# Nexthink V6.23

## Glossary and References

Generated: 3/03/2020 7:35 am

# Table of Contents

# Table of Contents

# Table of Contents

# Glossary

## Activity

An *activity* is a specific action detected in your IT infrastructure in which one or more of the objects monitored by Nexthink are involved. Nexthink supervises the following types of activities:

- Installation
- Execution
- Connection
- Web request
- Print job
- System boot
- User logon

Related concepts

- Object
- Event

## Alert

Alerts warn you of particular situations in your IT infrastructure. An alert usually indicates the occurrence of an issue that needs to be addressed. The idea behind alerts is that you do not need to look constantly for problematic cases in the system, but it is the system that can detect them and notify you when they happen. There are three types of alerts:

System alerts
: Predefined alerts that reveal special circumstances in the operation of the system.

Investigation-based alerts
: Triggered by conditions that you can specify in the Finder using investigations.

Service-based alerts
: Triggered by warning or error conditions on the real-time services that you add to your Portal.

Related tasks

- Receiving alerts
- Creating a service-based alert
- Creating an investigation-based alert

Related references

- System alerts

# Application

An *application* is an object that represents a set of executable programs bundled by a software manufacturer as a single software product.

Applies to platforms:

Related concepts

- Executable
- Binary
- Package
- Object

# Binary

A *binary* is an object that represents the physical image on disk of a particular version of an executable file. The sequence of bytes in the executable file completely characterizes the binary. To reduce the quantity of information needed, Nexthink does not identify a binary by its complete sequence of bytes, but by a hash number computed from this sequence.

Applies to platforms:

Thus, executable files that share the same name are only represented by the same binary in Nexthink if they share exactly the same sequence of bytes in disk as well. Two executable files with the same name may have a different sequence of bytes because of two reasons:

Versioning
: If the versions of the executable files are different, the files have a different binary image. Therefore, a binary is created for each version of the executable file.

Modified executable file
: If one of the executable files has been altered by any means, the binary image is necessarily different even when the versions of the executable files are the same. A modified file is an indication of malware.

Nexthink detects the existence of a binary in your IT infrastructure the first time that the binary is run on one of the monitored devices.

Related references

- Binary paths
- Maximum number of Binaries

Related concepts

- Application
- Executable
- Object

# Campaign

A *campaign* is a set of questions addressed to a group of users with the goal of getting their opinion on a particular matter. Usually, the subject matter of a campaign relates to the users' perception of the IT services and equipment that your organization provides to them.

Create your campaigns in the Finder and assign different target audiences for each campaign by using investigations on users. The targeted end-users receive notifications in their Collector equipped devices for answering the questions of your campaigns.

Track the results of your campaigns using investigations and metrics, as well as dashboards to follow their evolution.

Related tasks

- Creating a campaign

# Category

A *category* is a way to group objects of the same type into user-defined classes. Each class is identified by a keyword or tag. Only users with the right privilege level can create categories and keywords.

Related tasks

- Creating categories and keywords

Related concepts

- Keyword
- Object
- Hierarchy

# Connection

A *connection* is a link between a device and a destination through the use of network resources. There are two types of connections depending on the transport protocol used for communication:

TCP
: The connection has a status.

UDP
: The connection is stateless.

For TCP connections, the link between device and destination does not need to be fully established for Nexthink to record the connection. For UDP connections, it is not possible to know if the connection was established, due to the own nature of the protocol. Thus, for any protocol, every connection attempt that Nexthink detects is recorded on the Nexthink database no matter whether the connection is successful or not.

Some repetitive short-lived connections are automatically grouped into one single aggregated connection when some sort of scanning is detected:

Network scan
: A repeated attempt to connect to the same port on several destinations.

Port scan
: A repeated attempt to connect to different ports on the same destination.

Network and port scans can be launched by legitimate processes, but they may also indicate the existence of malicious activity in your network. Scanning communication ports in one or several machines is a widely used method to detect vulnerabilities in computer networks.

Only connections using the same transport protocol may be grouped into a single scan connection. Thus, there are four types of scan connections:

| Protocol \ Type | Network | Port |
|---|---|---|
| **TCP** | TCP network scan | TCP port scan |
| **UDP** | UDP network scan | UDP port scan |

See Network and port scan conditions to find out when Nexthink regards a set TCP and UDP connections as a scanning operation.

Related concepts

- Activity
- Port
- Device
- Destination

Related references

- Status of TCP connections
- Status of UDP connections
- Network and port scan conditions

# Dashboard

A *dashboard* is a panel with selected information elements about your IT infrastructure that is displayed as a web page in the Portal.

Related tasks

- Creating a dashboard

Related concepts

- Module
- Widget

# Destination

A *destination* is a computer that accepts incoming network connections. Therefore, destinations play the role of the server in the client-server model of communication. Devices, on the other hand, usually play the role of the client. However, devices can also function as servers in some cases. Devices that accept connections are thus listed both as devices and as destinations.

Applies to platforms:

Destinations are identified by their IP address. There are two special kinds of destinations:

external
: Destination that falls outside the networks supervised by the Engine.

multiple
: A virtual destination that represents a network scanning.

Related concepts

- Connection
- Device

# Device

A *device* is an object that represents any personal computer or computerized system from which Nexthink can extract information to produce end-user analytics.

Nexthink classifies devices into four types:

**Desktop**
: A conventional or virtualized personal computer running a client version of either Windows or Mac operating systems.

**Laptop**
: A portable personal computer running a client version of either Windows or Mac operating systems.

**Server**
: A computer that provides functionality (e.g. email, web, or directory services) to client computers (desktops and laptops) and runs a version of the Windows Server operating system.

**Mobile**
A phone or tablet equipped with Exchange ActiveSync.

See the list of operating systems supported by the Collector.

## Licensing

The licensing model of Nexthink is based on the number of devices of the different types in your setup. For licensing purposes, desktops and laptops are both considered **endpoints**; that is, regular devices that support the installation of the Collector.

Thus, when licensing your product, provide the number of:

- Endpoints (desktops plus laptops)
- Servers
- Mobile devices

Applies to platforms:
Related concepts

- Object

Related references

- Operating systems suported by the Collector
- Setting up a software license

# Domain

A *domain* is an object that represents a realm of administrative authority on the Internet and is identified by a name. Domain names are formed using the rules and procedures described in the Domain Name System (DNS). Domain names are organized into levels (subdomains), being the top-level domains the country codes, and the well-known *com*, *org*, *net* or *edu*, among others. Organizations typically register second or third-level domain names through accredited registrar companies to publicly offer their services on the Internet; for example `www.nexthink.com`. Note that subdomain levels in a domain name are inversed with respect to their hierarchy (top-level subdomains are placed last). The main purpose of the DNS is to identify areas of the Internet with easy to memorize names and to translate those names into numerical IP addresses that

routing devices understand. A complete domain name with all levels specified is known as a Fully Qualified Domain Name (FQDN).

Applies to platforms:

Domain names are a central part of Uniform Resource Locators (URLs), which reference individual resources in the Internet; for example `http://www.nexthink.com/what-is-nexthink/`. A URL can refer to a web page, a text file, an image, a video stream or any other kind of resource in the Internet. The first part of a URL is called the *scheme*. The scheme usually designates the protocol used in the connection, such as ftp or http. In web browsers, users typically type URLs in the navigation bar to retrieve a particular web page, but other applications may use URLs internally to get information from the Internet without necessarily displaying them.

Nexthink records the domains of all the web requests initiated from a monitored device, regardless or the application that made the request. Nexthink considers a connection to be a web request when the scheme of the URL is **http** or **https**; that is, when the connection uses the Hypertext Transfer Protocol (HTTP) or the Hypertext Transfer Protocol Secure (HTTPS), which is an encrypted version of HTTP with Transport Layer Security (TLS).

