorted.

At this point we group the data by time with `bucket`. The final step is to draw a chart using the `history` command, we pass `count` as a value for the Y axis and `time` for the X axis.

The `history` command is particularly suited for displaying a big amount of data, in the image below we can see that there are many hours of data to analyze.

```
link_events | where id_dst == 192.168.1.11 | sort time asc | bucket time
36000 | history count time
```



Using join

In this example we will join two data sources to obtain a new data source with more information. In particular we will list the links with the labels for the source and destination nodes.

We start by asking for the links and joining them with the nodes by matching the `from` field of the links with the `id` field of the nodes:

```
links | join nodes from id
```

After executing the query above we will get all the `links` fields plus a new field called `joined_node_from_id`, it contains the node which satisfies the `link.from == node.id` condition. We can access the sub fields of `joined_node_from_id` by using the dot syntax.

Because we want to get the labels also for the `to` field of the `links` we add another `join` and we exclude the empty labels of the node referred by `to` to get more interesting data:

```
links | join nodes from id | join nodes to id | where
  joined_node_to_id.label != ""
```

We obtain a huge amount of data which is difficult to understand, just use a `select` to get only the relevant information:

```
links | join nodes from id | join nodes to id | where
  joined_node_to_id.label != "" | select from joined_node_from_id.label to
  joined_node_to_id.label protocol
```

links | join nodes from id | join nodes to id | where joined_node_to_id.label != "" | select from joined_node_from_id.label to joined_node_to_id.label protocol

from	joined_node_from_id.label	to	joined_node_to_id.label	protocol
172.16.0.253		172.16.0.148	Modicon M340 BMX P34 2020	modbus
172.16.0.253		172.16.0.149	Modicon M340 BMX P34 2020	modbus
172.16.1.253		172.16.1.149	Modicon M340 BMX P34 2020	modbus
172.16.0.253		172.16.0.156	Modicon M340 BMX P34 2020	modbus
172.16.1.253		172.16.1.156	Modicon M340 BMX P34 2020	modbus
172.16.0.253		172.16.0.146	Modicon M340 BMX P34 2020	modbus
172.16.1.253		172.16.1.146	Modicon M340 BMX P34 2020	modbus
172.16.0.253		172.16.0.153	Modicon M340 BMX P34 2020	modbus
172.16.1.253		172.16.1.153	Modicon M340 BMX P34 2020	modbus
172.16.0.253		172.16.0.143	Modicon M340 BMX P34 2020	modbus

Computing availability history



In this example we will compute the availability history for a link. In order to achieve a reliable availability it is recommended to enable the *"Track availability"* feature on the desired link.

We start from the `link_events` data source, filtered by source and destination ip in order to precisely identify the target link. Consider also filtering by protocol to achieve a higher degree of precision.

```
link_events | where id_src == 10.254.3.9 | where id_dst == 172.31.50.2
```

The next step is to sort the events by ascending time of creation. Without this step the `availability_history` might produce meaningless results, such as negative values. Finally, we compute the `availability_history` with a bucket of 1 minute (60000 milliseconds). The complete query is as follows.

```
link_events | where id_src == 10.254.3.9 | where id_dst == 172.31.50.2 |
  sort time asc | availability_history 60000
```

Queries	
	link_events where id_src == 10.254.3.9 where id_dst == 172.31.50.2 sort time asc availability_history 60000
Result (link_events where id_src == 10.254.3.9 where id_dst == 172.31.50.2 sort time asc availa	
To assertion Excel 	
availability	time
100	09:01:00.000
34.075	09:02:00.000
21.41167	09:04:00.000
79.805	09:05:00.000
0	09:06:00.000
74.47167	09:08:00.000
25.78833	09:09:00.000
0	09:10:00.000
29.11167	09:11:00.000
71.36167	09:12:00.000

Chapter

5

Maintenance

Topics:

- [*System Overview*](#)
- [*Data Backup and Restore*](#)
- [*Reboot and shutdown*](#)
- [*Software Update and Rollback*](#)
- [*Data Factory Reset*](#)
- [*Support*](#)

In this chapter you will get the complementary information to keep the Nozomi Networks Solution up and running with ordinary and extraordinary maintenance tasks.

System Overview

In this section a brief overview of the Nozomi Networks Solution OS (N2OS) main components is given, so as to provide further background to administer and maintain a production system.