## Domain compaction

To avoid storing too many names for web domains, Nexthink has a strategy for compacting the names of those domains that share a common root, grouping them under a single name. On the other hand, internal domains are never compacted. In turn, domains that match the pattern defined in a web-based service are compacted only up to the point that the specified filters allow it. For instance:

- If a web-based service has a filter on domains ***.example.com**:
  A web request to **mail.example.com** is compacted to ***.example.com**.
- However, if an additional web-based service specifies the filter on domains **mail.*.com**:
  The domain **mail.example.com** is not compacted, as it must match both filters.

By default, Nexthink compacts domains when the domain name consists of more than five subdomain levels, or when the third or lower levels are repetitive (names with indexes) or automatically generated (random letters and digits). In those cases, the lower subdomains are replaced by the asterisk sign *. See the

table below for a few examples of compacted domains and a last example of a domain that is not compacted:

| Domain name | Stored domain |
|---|---|
| `exceed.just.five.domain.levels.com`<br>`again.exceed.just.five.domain.levels.com` | `*.just.five.domain.levels.com` |
| `dev01.cloud.example.com`<br>`dev02.cloud.example.com`<br>`8d271d.cloud.example.com` | `*.cloud.example.com` |
| `svn.cloud.example.com` | `svn.cloud.example.com` |

Note that compacted domains and FQDNs belonging to a same higher level domain can coexist in Nexthink. For instance, in the table above, both the compacted `*.cloud.example.com` and the FQDN `svn.cloud.example.com` are subdomains of `cloud.example.com`, but they are stored as separate domains in the Nexthink database. Thus, the asterisk does not refer to all the subdomains inside `cloud.example.com`, but only to those which are repetitive or randomly generated.

You can also specify a more aggressive compaction method that applies to all domains and not only to those complying with the pre-requisites above. The compaction in this case is made according to a vendor independent public list of domain suffixes, used to determine the highest level at which a domain can be registered. In this way, all domains are compacted up to the level that includes the name of the organization that registered the domain. From the Web Console, configure the compaction policy for domain names in the Engine.

## Domain replacement

Because web visits are very common, medium to large setups hit the maximum number of visited domains often, even when aggressive compaction methods are put in place.

To keep the list of visited domains up-to-date, starting from V6.18 the Engine reserves 20% of the maximum number of domains (by default, 50 000 out of 250 000 domains) to record the domains visited throughout the day. During the nightly cleanup of the Engine, if the number of stored domains exceeds 80% of the total capacity (by default, 200 000 domains), the domains which have not been visited for a longer period of time will be removed from the list.

## Domain category

The following categories exist:

**General:**

- Business application
- Search engine and portals
- Information technology
- Social
- News and information
- Advertisement and marketing
- Internal
- Other

**Communication:**

- VoIP [beware of special capitalization]
- Instant messaging
- Email

**High bandwidth:**

- Network storage
- Peer-to-peer
- Video, image and sound

**Potentially unwanted:**

- Games
- Proxy avoidance and hacking
- Spam
- Freeware and software download
- Malicious

Related tasks

- Specifying your internal networks and domains
- Establishing a data retention policy in the Engine

Related concepts

- Service

Related references

- Nightly task schedules timetable
- Public suffix list (external)

# Entity

An *entity* is a logical grouping of devices which are all reporting to the same Engine.

Entities are the basic building blocks of hierarchies. In a hierarchy tree, the entities are the leaf nodes.

The Service view of the Finder uses entities to provide a breakdown of the devices and the Engine computes some service-related errors and warnings at the entity level.

Related tasks

- Hierarchizing your infrastructure

Related concepts

- Hierarchy

Related references

- Service errors and warnings

# Event

An *event* is the basic unit of information that the Engine stores about meaningful occurrences within your IT infrastructure. All objects and activities in Nexthink are linked to one or more events. When an object or activity is no longer related to any event, it is eventually removed from the system. Events are timestamped and ordered sequentially using both the clock of the Engine and the local time reported by the Collectors.

In the Finder, within the context of investigations, *events* refer only to those basic events that represent errors or warnings from devices or applications. The

Device view of the Finder also displays the occurrences of these events in the **Errors** and **Warnings** timelines.

Related concepts

- Object
- Activity

Related references

- Errors and warnings for devices and executions
- Warnings tooltips
- Errors tooltips

# Executable

An *executable* is an object that encompasses all the binaries that refer to the same program. For instance, the executable `nxfinder.exe` comprises all the binary files with that name that refer to the different versions of the Nexthink Finder.

Applies to platforms:

Related concepts

- Binary
- Application
- Object

# Execution

An *execution* is an activity that indicates the loading of a binary into the memory of a computer to run it as a separate process.

If two (or more) executions of the same binary are separated by less than a few minutes, the Engine aggregates them into a single execution. The duration of the aggregated execution spans from the start of the first execution to the end of the last execution. The *cardinality* of the aggregated execution equals to the number of different independent executions.

This mechanism naturally groups repeated or parallel executions of the same program, helping the Engine save space for data retention.

Related concepts

- Activity
- Binary
- Executable

## Hierarchy

A *hierarchy* is a way to organize your devices into a structured set of levels.

Related tasks

- Hierarchizing your infrastructure

Related concepts

- Category
- Entity

## Installation

An *installation* is an activity that makes a software package available for use in a device. Nexthink detects both the installation and removal of packages in the devices of end-users.

Related concepts

- Activity
- Package

## Investigation

An *investigation* is a construct to query the Nexthink database. You can run investigations on objects, activities, or events.

Related tasks

- Creating an investigation

Related concepts

- Object
- Activity
- Event

## Keyword

A *keyword* or *tag* is a label to differentiate objects of the same type according to a category. For example, the keywords of the predefined category **Application type** help you distinguish among different types of applications by identifying them as **browser**, **mail client** or **antivirus**.

The process of assigning a keyword to an object is called *tagging*. An object can be tagged with at most one keyword per suitable category. Tagging can be either manual or automatic.

Related concepts

- Category

## Metric

A *metric* is a quantifiable measure of a part or a feature of your IT infrastructure in which you are particularly interested. Metrics offer you a way to define key indicators out of selected IT objects, their properties and their activities.

Define metrics in the Finder in a similar way to how you define investigations and follow their evolution over time from the Portal. Metrics are computed daily during the nightly data collection of the Portal.

Depending on what you want to measure, choose among three types of metrics:

Count metric
: A count metric measures the number of objects that fulfil a custom set of conditons. Count metrics optionally include a ratio between the number of counted objects (the value of the metric itself) and the number of objects that fulfil a second set of conditions, usually more relaxed than the

conditions defining the metric.

Quantity metric
: A quantity metric groups the individual values of a particular numeric quantity (aggregate, score, etc.) of devices or users and computes a single value out of them. Depending on the quantity, choose among getting the average, the overall sum, the maximum, or the minimum of all the individual values of the aggregate. Note that quantity metrics are only available for objects of types *device* and *user.*

Top metric
: A top metric holds an ordered list of objects that have the highest or lowest quantity of an aggregated value. Specify the number of elements in the list and the way to compute the aggregated value. Compute either the average, the sum, the maximum, or the minimum of the individual aggregate values for each object, depending on the type of aggregate.

Related tasks

- Creating a metric

# Module

Depending on the context, a *module* can refer to:

- One of the parts of Nexthink that can be purchased independently from the core product, adding extra capabilities to it.
- A container of Portal dashboards.

Related concepts

- Dashboard
- Widget

# Object

An *object* is a representation of an identifiable element in your IT infrastructure whose properties, activities and generated events are monitored by Nexthink. An object can refer either to a hardware or to a software element.