Partitions and Filesystem layout

In this section we will give a look at the N2OS filesystem, services and commands.

The first thing to know about the N2OS structure is the presence of four different disk partitions:

1. N2OS 1st partition, where a copy of the OS is kept and run from. Two different partitions are used by the install and update process in order to deliver fast-switch between the running release and new versions
2. N2OS 2nd partition, that copes with the first one to provide reliable update paths.
3. OS Configuration partition, located at /cfg, where low-level OS configuration files are kept (for instance, network configurations, shell admin users, SSH keys, etc). This partition is copied on /etc at the start of the bootstrap process.
4. Data partition, located at /data where all user data is kept (learned configuration, user-imported data, traffic captures, persistent database)

N2OS partition 1 (/)
N2OS partition 2 (backup /)
OS Configuration (/cfg)
Data partition (/data)

Figure 42: The N2OS standard partition table

A closer look at the /data partition reveals some sub-folders, for instance:

1. cfg: where all automatically learned and user-provided configurations are kept. Two main configuration files are stored here:
 - a. n2os.conf: for automatically learned configurations
 - b. n2os.conf.user: for additional user-provided configurations.
2. data: working directory for the embedded relational database, used for all persistent data
3. traces: where all [traces](#) are kept and rotated when necessary.
4. rrd: this directory holds the aggregated network statistics, used for example for the [Traffic](#) on page 31.

Core Services

There are some system services that you need to know for proper configuration and troubleshooting:

1. n2osids, the main monitoring process. It can be controlled with

```
service n2osids <operation>
```

(<operation> can be any of `start`, `stop`, `restart`). Its log files are under /data/log/n2os and start with n2os_ids*.

2. n2ostrace, the tracing daemon. It can be controlled with

```
service n2ostrace <operation>
```

Its log files start with n2os_trace* and are located under /data/log/n2os.

3. n2osva, the Asset Identification and Vulnerability Assessment daemon. It can be controlled with

```
service n2osva <operation>
```

Its log files start with n2os_va* and are located under /data/log/n2os.

4. n2ossandbox, the file sandbox daemon. It can be controlled with

```
service n2ossandbox <operation>
```

Its log files start with n2os_sandbox* and are located under /data/log/n2os.

5. nginx, the web server behind the web interface. It copes with unicorn to provide the https service up and secured. It can be controlled with

```
service nginx <operation>
```

In order to be able to perform any operation on these services, you need to obtain the privileges using enable-me. For instance, the following commands allow to restart the n2osids service:

```
enable-me
service n2osids restart
```

Several other tools and daemons are running in the system to deliver N2OS functionalities.

Data Backup and Restore

In this section you will get informed about available methods to backup the system and, respectively, to restore it from a backup. Please note that a backup will contain just the data -- **the system software will be left untouched**.

Two different kinds of backup are available: **Full Backup** and **Environment Backup**. The former contains all data while the latter lacks of historical data, extended configurations and some other information. Both can be executed while the system is running. **Environment Backup** can be used to restore the most important part of the system on another appliance for analysis or as delta backup when a full backup is available.

Full Backup

Command line

To create a new backup go to a terminal and execute command:

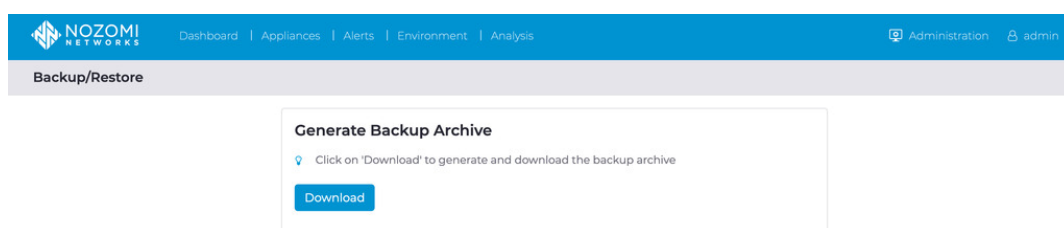
```
n2os-fullbackup
```

The backup file can now be copied through *SFTP* to a remote location of your choice. The file to copy is:

```
admin@<appliance_ip>:/data/tmp/<backup_hostname_date.nozomi_backup>
```

Web application

To create a new backup go to Administration > Backup/Restore and click on 'Download' button to generate and download the backup archive.



Full Restore

In this section you will learn how to restore from a full backup the N2OS software of an existing installation.

1. Copy via *SFTP* the backup archive from the location where it was saved to the `admin@<appliance_ip>:/data/tmp/<backup_hostname_date.nozomi_backup>` path of the appliance. For instance, using the *scp* command line:

```
scp <backup_location_path>/<backup_hostname_date.nozomi_backup>
admin@<appliance_ip>:/data/tmp/<backup_hostname_date.nozomi_backup>
```

2. Go to a terminal and execute this command

```
n2os-fullrestore /data/tmp/<backup_hostname_date.nozomi_backup>
```

Now you have completely restored the previous backup.

Environment Backup

In this section you will learn how to backup the Environment backup of an existing installation.

1. Issue the *save* command from the CLI
2. Copy via *SFTP* the content of the `/data/cfg` folder to a safe place.

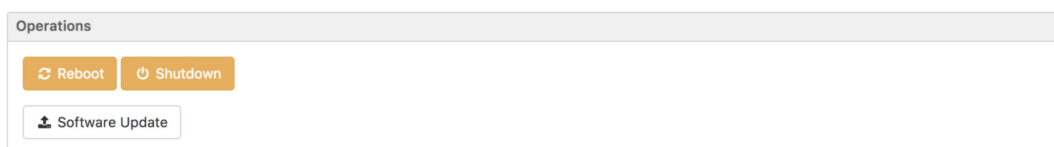
Environment Restore

In this section you will learn how to restore a Nozomi Networks Solution Environment to an existing installation.

1. Copy the saved content of the `cfg` folder to the `/data/cfg` folder into the appliance.
2. From the console, issue the `service n2osids restart` command.

Reboot and shutdown

Reboot and shutdown commands can be performed from the web interface under Administration > Operations



In addition, both commands can be entered in the text console or inside an SSH session.

To reboot the system issue the following command:

```
enable-me
shutdown -r now
```

To properly shutdown the system issue the following command:

```
enable-me
shutdown -p now
```

Software Update and Rollback

In this section you will get informed about available methods for updating the system to a newer release and rolling back to the previous one.

Rolling back to the previously installed release is transparent, and all data is migrated back to the previous format. However, rollback to a release older than the previously installed one requires to have a [full backup](#) available to restore.

Although the software update is built to be transparent to the user and to preserve all data, we suggest to always have at least a [Environment Backup](#) of the system in a safe place.

An interesting aspect of the Nozomi Networks Solution update file is that it applies to both the Guardian and the CMC, and will work for all the physical and virtual appliances to make the updating experience frictionless. Special considerations need to be done for the Container, where different update commands and procedures apply.

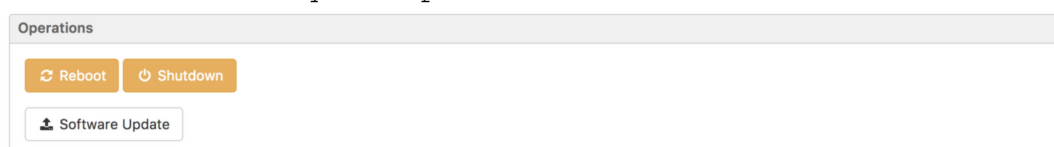
Update: Graphical method

In this section you will learn how to update the Nozomi Networks Solution software of an existing installation.

You need to already have the new *VERSION-update.bundle* file that you want to install.

A running system must be updated with a more recent N2OS release.

1. Go to Administration > System operations



2. Click on **Software Update** and select the *VERSION-update.bundle* file

Warning: the system must be at least version 18.5.9 to support the *.bundle* format; if your system is running a version lower than 18.5.9 you must first update to 18.5.9 to proceed

The file will be uploaded

3. Click the **Proceed** button

Warning: if updating version 18.5.9, the system prompts to insert the checksum that is distributed with the *.bundle*; only after checksum verification the button is enabled

The update process begins. Wait some minutes for the update to complete.

Update: Command line method

In this section you will learn how to update the Nozomi Networks Solution software of an existing installation.

You need to already have the new update file you want to install.

A running system must be updated with a more recent N2OS release.