There are ten types of objects in Nexthink. The availability of each type of object and the information that they carry depend on the platform of the end-user

devices that originated them. Object types that are available for multiple platforms can be divided into *specific* to a platform and *shared* by platforms. Objects which are specific to a platform belong to one platform only. For instance, devices are platform specific because one device may be either a Windows device, a Mac OS device, or a Mobile device. On the other hand, users are platform shared, because the same user may have been seen in Windows, Mac OS, or Mobile.

| Available for platforms | Object type | Platform specificity |
|---|---|---|
| | User | Shared |
| | Device | Specific |
| | Package | Specific |
| | Application | Specific |
| | Executable | Specific |
| | Binary | Specific |
| | Port | Shared |
| | Destination | Shared |
| | Domain | Shared |
| | Printer | Shared |

Related concepts

- Activity
- Event

# Package

A *package* is an object that represents a software product in its distributable form. Packages can be installed on a device or uninstalled from a device.

Applies to platforms:

Nexthink classifies packages into *program* packages and *update* packages:

Program
: A program package holds a complete software product ready for installation.

Update
: An update package, also called a *patch*, holds bug fixes, security fixes and any other kind of improvements for a program package.

In the Windows operating system, the dialog **Add** / **Remove Programs** in the Control Panel shows the list of packages installed in the computer.

Packages should not be confused with applications. Whereas packages represent a software product in its installable form, applications represent a software product in its executable form. Accordingly, Nexthink detects the presence of a package when it is installed and acknowledges the presence of an application when it is executed; that is, when one of the executable files that make up the application is run.

The link between an application and a corresponding package cannot be always established, so you cannot navigate from one to the other in the Finder. Still, the link is sometimes known for individual executables of the application. If this is the case, you can find the name of the package in the field **packages** of the executable. One executable can in fact be linked to several packages, yet only the names of the packages are given and drilling-down from executables to packages is not allowed either.

Related concepts

- Installation
- Application
- Executable
- Object

# Platform

The *platform* of a device is linked to the way Nexthink extracts information from it, either using the Windows Collector, the Mac OS Collector or the Mobile Bridge. The platform relates thus to the underlying hardware or operating system family of the device. Nexthink supports three types of platforms:

- Windows platform
- Mac OS platform
- Mobile platform

Investigations may apply to one or more platforms. When multiple platforms are selected in an investigation, only the fields that are common to all of the selected platforms are available in the results of the investigation.

Besides devices, other objects may depend on the platform of the device that generated them.

Related concepts

- Object
- Device

# Port

A *port* is a communication resource identified by a transport protocol, TCP or UDP, and a number between 1 and 65  535. Ports are associated to connections. A device connects to a destination through a particular port.

Applies to platforms:

In reality, two ports are needed to establish a connection: one port in the device and another port in the destination. However, only the port of the destination is important, because it is the one that usually determines the network service. Server applications on destinations listen to connection requests on well-known ports that are associated to standardized network services. For example, the SMTP mail service uses TCP port 25 for incoming requests. For this reason, Nexthink stores information about the destination ports only.

The type of the port is determined by the transport protocol. There are two types of ports:

- TCP
- UDP

In addition to TCP and UDP ports, there are two other special types of ports in Nexthink which are associated to connections identified as port scans:

- TCP port scan
- UDP port scan

These special types of ports are not linked to just one but to multiple port numbers. To reflect this particularity, Nexthink assigns to these ports the port number 0. which is actually a reserved port number that is never used.

Related concepts

- Connection
- Device
- Destination

- Object

Related references

- Network and port scan conditions

# Print job

A *print job* is an activity that represents a unit of work sent to a printer. A single print job may consist of one or several files to be printed. A print job puts in relationship a user and a device with a printer.

Related references

- Information on printers and printing

Related concepts

- Printer
- Activity

# Printer

A *printer* is an object representing a computer peripheral that is capable of printing text or pictures on a physical support, typically paper.

Applies to platforms:

Related references

- Information on printers and printing

Related concepts

- Print job
- Object

# Score

A *score* is a numerical value whose purpose is to offer you a high level view on the status of either a device or a user.

A score groups a coherent set of lower level analytics according to a particular concern and summarizes them into a single value.

Related tasks

- Computing scores

Related concepts

- Device
- User

# Service

A *service* represents an IT service in your organization, such as the mail service or the directory service. Nexthink lets you measure the quality of your IT services as it is perceived by the end-users.

You are however not limited to monitor well-known IT services like mail. Rather, with Nexthink you can define the services that you want to monitor by specifying the resources that they need to operate. These resources characterize and identify each service. In this way, you can monitor any service that matches your own definitions. Because you see the connectivity of end-users, how they actually use the service and who are impacted when the service is malfunctioning, the distinctive user-centered approach of Nexthink provides an advantage over other server-centered solutions.

Services in Nexthink are divided into *connection-based services* and *web-based services*:

Connection-based services
: Monitor connections at the transport level (TCP). Connection-based services are simply known as services.
: Applies to platforms:

Web-based services

Monitor web requests (HTTP/TLS) and responses (HTTP) at the application level, letting you drill-down to their underlying connections as well.
Applies to platforms:

Note that the monitoring of web-based services is only available for Windows devices.

## Connection-based services

Any IT service that requires TCP networking for its operation is suitable to be modelled as a service in Nexthink. With Nexthink services, you can supervise the state of your deployed IT services at a glance. If you wish to examine the data in depth, you can drill down through a service and get detailed information about:

- Network traffic associated to the Service.
- Load supported by each server.
- Connectivity of client computers to the Service.
- Crashes of applications related to the Service.
- Users impacted in case of Service failure.
- Performance of the Service in general.

An IT service can be characterized by the resources that are required to access it: the client applications that may be needed to access the service, the network ports that may be reserved to connect to the service or the servers in an organization that may be dedicated to provide the service. In Nexthink, you define a Service precisely by combining one or more of these resources:

- Device
- Executable
- TCP Port
- Destination

Nexthink associates every connection or connection attempt that matches the definition of a Service to it.

## Web-based services

In addition to devices, executables, ports and destinations, web-based services are also characterized by domains.

Web based services detect request errors at the application (HTTP) level.

Related tasks

- Analyzing service quality
- Creating a service

Related references

- Service errors and warnings

Related concepts

- Device
- Executable
- Port
- Destination
- Connection
- Domain

## System boot

A *system boot* is an activity of a device indicating that the device has been switched on.

Related concepts

- Activity

Related references

- Boot and logon duration

## User

A *user* is an object that represents an individual account in a device (local user) or in a group of devices (domain user). The account may identify a physical user or a system user. Physical users, that is, persons who work in front of a device, are also called *end-users*.

Depending on the context, a user may also refer to an operator of the Finder or the Portal. In these cases, context information is usually enough to differentiate

between Nexthink users and end-users.

### Users across platforms

A user may have access to more than one kind of device. Those devices may, in turn, belong to different platforms. For a user that has access to multiple devices, possibly belonging to different platforms, Nexthink can detect that there is only one single user.

To be able to detect that a user accessing multiple devices is actually the same person, the accounts of the user must be unified in an Active Directory. For instance, to detect that a Mac user is the same as a Windows or Mobile user, the Mac device of the user must join the Active Directory.

Applies to platforms:

Related concepts

- Device
- Object

## User logon

A *user logon* is an activity that takes place when a user authenticates in a device to open a session and gain control over it.

Related concepts

- User

Related references

- Boot and logon duration

## Web request

A *web request* is a message sent from a client application to a server using the standard web protocol HTTP or its secure version over TLS.

Nexthink is able to record not only the web requests of web browsers, but of any application that runs on the device of the end-user.