1. Go to a terminal and `cd` into the directory where the *VERSION-update.bundle* file is located. Then copy the file to the appliance with:

```
scp VERSION-update.bundle admin@<appliance_ip>:/data/tmp
```

2. Start the installation of the new software with:

```
ssh admin@<appliance_ip>
```

```
enable-me
```

```
install_update /data/tmp/VERSION-update.bundle
```

The appliance will now reboot with the new software installed.

Rollback to the previous version

In this section you will learn how to rollback the software to the very previous version. If you would like to rollback to a release older than the previous one, follow the instructions in the next section.

You need to have performed a release update at least once.

1. Go to the console and type the command

```
rollback
```

2. Answer `y` to the confirmation message and wait while the system is rebooted. All configuration and historical data will be automatically converted to the previous version, thus no manual intervention will be required.

Rollback to an older version

In this section you will learn how to rollback the software to a version older than the previous one.

You need to have a full backup available. If you do not have one, you cannot rollback to an older release. Please note that this operations takes longer time than [rolling back to the very previous](#) one, requires a [full backup](#) and does not preserve recently changed data.

1. Take the software update file `VERSION-update.bundle` that you want to rollback to, and install it like if it was a new version as explained in [Software Update and Rollback](#) on page 67.

Warning: if you want to rollback to a version older than 18.5.9, you have to rollback to version 18.5.9 first since support for the previous file format was removed since

2. At reboot, ignore any error and log into the console.
3. Now follow the steps for a [full restore](#) with a backup file from the very same version of the software that has just been re-installed.

Data Factory Reset

In this section you will learn how to completely erase the N2OS data partition. IP configuration will be kept, and the procedure is safe to execute remotely. **Executing this procedure will cause the system to lose all data!**

1. Go to a terminal and execute the command:

```
n2os-datafactoryreset -y
```

2. The system will start over with a fresh data partition. Refer to [Setup Phase 2](#) on page 12 to complete the configuration of the system.

Data Factory Reset with Sanitization

In this section you will learn how to how to completely erase the N2OS data partition sanitizing the disk space using the U.S. DoD 5220-22M 7-pass scheme.

This process erase the N2OS data partition in accordance to the [clear guidelines suggested by the NIST in document 800-88 rev1](#).

Configurations like network and console password settings will be kept.

Executing this procedure will cause the system to lose all data!

1. Go to a terminal and execute the command:

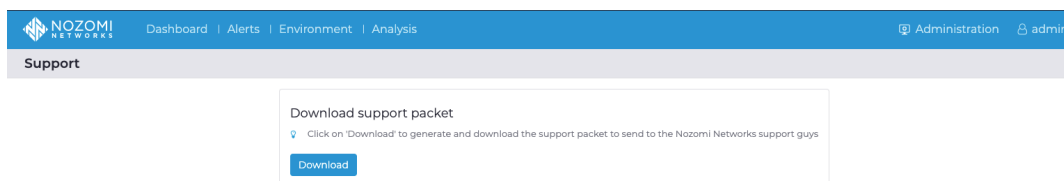
```
n2os-datasanitize -y
```

2. The system will start over with a fresh data partition. Refer to [Setup Phase 2](#) on page 12 to complete the configuration of the system.

Support

In this section you will learn how to generate the archive needed to ask support to Nozomi Networks.

Go to [Administration](#) > [Support](#) click on download button and your browser will start downloading the support packet file. Send an email to support@nozominetworks.com attaching the file.



Chapter 6

Scriptable protocols

Topics:

- [Setup](#)
- [Writing a scriptable protocol](#)
- [API reference](#)

In this manual we will cover the Lua scripting API for building a custom protocol decoder.

Setup

To add a new scriptable protocol:

1. Copy the Lua script in `/data/scriptable_protocols/`
2. Configure Guardian with this rule `probe scriptable-protocol <protocol_name> <script_name>` in `n2os.conf.user` (`<script_name>` is just the name of the file)
3. Execute `service n2osids stop`, the ids process will be restarted automatically.

After this steps the new protocol is loaded in Guardian and will analyze the network traffic.

Writing a scriptable protocol

The language used to write a scriptable protocol is Lua, please refer to the official Lua documentation (<https://www.lua.org/start.html>) to learn more.

This is a minimal protocol implementation:

```
function can_handle()
    return true
end
```

From the example we can see that the only mandatory thing to do is to define a function called `can_handle` which returns `true` if it recognize the target protocol.

Of course this implementation is not very useful and it will try to handle every packet so let's write something more complex to detect and analyze some modbus traffic:

```
function can_handle()
    return packet.source_port() == 502 or packet.destination_port() == 502
end
```

Here we can see a usage of the API to retrieve the packet ports. In this way the check is a bit more accurate but it's still insufficient to detect a modbus packet in the real world.

Let's start to do some deep packet inspection:

```
function can_handle()
    if data_size() < 8 then
        return false
    end

    local has_right_port = packet.source_port() == 502 or
        packet.destination_port() == 502

    fwd(2)
    local has_right_protocol_id = consume_n_uint16() == 0
    local expected_length = consume_n_uint16()

    return has_right_port and
        has_right_protocol_id and
        remaining_size() == expected_length
end
```

WARNING: don't use global variables. Variables defined outside of the `can_handle` and `update_status` functions are global and their status is shared across every session of the same protocol.

NOTE: the `fwd` and `consume_*` functions will move forward the payload pointer.

NOTE: the result of the `remaining_size` function depends on the position of the payload pointer.

In this example we use the API to inspect the content of the payload. First we check that there are enough bytes, a modbus packet is at least 8 bytes long. Then we check the port in the same we did in the previous example, then we skip two bytes with the function `fwd` and we read the next two 16 bit integers. We check that the protocol id is zero and that the length written in the packet match with the remaining bytes in our payload. If every check pass we return `true` saying to Guardian that the next packets in this session should be analyzed by this protocol decoder.

A protocol with just the `can_handle` function implemented will only create the node and the session in the network view but the link is still missing from the graph, no additional information will be displayed in the process view.

To extract more information from the modbus packets we are going to implement the `update_status` function:

```
function get_protocol_type()
    return ProtocolType.SCADA
end

function can_handle()
    return is_modbus()
end

function update_status()
    if not is_modbus() then
        return
    end

    local is_request = packet.destination_port() == 502
    local rtu_id = consume_uint8()
    local fc = consume_uint8() & 0x7f

    if is_request then
        is_packet_from_src_to_dst(true)
        set_roles("master", "slave")

        if fc == 6 then
            local address = consume_n_uint16()

            local value = DataValue.new()
            value.value = read_n_uint16()
            value.cause = DataCause.WRITE
            value.type = DataType.ANALOG
            value.time = packet.time()

            execute_update_with_variable(FunctionCode.new(fc), RtuId.new(rtu_id),
                "r"..tostring(address), value)
            return
        end
    end

    execute_update()
end
```

NOTE: to avoid duplication we created a `is_modbus` function from the content of the previous `can_handle` function.

NOTE: the `is_modbus` function has the effect to advance the payload pointer by 6 bytes, so when can directly read the `rtu_id` without further payload pointer manipulations.

NOTE: we defined the `get_protocol_type` function to define the protocol type

In this example of `update_status` we read more data from the payload and we decode the write single register request. We can understand the direction of the communication so we call `is_packet_from_src_to_dst` with `true` to notify Guardian and create a link and we call `set_roles` to set the roles on the involved nodes.

To insert a variable in Guardian there is the `execute_update_with_variable` function, it takes 4 arguments: the function code, the `rtu_id`, the variable name and the value. The `FunctionCode` and `RtuId` objects can be constructed from a string or a number, the `DataValue` object can be constructed with the empty constructor and then filled with the available information.

With the next example we cover a more complex case and we store some data in the session to handle a request and a response:

```

local PENDING_FC = 1
local PENDING_START_ADDR = 2
local PENDING_REG_COUNT = 3

function update_status()
  if not is_modbus() then
    return
  end

  rwd()

  local is_request = packet.destination_port() == 502
  local transaction_id = consume_n_uint16()
  fwd(4)

  local rtu_id = consume_uint8()
  local fc = consume_uint8() & 0x7f

  if is_request then
    is_packet_from_src_to_dst(true)
    set_roles("master", "slave")
    session.set_pending_request_number(transaction_id, PENDING_FC, fc)

    if fc == 3 then
      if remaining_size() < 4 then
        return
      end

      local start_addr = consume_n_uint16()
      local registers_count = consume_n_uint16()

      session.set_pending_request_number(transaction_id, PENDING_START_ADDR,
start_addr)
      session.set_pending_request_number(transaction_id, PENDING_REG_COUNT,
registers_count)
    end
  else
    is_packet_from_src_to_dst(false)
    local req_fc = session.read_pending_request_number(transaction_id,
PENDING_FC)

    if fc == req_fc then
      if fc == 3 then
        local start_addr = session.read_pending_request_number(transaction_id,
PENDING_START_ADDR)
        local reg_count = session.read_pending_request_number(transaction_id,
PENDING_REG_COUNT)
        session.close_pending_request(transaction_id)

        if remaining_size() < 1 then
          return
        end

        local byte_count = consume_uint8()

        if remaining_size() ~= byte_count or
          reg_count * 2 ~= remaining_size() then
          send_alert_malformed_packet("Packet is too small")
          return
        end

        for i = 0, reg_count - 1, 1 do
          local value = DataValue.new()
          value.value = consume_n_uint16()
        end
      end
    end
  end
end