Related concepts

- Activity
- Connection

## Widget

A *widget* is a self-contained visual and logical software component to build dashboards in the Portal. Widgets display the results computed for metrics.

Related tasks

- Creating a dashboard

Related concepts

- Module
- Dashboard
- Metric

Related references

- Types of widgets

# Search and information display

## Search in Finder

### Overview

The Finder divides the results of a search in the Start page into two columns:

1. The left-hand side column, entitled **Investigations**, shows both existing investigations that match the search terms and automatically generated investigations that the system infers from the search terms and are suggested to the user. Because of the automatic inference, this part is also known as the *smart search*. The display of results is as follows:
    - An icon that indicates the type of object or activity on which the investigation is based.
    - A label **Suggested**, if the investigation was automatically generated.
    - The name of the investigation.
    - The time frame that restrains the results to a particular interval of time.
2. The right-hand side column shows search results based on the name of objects (i.e. **Devices**, **Executables**, etc), **Services**, **Metrics**, **Scores**, **Remote actions**, and **Categories**.

Applies to platforms:

### Suggested investigations

The Finder will use the typed words to suggest investigations. It will lookup if the words match:

- An object type (e.g. *device*) or an activity type (e.g. *connection*)
- The name of a platform if you want to filter the results depending on the kind of devices (e.g. *windows*).
- A keyword (e.g. *crash*, *performance*).
- A condition on an object type.
- Names of objects.
- Names of services.
- Names of entities.
- The name of a category (e.g. *NXT - Server type*) or one of its keywords (e.g. *Proxy*).

- A timeframe

In order to iteratively reduce the scope of the search, we recommend that you type the words following the previous order. After the first typed word, the Finder will provide you with search results that you can refine when typing more words. But this is not mandatory, as the Finder does not take words order into account.

When the Cross-Engine search features are enabled in the Finder, the suggested investigations additionally look for words matching the following items in all Engines, subject to the domain view of the Finder user:

- All users and devices.
- Domains seen in the last 5 days.
- Any other object seen in the last 7 days.

***Objects, activities and platforms***

Find below the list of objects and activities that you can use:

| Objects | Activities | Platforms |
|---|---|---|
| • users<br>• devices<br>• packages<br>• applications<br>• executables<br>• binaries<br>• ports<br>• destinations<br>• domains<br>• printers | • installations<br>• executions<br>• connections<br>• web requests<br>• print jobs<br>• system boots<br>• user logons | • windows<br>• mac<br>• mobile |

For example, search for packages.

| Search | Finder suggestions |
|---|---|
| packages | All packages - full period |

When the Cross-Engine search features are enabled in the Finder, the search tool looks for objects across all Engines and for all other shared items such as metrics, categories, services or remote actions. Displayed users and devices are limited to the domain view of the user that launched the search; while other objects and items may be outside the domain view of the user. In the latter case,

the user cannot investigate further the details of the object.

***Keywords***

As an example, you can look for errors and warnings in devices or applications using keywords. For instance, type *errors* in the Search box to get a list of any kind of error. You get the same results if you use synonyms of *error* such as *issue*, *problem* or *failure*.

If you want to be more specific in the kind of errors that you want to know about, you can use any of the following (or a valid synonym):

- system crash
- application crash
- application freeze (not responding)
- high cpu
- high memory

For example, to look for application crashes, just type in *application crash*:

| Search | Finder suggestions |
|---|---|
| application crash | Application crashes - today |

***A condition on an object type***

For example, you can type the name of an existing user and the Finder will show you suggested investigations that use the condition on the user name.

| Search | Finder suggestions |
|---|---|
| user *UserName* | Devices used by user *UserName* - full period |

***Names of objects***

Type in names of objects in your queries to look for a concrete instance of an object. As a Finder user, you may need to have the right privilege level to see the names of some objects (see step 3 of defining the profile of a user). Otherwise, they appear as anonymized in the search tool and you will be unable to search them by name.

As an example, type in the *name* of a device or a user in the Search box. You do not need to type in a full name. The Search fills the list of suggestions with investigations related to the objects with that *name* inside their properties. The Finder highlights the *name* in the list of results.

If the Finder detects that many objects match the *name*, it may infer that the word that you typed in is in fact a fragment of the actual name. In this case, the suggested investigations relate to groups of objects whose properties match the fragment. This is indicated by displaying the asterisk ***** wild card surrounding the *name*.

When you type names in the Search box, you can get a mix of suggested investigations that either match one object exactly or match a group of objects. For each investigation, the Finder may interpret the word as a full name or as a fragment. For example:

| Search | Finder suggestions |
|---|---|
| *nxtc* | Application matching *nxtc*fg.exe - full period<br><br>Applications used to access domain **nxtc** |

***Names of services***

Similarly to names of objects, look for names of services in the Search box to get investigations related to a particular service. For instance, if you have a service called *Mail Service*, start typing *mail* and you will get the following results (among others):

| Search | Finder suggestions |
|---|---|
| *mail* | Applications used for **Mail Service** - today<br><br>Devices using **Mail Service** - today ... |

***Names of entities***

If you have defined a set of entities for building up your hierarchies, type in the names of your entities in the Search box for the Finder to suggest investigations related to objects in those entities.

***Suggested investigations based on categories***

Use the **names of categories** to refine suggested investigations. For instance, given a category *RAM* that classifies devices according to the quantity of memory installed, the result of looking for devices with that category is the following:

| Search | Finder suggestion |
|---|---|
| device *RAM* | Devices with *RAM* - full period |

Where the name of the category is highlighted in the list of results and preceded by the label icon that identifies it as a category (not shown in the table).

Instead of the name of a category, you can directly use **the name of the keywords of the category**. For instance, let us assume that the keywords of the category RAM are:

- 2GB
- 3GB
- 4GB

You can directly look for devices using one of these keywords, or even combine several keywords, by typing:

| Search | Finder suggestion |
|---|---|
| device *2GB* | Devices with *RAM* set to *2GB* - full period |
| device *3GB 4GB* | Devices with *RAM* set to *3GB* or *4GB* - full period |

Alternatively, you can directly use the name of a category without specifying the type of object and optionally combine it with one of its keywords. In this case, the Finder deduces the type of object to which the category applies:

| Search | Finder suggestion |
|---|---|
| *RAM 1GB* | Devices with *RAM* set to *1GB* - full period |

***Timeframe control***

Limit the suggestions of the Finder to a particular time interval by specifying a timeframe. Find below the words that you can use to define a timeframe for the suggested investigations:

- Full period: The full time interval stored in the database of the Engine.
- Today: The current day (from 0 hours to the current time).
- Yesterday: The full day before today.
- Last hour: The last 60 minutes (including the current minute)
- Last week: The last seven days (including today).

***Platform control for suggestions***

If you use one of the platform names in your search, suggestions are adapted to match the available information for that platform. For instance, if you use the keyword **mobile** within a search for devices, the Finder suggests investigations about the access state, access rules and security policy of mobile devices.

Note that platform control in the smart search is only activated if devices of platforms other than Windows are detected inside your installation. If you only have Windows devices, the platform keywords (windows, mac os and mobile) are not recognized as such, but just as normal terms of your search.

### *Synonyms*

To make its use more natural, the Search tool of the Finder has the ability to recognize the singular and plural forms of these words as well as some of their synonyms. In many cases you can use your own words to look for information in the Finder and still get the expected results. For instance, instead of looking for *devices*, you can search *computers*, *PCs* or *workstations*.

Once you get used to Nexthink terminology, however, you may find more practical, accurate or even easier to stick to the official terms to designate objects or activities.

### *Using quotes*

When searching, you can use quotes to:

- Force the search on words with less than two letters. Normally, words with less than two letters are ignored by the Finder.