```

```

    value.cause = DataCause.READ_SCAN
    value.type = DataType.ANALOG
    value.time = packet.time()

    execute_update_with_variable(FunctionCode.new(fc),
                                RtuId.new(rtu_id),
                                "r"..tostring(start_addr+i),
                                value)
end

    return
end
end
end

execute_update()
end

```

This time we are focusing on the read holding register function code, to understand the communication and create a variable we need to analyze both the request and the response and we need to keep some data from the request and use it in the response. To achieve this we can use the functions provided by the `session` object.

API reference

Available LUA libraries

- base
- string
- table
- math
- bit32
- debug
- utf8

Data types

Class FunctionCode	
Constructors	<ul style="list-style-type: none"> • FunctionCode.new(<string>) • FunctionCode.new(<number>)
Class RtuId	
Constructors	<ul style="list-style-type: none"> • RtuId.new(<string>) • RtuId.new(<number>)
Class DataValue	
Constructors	<ul style="list-style-type: none"> • DataValue.new()
Read/write properties	<ul style="list-style-type: none"> • DataValue.value (number) • DataValue.str_value (string) • DataValue.cause (DataCause) • DataValue.time (number, milliseconds since epoch) • DataValue.type (DataType)
Class Variable	
Methods	<ul style="list-style-type: none"> • set_label(<string>)
Class Node	
Methods	<ul style="list-style-type: none"> • set_property(<key>, <value>) • get_property(<key>) • delete_property(<key>)
Enum DataCause	
Values	<ul style="list-style-type: none"> • DataCause.READ_SCAN • DataCause.READ_CYCLIC • DataCause.READ_EVENT • DataCause.WRITE
Enum DataType	

Values	<ul style="list-style-type: none"> • <code>DataType.ANALOG</code> • <code>DataType.DIGITAL</code> • <code>DataType.BITSTRING</code> • <code>DataType.STRING</code> • <code>DataType.DOUBLEPOINT</code> • <code>DataType.TIMESTAMP</code>
Enum	<code>ProtocolType</code>
Values	<ul style="list-style-type: none"> • <code>ProtocolType.SCADA</code> • <code>ProtocolType.NETWORK</code>

Functions

Syntax	<code>data(<index>)</code>
Parameters	<ul style="list-style-type: none"> • <code>index</code>: the position of the byte to read, starting from 0
Description	Return the value of the byte from the specified position, return 0 if index is out of bounds
Syntax	<code>data_size()</code>
Description	Return the total size of the payload
Syntax	<code>remaining_size()</code>
Description	Return the size of the payload from the pointer to the end. The result depends on the usage of functions <code>fwd()</code> , <code>rwd()</code> and <code>consume_*</code> .
Syntax	<code>fwd(<amount>)</code>
Parameters	<ul style="list-style-type: none"> • <code>amount</code>: the number of bytes to skip
Description	Move the payload pointer by the specified number of bytes.
Syntax	<code>rwd()</code>
Description	Move the payload pointer to the beginning of the payload.
Syntax	<code>read_uint8()</code>
Description	Read an unsigned 8bit integer at the payload pointer position.
Syntax	<code>read_int8()</code>
Description	Read an signed 8bit integer at the payload pointer position.
Syntax	<code>read_n_uint16()</code>
Description	Read a network order unsigned 16bit integer at the payload pointer position.
Syntax	<code>read_h_uint16()</code>
Description	Read a host order unsigned 16bit integer at the payload pointer position.
Syntax	<code>read_n_int16()</code>