- Force the search to ignore spaces between words and consider the words together. For example, you can search for application with name that contains spaces. Let's say you search for *name of my application* (i.e. a name with spaces):

| Search | Finder suggestion |
|---|---|
| Application *"name of my application"* | Application matching *name of my application*- full period |

- Avoid name clashes with reserved words. The quotes instruct the Finder that the content inside is the value of an object name and not the name of a type of an object or activity. For instance, you get different results when you type the word *user* in the Search box with quotes and without quotes:

| Search | Finder first suggestion |
|---|---|
| user | User logons - today |
| *"user"* | Devices with package *user* - full period |

## User's investigation

The Finder will search if the user's investigation contains all the words and if one of the words is the name of an object or an activity type. If this is the case, we will also check if a word matches the object of the conditions.

For example, let's say that the user have a saved investigation named *InvestigationABC* based on devices:

| Search | Finder suggestion |
|---|---|
| device *InvestigationABC* | *InvestigationABC* |

***Time frame control***

By default, the original timeframe is used. But it can be modified, using the "timeframe control" described for suggested investigations. It will apply if the underlying investigation is compatible with it.

| Search | Finder suggestion |
|---|---|
| device *InvestigationABC* today | *InvestigationABC* - today |

***Platform control for investigations***

Using platform keywords in the search makes the Finder suggest only those user investigations that are suitable for all the enumerated platforms.

***Using synonyms and quotes***

The use of "synonyms" and "quote" described above for suggested investigations is the same for user's investigations.

***Show in investigations list***

If you want to modify the user's investigation, you can do a right-click and select the option "show in investigations list". Then you can modify the original investigation with a right-click and selecting "edit".

## Objects search

Up to now, we have discussed the results that the Search tool displays in the left column of the Start page under the title **Investigations**. This section covers the results of the Search tool that are displayed in the right column of the Start page.

The main use of the right column is to look for a single existing object in the database when you know its name, or at least part of it. In this case, the Finder does not have to deduce anything. It just performs a pure search by matching the terms that you type in with the names of objects or investigations in the database. Results are organized by type of object.

Using quotes will work in the same way as on the left panel. To increase the number of results, you can use wildcards:

*****
: To substitute for zero or more characters

**?**
: To substitute for zero or one character

The Finder runs the right and left panel search in parallel, so you do not have to choose between either one of them. Using wildcards, however, is not yet supported by the investigation search, which is likely to show no suggestions at all if you type in an asterisk or a question mark in your search.

Related tasks

- Adding users

Related concepts

- Object
- Activity
- Category
- Entity
- Service

# Keyboard shortcuts for column display selection

To efficiently choose the columns to display in investigations and alerts or the additional fields to display in metrics, learn the following keyboard shortcuts. In many cases, using the keyboard to select a display field may be faster and more convenient than using the mouse.

The **Ctrl** key enables the access to the keyboard shortcuts of the display field selector, which is shown when:

- Editing the options of an investigation.
- Navigating the results of an investigation.
- Creating a metric.

To quickly select fields with the keyboard:

1. Start typing in the box to filter the available fields by text content.
2. Press **Ctrl** to highlight a field in the selection dialog.
3. Press **Ctrl+Arrow key** to navigate away from the highlighted field:
    - ♦ Press **Ctrl+Right arrow** or **Ctrl+Down arrow** to move to the next available field.
    - ♦ Press **Ctrl+Left arrow** or **Ctrl+Up arrow** to move to the previous available field.
4. Use the **Home** and **End** keys to navigate to the beginning and to the end of the dialog:
    - ♦ Press **Ctrl+Home** to move to the first available field in the dialog.
    - ♦ Press **Ctrl+End** to move to last available field in the dialog.
5. Press **Ctrl+Space** to add the highlighted field to the display list and move to the next available field.
    - ♦ Alternatively, press **Ctrl+Shift+Space** to add the highlighted field to the display list and move to the previous available field.
    - ♦ While holding the **Ctrl** key, the text filter that you typed at the beginning of the procedure is applied.
        - ◊ After adding one or more fields to the display list, releasing the **Ctrl** key removes the text filter.
        - ◊ If you added no field to the display list, the text filter is maintained.

Note that disabled or already selected fields are not available for selection.

Related tasks

- Editing the options of an investigation
- Navigating the results of an investigation
- Creating a metric

# Campaign display compatibility

## Overview

Learn here about the different capabilities of the Collector to display campaigns depending on the version of the Collector and the platform, Windows or Mac, of the device on which the Collector runs.

## Feature compatibility

Both Windows and Mac Collector support all the main campaign features:

- One-off, recurring and continuous campaigns.
- Single answer, multiple answer and opinion scale questions.
- Workflows.
- Internationalization and translations.
    - The following languages are supported in campaign notifications starting from V6.22 only: Dutch, Norwegian, Danish, Swedish, Finnish, Afrikaans, Hindi, Japanese, Korean, and Vietnamese?. See here the full list of supported languages.
- Links and placeholders in campaign descriptions.

In addition, the Windows Collector supports the branding of campaigns. Because the Mac Collector uses the standard notification mechanism of macOS to display campaigns, the looks of a campaign are fixed. Therefore, the Mac Collector is still compatible with all campaigns but it ignores the specified brand logo and colors of the company, if any.

Starting from V6.21, the following feature is available in both Windows and Mac Collectors:

- Skip campaign notification.

Configuring the notification of all campaigns is still required for backwards compatibility, as Collectors previous to V6.21 ignore the setting to skip the notification.

## Protocol compatibility

On V6.12, the Nexthink Engage module introduced a more efficient communication protocol through the TCP channel between the Engine appliances and the Collector to get feedback from the end-users. Hence, Collectors previous to V6.12 are not able to display campaigns created with Nexthink V6.12 or later. For more information, see a previous version of this same article.

Related tasks

- Creating a campaign
- Branding of campaigns

# Real-time and consolidated service data

## How service data is computed

Nexthink starts to accumulate information about a service only after you have created it; thus, the service is empty right after its creation. As time passes, Nexthink collects data related to the service and stores the aggregated results in intervals of 10 minutes. The system keeps up to a maximum of six aggregated values that correspond to the six last 10 minutes intervals. After one hour, all the six aggregated values are filled with some data. At this point, the system erases the oldest aggregate and reuses it for a new 10 minutes interval. By rotating the stored results in this way, the system consistently displays the evolution of the service during the last hour with a granularity of 10 minutes.

In a similar way, for longer time intervals, the system stores an hourly aggregated result each time that it crosses an hour boundary, A total of 24 hourly aggregated results per service are stored and rotated, giving a dynamic full day view of the service.

Finally, the system also stores the daily aggregated results of a service when a day boundary is crossed. For each service, there are up to seven daily results stored to build up a full week of service data; although these latter are visible from the Portal only, not from the Finder.

The labels of the two rows that display the metrics in the Finder change thus with time according to the mechanisms for aggregating results described above. Both rows start with the label **Last 10 minutes**. After ten minutes have passed, both labels successively display **Last 20 minutes**, **Last 30 minutes**, etc until an hour boundary is crossed. At this point, the short-term row still increases every 10 minutes, while the long-term row label displays **Last 1 hour**, and increases to **Last 2 hours**, **Last 3 hours**, etc after every hour. When all results are available for the last 24 hours, the short-term row label displays **Last 60 minutes** and the long-term row label displays **Last 24 hours**. The labels no longer change after that point.

Aggregated results per service

| Quantity | Interval |
|---|---|

| | |
|---|---|
| 6 results | 10 minutes |
| 24 results | 1 hour |
| 7 results | 1 day |

## Discrepancies between the real-time view and the consolidated view of services

In some situations, it is possible that you find small discrepancies between the real-time values of a service (the Service View in the Finder or a service widget in the Portal) and the values that you get when you drill-down from the real-time view or perform an equivalent investigation. For instance, it may happen that the Service View of the Finder reckons the number of devices using a service to be 10 devices, for a given service and a particular hour of the day; whereas if you drill-down from the Service View to obtain the detailed list of the devices involved, you may get a list with only 9 devices. These differences arise because of the two separate ways in which the Engine collects and organizes service-related events.

On one hand, for the consolidation of events in the long-term, the Engine processes events to extract their timing information. Since events represent end-user information, an event reaches the Engine some time after the Collector has generated it in the device of the end-user. Nevertheless, the Engine can precisely determine the moment in time at which an event really began. This value is the final timestamp of the event.

On the other hand, the real-time views of services require the Engine to react immediately. Thus, the Engine takes into account every new event that matches the definition of a service as soon as the Engine receives the event from a Collector. The Engine aggregates these events during intervals of 10 minutes, 1 hour and 1 day, according to the explantion given above. The Service View displays the values collected within these intervals.

Because of this difference in treatment, if the interval between the start time of a service-related event and the moment of its reception by the Engine crosses a 10 minutes interval boundary (Engine time), you may get discrepancies between the real-time view and the consolidated view of the event. Indeed, if an event is generated short before the end of a 10 minutes interval and the Engine receives it once the interval is over, the Engine properly consolidates the event according to its real start time, but it cannot aggregate the event to the appropriate 10 minutes interval, because that interval is already closed for aggregation. Therefore, the Engine has to report the event in the next 10 minutes interval.

Related tasks

- Analyzing service quality
- Observing service performance
- Following the evolution of a service

Related concepts

- Service

Related references

- Timestamping of events

# Service errors and warnings

Nexthink constantly analyzes the state of services and provides information with respect to potential errors or warnings.

Remember that only Windows devices support web-based services.

## Connection-based services

Applies to platforms:

| | Type | Description |
|---|---|---|
| **Failed connections** | Device-level error | A device is marked in error state if it fails to connect to the destination for 60 seconds. |
| **Application crashes** | Device-level error | A device is marked in error state if the binary used to connect to the service experiences an application crash. |
| **Network response time** | Entity-level warning | All active devices in an entity are marked in warning state if the average network response time for the entity is 3 times greater than the automatically computed baseline for the previous 7 days. |

## Web-based services

Applies to platforms:

| | Type | Description |
|---|---|---|

| | | |
|---|---|---|
| **Application crashes** | Device-level error | A device is marked in error state if the binary used to connect to the service experiences an application crash. |
| **Failed HTTP request (5xx)** | Entity-level error | All active devices in an entity are marked in error state if the total number of failed HTTP requests with status 5xx is 3 times greater than the automatically computed baseline for the previous 7 days. |
| **Failed HTTP request (4xx)** | Entity-level warning | All active devices in an entity are marked in warning state if the total number of failed HTTP requests with status 4xx is 3 times greater than the automatically computed baseline for the previous 7 days. |
| **Redirected HTTP request (3xx)** | Entity-level warning | All active devices in an entity are marked in warning state if the total number of redirected HTTP requests with status 3xx is 3 times greater than the automatically computed baseline for the previous 7 days. |
| **Web request duration** | Entity-level warning | All active devices in an entity are marked in warning state if the average web request duration for the entity is 3 times greater than the automatically computed baseline for the previous 7 days. |

## Computation of averages and detection of outliers

Metrics described in the table above as *Entity-level* errors and warnings are computed for a set of devices instead of individual devices. The goal is to reduce false positives on metrics which are subject to a high degree of variation. For instance, a device might experience a long network response time during a few connections, but this usually does not mean that the service is compromised for this device. By computing such metrics at the entity (or location) level, we can obtain a more accurate representation of the actual quality of service.

For every metric that is computed at the *Entity-level*, there are minimum limits defined for issuing warnings. These are absolute minimum values below which the service quality is guaranteed, even in the case of a baseline violation. For a given metric, if the baseline is very low because the service has been performing extremely well in the past, even in the case that the computed average for the period is 3 times higher than the baseline, a warning is not issued when the average does not exceed the minimum limit.

In addition, an algorithm is put in place to detect and eliminate outliers. If a limited number of devices cause the mean value to exceed the error or warning level, the algorithm removes them from the computation of the baseline. The maximum number of devices that the algorithm can consider as outliers depend on the total number of devices in the Entity:

- 10% of the devices, if the total number of devices in the Entity is less than

100.
- 10 devices, if the total number of devices in the Entity is greater than or equal to 100.

Related concepts

- Service

# Errors and warnings for devices and executions

## Overview

Find below the list of errors and warnings on devices and executions that you can use in your investigations. To the right of each error or warning condition, find the platforms to which the condition applies.

| Subject | Warnings | Errors |
|---|---|---|
| **Device** | • High overall CPU usage<br>• High thread CPU usage (deprecated)<br>• High IO usage<br>• High memory usage<br>• High page faults | • System crash<br>• Hard reset<br>• SMART disk error |
| **Execution** | • High thread CPU usage<br>• High memory usage | • Application not responding<br>• Crash |

The Device view of the Finder also highlights this kind of events for a particular device in the **Errors** and **Warnings** timelines.

Find the definition of each one of these events in the articles that describe their corresponding tooltips. The Finder displays these tooltips when you hover your mouse over a particular occurrence of the event in the timelines of the Device View or the User View.

Note however that the **High CPU usage** warning has been renamed to **High thread CPU usage**, when applied to executions, and to **High thread CPU usage (deprecated)**, when applied to devices. Indeed, on devices, the warning has been deprecated in favor of the new **High overall CPU usage** warning. See the

explanation in the section below.

## Reported application crashes

Applications can provide their own crash handlers, effectively masking their failures from Nexthink.

In the case of the Mac platform, Nexthink reports the crashes of all applications that use the standard logs in macOS to record their crash events. If an application reports a crash by the standard means, macOS shows up a warning. All applications crashes that trigger the warning are reported correctly by Nexthink. For instance, if Chrome crashes, the macOS user sees the following warning:

## Device warnings on CPU usage

There are two warnings generated by the Engine to indicate a high CPU condition on a device:

- **High thread CPU usage (deprecated)**
- **High overall CPU usage**

The first warning, **High thread CPU usage (deprecated)**, is triggered when the CPU load is above 80% in a single logical processor (hardware *thread*) of a device for more than 30 seconds. This threshold has demonstrated to be somewhat low for the high capacity of modern CPUs with multiple cores. For instance, a quad-core device with hyper-threading technology has two hardware threads per core, making a total of 8 logical processors (threads) and a capacity of 800%. Thus, a load of 80% represents only the 10% of the total capacity of the CPU, which is certainly not that significant.

The second warning, **High overall CPU usage**, takes into account the total capacity of the CPU in a device. It is triggered when the CPU load is above 70% over all the logical processors combined for more than 30 seconds. That is, the threshold is 70% over a normalized CPU capacity of 100%. For the same quad-core device as in the example above, that would mean a 560% load over 800% of total capacity, which is indeed a high CPU load.

## Aggregates on CPU usage warnings

For the previous warning events, two aggregates let you know the percentage of time that devices are under a high CPU condition. There is also a similar aggregate for applications:

| Name | Applies to | Description |
|---|---|---|
| High device thread CPU time ratio (deprecated) | • device | Aggregates the duration of the warning **High thread CPU usage (deprecated)** and divides by the uptime of the device. |
| High device overall CPU time ratio | • device | Aggregates the duration of the warning **High overall CPU usage** and divides by the uptime of the device. |
| High application thread CPU time ratio | • application<br>• executable<br>• binary | Time that the execution of the application was in high CPU divided by the total execution time. |

Typically, the aggregate **High device thread CPU time ratio (deprecated)** displays higher ratios of time spent by devices under a high CPU condition. Those ratios may be unrealistically high for devices with multiple-core CPUs for the same reasons as explained above for its associated warning. Preferably use the aggregate **High device overall CPU time ratio**.