Description	Read a network order signed 16bit integer at the payload pointer position.
Syntax	<code>read_h_int16()</code>
Description	Read a host order signed 16bit integer at the payload pointer position.
Syntax	<code>read_n_uint32()</code>
Description	Read a network order unsigned 32bit integer at the payload pointer position.
Syntax	<code>read_h_uint32()</code>
Description	Read a host order unsigned 32bit integer at the payload pointer position.
Syntax	<code>read_n_int32()</code>
Description	Read a network order signed 32bit integer at the payload pointer position.
Syntax	<code>read_h_int32()</code>
Description	Read a host order signed 32bit integer at the payload pointer position.
Syntax	<code>read_n_uint64()</code>
Description	Read a network order unsigned 64bit integer at the payload pointer position.
Syntax	<code>read_h_uint64()</code>
Description	Read a host order unsigned 64bit integer at the payload pointer position.
Syntax	<code>read_n_int64()</code>
Description	Read a network order signed 64bit integer at the payload pointer position.
Syntax	<code>read_h_int64()</code>
Description	Read a host order signed 64bit integer at the payload pointer position.
Syntax	<code>read_n_float()</code>
Description	Read a network order float at the payload pointer position.
Syntax	<code>read_h_float()</code>
Description	Read a host order float at the payload pointer position.
Syntax	<code>read_n_double()</code>
Description	Read a network order double at the payload pointer position.
Syntax	<code>read_h_double()</code>
Description	Read a host order double at the payload pointer position.
Syntax	<code>read_string()</code>
Description	Read a string at the payload pointer position until the null terminator.
Syntax	<code>read_string_with_len(str_len)</code>

Description	Read a string at the payload pointer position for <code>str_len</code> bytes.
Syntax	<code>consume_uint8()</code>
Description	Read an unsigned 8bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_int8()</code>
Description	Read an signed 8bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_n_uint16()</code>
Description	Read a network order unsigned 16bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_h_uint16()</code>
Description	Read a host order unsigned 16bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_n_int16()</code>
Description	Read a network order signed 16bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_h_int16()</code>
Description	Read a host order signed 16bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_n_uint32()</code>
Description	Read a network order unsigned 32bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_h_uint32()</code>
Description	Read a host order unsigned 32bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_n_int32()</code>
Description	Read a network order signed 32bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_h_int32()</code>
Description	Read a host order signed 32bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_n_uint64()</code>
Description	Read a network order unsigned 64bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_h_uint64()</code>

Description	Read a host order unsigned 64bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_n_int64()</code>
Description	Read a network order signed 64bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_h_int64()</code>
Description	Read a host order signed 64bit integer at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_n_float()</code>
Description	Read a network order float at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_h_float()</code>
Description	Read a host order float at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_n_double()</code>
Description	Read a network order double at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_h_double()</code>
Description	Read a host order double at the payload pointer position and move the pointer after the data.
Syntax	<code>consume_string()</code>
Description	Read a string at the payload pointer position until the null terminator and move the pointer after the data.
Syntax	<code>consume_string_with_len(str_len)</code>
Description	Read a string at the payload pointer position for <code>str_len</code> bytes and move the pointer after the data.
Syntax	<code>set_roles(<client_role>, <server_role>)</code>
Parameters	<ul style="list-style-type: none"> <code>client_role</code>: the role of the client <code>server_role</code>: the role of the server
Description	Set the roles of the involved nodes, valid values are: "master", "slave", "historian", "terminal", "web_server", "dns_server", "db_server", "time_server", "other"
Syntax	<code>set_source_type(<node_type>)</code>
Parameters	<ul style="list-style-type: none"> <code>node_type</code>: the type of the source node
Description	Set the type of the source node, valid values are: "switch", "router", "printer", "group", "OT_device", "broadcast", "computer"

Syntax	<code>variables_are_on_client()</code>
Parameters	
Description	Notify to Guardian that the variables should be added to the client node

syntax	<code>is_packet_from_src_to_dst(<is_from_src>)</code>
parameters	<ul style="list-style-type: none"> • <code>is_from_src</code>: true is the direction is from src to dst, false otherwise
description	notify Guardian about the direction of the packet, this function must be called to obtain a link creation

syntax	<code>execute_update()</code>
parameters	
description	notify Guardian about the a packet, at least one variant of <code>execute_update</code> should be called for every packet