The aggregate **High application thread CPU time ratio** lets you find applications with high CPU consumption or, when applied to binaries, compare the different versions of a same application and see which one stays longer in a high CPU condition.

Related tasks

- Changing the thresholds of High CPU warnings

Related concepts

- Event

Related references

- Warnings tooltips
- Errors tooltips
- Memory and CPU usage

# Types of widgets

## Overview

Learn here about the different types of widgets that you can use on your dashboards for visually displaying data related to the computation of metrics. Use these types of widgets on dashboards within Basic modules, where you can choose the visualizations of your metrics individually.

You find similar widgets on dashboards within Service Monitoring modules, but the types of widgets and the layout are fixed on these dashboards. On their side, Software Metering modules display only one kind of widget on their dashboards, which is completely different from the widgets on the dashboards of both Basic and Service Monitoring modules.

In this article, we focus on the types of widgets for Basic modules, whose purpose is to display the values of metrics.

## KPI

A *KPI* (Key Performance Indicator) widget is mainly a single numerical figure that reflects the inner value of the represented metric (or metrics).

The figure in a KPI widget is either an absolute or a relative number (a percentage). It can be a percentage only if the definition of the associated metric includes a ratio computation; that is, a comparison between the value of the metric itself and the value returned by an additionally supplied investigation (usually a less constrained version of the investigation that computes the value of the metric).

In addition to the main figure, a KPI widget may also include a secondary figure that displays the variation in the value of the metric with respect to its previous value; where the meaning of *previous* depends on the time frame selected in the dashboard. Moreover, when specified in the definition the associated metric, a KPI widget indicates whether the increase or a decrease in the value of the metric is good or bad by coloring the arrow next to the variation figure in green or in red, respectively. If you decided that a variation in the value of the metric is not necessarily good or bad, the arrow is painted blue.

If the definition of a metric includes thresholds that limit the ranges of values that can be considered normal, worrying (optional), or bad, you can configure the KPI widget to display threshold information in the form of a colored dot that precedes

the main KPI figure. The dot is green if the value lies within the normal range, yellow if in the worrying range (which exists only if you define two thresholds), and red if in the bad range.

The following example shows a KPI widget related to a metric that counts the number of devices with CPU issues. The definition of the associated metric includes a ratio (devices with CPU issues compared to the total number of devices) which is the main figure in the widget. An increase in the value of the metric is obviously bad (it implies more devices with CPU issues), so the increase is displayed with a red arrow. Two thresholds are defined: when up to 10% of the total number of devices have CPU issues, the situation is considered normal; from 10% to 20% the situation becomes worrying; more than 20% devices with CPU issues is considered bad.

A single KPI widget can display the values of more than one metric. Arrange the individual visualizations of each metric vertically or horizontally within the KPI widget.

The KPI widget is compatible with count and quantity metrics. Top metrics are not suitable for being displayed as a single figure.

## Table

A *table* widget arranges the value of a metric (or metrics) in a grid. Add a table widget to your dashboards to display either a top metric or a group of count and quantity metrics.

### *Displaying a top metric*

When displaying a top metric in a table widget, the table shows the list of top objects as rows, while the columns are the display fields chosen in the configuration of the top metric. You can limit the number of fields displayed in the configuration of the table widget: you are allowed to turn off only those which are not essential to the definition of the metric.

A single table widget may display at most one top metric, which takes the whole of the widget. Therefore, it is neither possible to combine a top metric with another top metric nor with any other kind of metric in the same table widget.

### *Count and quantity metrics*

The display of count and quantity metrics in a table widget is very flexible. Add up to 50 metrics of these types to a single table widget. For each metric, choose to display either the value of the metric itself or a computed ratio value (when specified in the definition of the metric). If the metric defines thresholds, choose among displaying the status (a green, yellow, or red dot) next to the value, the status alone, or not to display threshold information at all and display only the value. You can also choose to display the variation of the metric with respect to its previous value. These choices are intendedly very similar to those that you can find in the KPI widget.

Arrange the values into the rows and columns of the table according to the hierarchy, the metric names, or the grouping criteria of the metrics that you have added to the widget (if specified in the definition of the metrics). Beware that, depending on the particular metrics that you choose, not all combinations are allowed.

## Line chart

A *line chart* graphically displays the historical evolution of the value of a metric (or metrics) with time. Line charts let you visually find out significant events in the history of metrics, compare values of different metrics along time and discover trends.

Add up to 5 metrics to a single line chart. All the metrics in the same line chart must be expressed in the same units, where the word *unit* must be understood in a broad sense. Note, for instance, that you can add count metrics of different objects, such as devices and binaries, to the same line chart, because the units are compatible (they are always a number of objects). You can even mix count metrics with those quantity metrics that measure a number of events. However, you cannot mix a metric that counts devices with a metric that measures the average boot time of devices, since a number of devices is not compatible with time units.

Make each line in the chart represent either the value of a metric or a computed ratio (when specified in the definition of the metric). When a metric defines thresholds, optionally display them in the line chart as horizontal lines. If you defined just one threshold, the chart displays a horizontal red line. When two thresholds are defined, the chart displays both a yellow and a red horizontal lines at the level specified by the thresholds. Points of the line above the thresholds are painted with the corresponding color. Even when you decide to show the thresholds, they may not be visible as horizontal lines in the chart if all the values of the metric are under the specified limits. Besides, displaying thresholds is only available when you display just one metric in the chart, since having thresholds in the same line chart for more than one metric would be too confusing.

Set the scale of the line chart to be either:

- **Automatic**, for the Portal to adapt the range of the vertical axis in the line chart to the values of the represented metric.
- **Fixed**, to specify the minimum and maximum values representable in the line chart. Useful if you know in advance the range of values of the represented metric (e.g. a metric based on a score). In addition, fixed scales favour the comparison of line charts in dashboards. Specify:
    - ♦ **from**, the minimum value for the vertical axis in the line chart.
    - ♦ **to**, the maximum value for the vertical axis in the line chart.

Line charts do not span to the time frame selected in a dashboard, but further to the past. When you select a time frame for a dashboard (day, week, month, or quarter), each point in the line of a line chart represents the value of the metric aggregated for that period. The last point in the line, corresponds to the selected date in the dashboard, while the values to the left are past values. How far the line chart goes into the past depends on the selected time frame and on the amount of available historical data. The minimum span gradually grows into the maximum span as the Portal computes more and more data:

| Time frame | Line chart min span | Line chart max span |
|---|---|---|
| Day | 30 days (1 month) | 60 days (2 months) |
| Week | 12 weeks (3 months) | 52 weeks (1 year) |
| Month | 12 months (1 year) | 24 months (2 years) |
| Quarter | 8 quarters (2 years) | 16 quarters (4 years) |

The line chart widget is compatible with count and quantity metrics, but it is not adapted to display top metrics.

## Bar chart

A *bar chart* graphically displays the values of count or quantity metrics in horizontal bars, letting you compare results visually with just a glimpse of the eye.

Depending on the metrics displayed on a bar chart, the bars that represent the measured values may be arranged differently:

- If the bar chart displays just one metric, arrange the bars either:
  - By the grouping criterion selected in the definition of the count or quantity metric.
  - By hierarchy. If you chose two grouping criteria (or none at all) when creating the metric, this is the only method available for arranging the bars.
- If the bar chart displays more than one metric, bars are arranged necessarily by metric.