syntax	<code>execute_update_with_function_code(<function_code>, <rtu id>)</code>
parameters	<ul style="list-style-type: none"> • <code>function_code</code>: an object of type functioncode • <code>rtuid</code>: an object of type rtuid
description	notify Guardian about the a packet with a function code and a rtu id

syntax	<code>execute_update_with_variable(<function_code>, <rtu id>, <var_name>, <value>)</code>
parameters	<ul style="list-style-type: none"> • <code>function_code</code>: an object of type functioncode • <code>rtu_id</code>: an object of type rtuid • <code>var_name</code>: the name of the variable • <code>value</code>: an object of type datavalue containing the value of the variable and some information about the data
description	notify Guardian about the a packet with a function code, a rtu id, a variable name and a variable value

syntax	<code>execute_update_with_function(<function_code>, <rtu id>, <var_name>, <value>, <function>)</code>
parameters	<ul style="list-style-type: none"> • <code>function_code</code>: an object of type functioncode • <code>rtu_id</code>: an object of type rtuid • <code>var_name</code>: the name of the variable • <code>value</code>: an object of type datavalue containing the value of the variable and some information about the data • <code>function</code>: the function will be called passing <code>variable</code> as an argument
description	notify Guardian about the a packet with a function code, a rtu id, a variable name, a variable value and a function that give the possibility to directly access the variable

syntax	<code>send_alert_malformed_packet(<reason>)</code>
parameters	<ul style="list-style-type: none"> • <code>reason</code>: a message to be displayed in the alert
description	raise an alert of type sign:network-malformed or sign:scada-malformed

syntax	<code>packet.source_id()</code>
description	return the source node id

syntax	<code>packet.destination_id()</code>
description	return the destination node id

syntax	<code>packet.source_ip()</code>
description	return the source node ip

syntax	<code>packet.destination_ip()</code>
description	return the destination node ip

syntax	<code>packet.source_mac()</code>
description	return the source node mac

syntax	<code>packet.destination_mac()</code>
description	return the destination node mac

syntax	<code>packet.source_port()</code>
description	return the source node port

syntax	<code>packet.destination_port()</code>
description	return the destination node port

syntax	<code>packet.is_ip()</code>
description	return true if the packet is an ip packet

syntax	<code>packet.transport_type()</code>
description	return the transport layer type, can be "tcp", "udp", "ethernet", "icmp" or "unknown"

syntax	<code>packet.source_node()</code>
description	returns the source node

syntax	<code>packet.destination_node()</code>
description	returns the destination node

syntax	<code>packet.time()</code>
description	return the packet time

syntax	<code>session.set_pending_request_number(<request_id>, <key>, <value>)</code>
parameters	<ul style="list-style-type: none"> request_id: a number used to uniquely identify the request key: a number used to separate different values in the same request value: the number to store

description	store a number on the session
--------------------	-------------------------------

syntax	<code>session.read_pending_request_number(<request_id>, <key>)</code>
parameters	<ul style="list-style-type: none"> request_id: a number used to uniquely identify the request key: a number used to separate different values in the same request
description	read a number from the session

syntax	<code>session.set_pending_request_string(<request_id>, <key>, <value>)</code>
parameters	<ul style="list-style-type: none"> request_id: a number used to uniquely identify the request key: a number used to separate different values in the same request value: the string to store
description	store a string on the session

syntax	<code>session.read_pending_request_string(<request_id>, <key>)</code>
parameters	<ul style="list-style-type: none"> request_id: a number used to uniquely identify the request key: a number used to separate different values in the same request
description	read a string from the session

syntax	<code>session.has_pending_request(<request_id>)</code>
parameters	<ul style="list-style-type: none"> request_id: a number used to uniquely identify the request
description	return true if there are values stored with the request_id

syntax	<code>session.has_pending_request_value(<request_id>, <key>)</code>
parameters	<ul style="list-style-type: none"> request_id: a number used to uniquely identify the request key: a number used to separate different values in the same request
description	return true if there are values stored with the request_id and key

syntax	<code>session.close_pending_request(<request_id>)</code>
parameters	<ul style="list-style-type: none"> request_id: a number used to uniquely identify the request
description	close the pending request and delete the associated data

syntax	<code>log_d(<msg>)</code>
parameters	<ul style="list-style-type: none"> msg: the message to log
description	log a debug message

syntax	<code>log_e(<msg>)</code>
parameters	<ul style="list-style-type: none"> msg: the message to log
description	log an error message