Sort the results in the bar chart either by value or by name (the names of the metrics, the nodes in the hierarchy or the labels of the *group by* option). To see the most important values first, it is usually recommended to sort the bar chart by

value in descending order as best practice.

If the bar chart holds a count metric, choose between displaying its value or one of the two following ratios:

**Ratio defined in the metric**

Because the ratio defined in the metric compares arbitrary groups (those determined by the conditions in both the metric and ratio definitions), the value of each bar is independently calculated and the sum of all the bars in the widget does not add up to 100% in general. This option is only valid if the count metric actually defines a ratio.

**Ratio of total**

In bar charts, you are seldom interested in displaying the ratio defined in the metric (if any), but rather the distribution of each grouping option with respect to the total number of objects seen in the widget. The bars in a widget that displays the ratio of total do add up to 100% and the related count metric does not need to define a ratio for this option to be valid.

If the bar chart displays a metric that defines thresholds, tick the box **Threshold** for the chart to paint the bars in a specific color depending on the value of each bar crossing the threshold or not: green, yellow (only for metrics with two thresholds), or red. If the metric does not define any threshold or you do not tick the **Threshold** box, bars are depicted in blue. If multiple metrics are displayed in the same bar chart, each one will be colored independently.

Choose the minimum number of bars that you wish to see in the chart. More bars may be visible if space in the dashboard allows it. Otherwise, when more bars than those visible are available, a small down arrow appears at the bottom right corner of the chart. Hover the mouse cursor over the bar chart and the small arrow turns into a slider. Use the slider to scroll through all the values in the bar chart. If you leave the bar chart after scrolling, small arrows appear again to indicate the availability of more results either to the top or to the bottom of the chart, or both, when applicable.

The width of the bar chart is automatically scaled to accommodate enough space for the highest value to display. For bar charts with only one metric, it is possible to fix the scale to a maximum value that is meaningful for you. in the configuration of the bar chart, choose between **Automatic** or **Fixed**. In the latter case, specify the maximum value that the chart should show. If this maximum fixed value is exceeded by a bar in the chart, the widget is automatically redimensioned to fit the new maximum.

## Title widget

A *title widget* lets you group several other widgets under the same title. The title widget does not hold metrics by itself, but acts as an umbrella for other widgets that do display metric results.

Title widgets are particularly useful for organizing widgets of different types into logical sets, offering you more flexibility in the design of layouts for your dashboards.

## Compatibility matrix

Not every type of widget is suitable for displaying any kind of metric. In the table below, find the compatibility matrix among the types of metrics and the widgets to display them.

| Metric Type | KPI | Table | Line chart | Bar chart |
|---|---|---|---|---|
| **Count metric** | Ok | Ok | Same units | Same units |
| **Quantity metric** | Ok | Ok | Same units | Same units |
| **Top metric** | - | Only one | - | - |

Same units
: If you add more than one metric to the chart, the metrics must share the same units.

Only one
: The widget accepts at most one metric.

Related tasks

- Creating a metric

# Widget compute state in charts

## Overview

For line chart widgets, the state of the computation of the associated metric for a particular date is directly available from the chart. When the metric is successfully computed, the Portal displays a solid line in the line chart.

When the computation of the metric is consistently successful for several days, the solid line joins the points that represent the value of the metric along the whole selected period. Hovering the mouse cursor over the chart displays, the value of the computation on the selected date:

If a metric defines threshold values, make them visible in the line chart by ticking the option **Thresholds** when configuring the widget. The lower threshold is depicted as a horizontal orange line and the higher threshold as a horizontal red line. Likewise, value points in the chart are highlighted in orange when they exceed the first threshold value or in red when they exceed the second threshold value:

However, the Portal is not always able to compute the metric of a widget on a particular date. When this happens, the chart of the widget replaces the solid blue line by a dashed line or by empty space to indicate that the computation could not be carried out. Hovering over a non computed area of a chart with the mouse displays a message that helps you identify the cause for the lack of data.

We list below the reasons why there might be a lack of data in the chart of a widget and we illustrate them with example figures from the Portal. We finally discuss the

## Line charts

### *First computation of the widget*

Before the first computation of a widget, there is no data available for it. Therefore, the chart does not display any line before the first computation of the widget.

If you hover the mouse over the empty area, the chart displays the date, the name of the metric and the message **no data computed**:

### *Not computed*

A widget can miss some days of computation because the Portal or the Engine were stopped or because the connection between the Portal and the Engine was lost for some time.

If the computation was not run on one or more days, the chart displays a dashed straight red line between the two dates where there is actually some data. Additionally, when you hover the mouse over it, the chart displays the message **no data computed**.

### *No activity*

The widget was not computed because there was not data available for it. The connection with the Engine is fine in this case, but the Engine just does not provide proper data for the widget.

This occurs with activity widgets and service widgets when the type of activity that they monitor did not take place over the requested period of time and providing a zero value does not make sense. For example, a widget that computes the average response time of an application will show no activity if the application was not executed over the specified period. Showing zero response time for the application is not a valid option in this case.

This situation can also happen with inventory and issue widgets that refer to objects grouped by a specific keyword. If you change the group by property of the widget and you do not recompute it for the past dates, the data for those dates will be lost showing no activity.

The chart displays a dashed blue line at the level of the axis and the message **no activity** when hovering with the mouse.

## Other widgets

KPI, table, and bar char widgets represent the absence of computed metric data by a dash sign ( **-** ) in the place of the value.

## All widgets

### *One day only*

Because of an undefined aggregation strategy, the computation of some metrics does not make sense for periods longer than one day. When selecting a period longer than one day in the Portal, the widgets related to these metrics show the following message:

### *Configuration issues*

If a widget has a configuration error, e.g. because the associated metric has been deleted, the Portal displays the following message in the middle of the dashboard area assigned to the widget:

Related references

- Types of widgets
- Widget

# Errors in the execution of remote actions

The execution of a remote action can fail because of different reasons. Find below the list of possible errors that a failed execution of a remote action can return.

The Finder displays the error message in the field **Execution status details**:

Timeout
: **The script timed out and was terminated before completion**.

Execution of remote actions is disabled
: **The remote action could not be executed: the script execution policy is set to disabled**.

Invalid signature or certificate
: **The remote action could not be executed: the script signature is invalid or the certificate is not trusted**.

User context
: **The remote action could not be executed in the context of the user: no user is logged in, there is more than one user logged in (and none are active) or this is a server operating system**.

Self-help scenario
: **Engage license not enabled or trying to run on a server**

# Top results of Cross-Engine investigations

## Overview

Investigations that return a specified number of top objects, which are ordered according to a particular criterion, may yield surprising results when targeting multiple Engines simultaneously.

Learn how these top investigations are executed in Cross-Engine contexts to avoid misunderstandings.

## Individual execution of top investigations

When targeting multiple Engines, a top investigation executes first on each Engine individually and then aggregates the results. For instance, suppose that you are looking for the top 4 domains ordered by the highest number of visiting devices across two Engines.

| Engine 1 | | Engine 2 | |
|---|---|---|---|
| **Domain** | **Number of Devices** | **Domain** | **Number of Devices** |
| community.nexthink.com | 300 | community.nexthink.com | 350 |
| doc.nexthink.com | 200 | www.nexthink.com | 150 |
| www.nexthink.com | 150 | doc.nexthink.com | 50 |
| design.nexthink.com | 50 | design.nexthink.com | 40 |

The Cross-Engine investigation returns the total number of devices by adding the results in both Engines.

| Domain | Number of Devices |
|---|---|
| community.nexthink.com | 650 |
| www.nexthink.com | 300 |
| doc.nexthink.com | 250 |
| design.nexthink.com | 90 |

## Aggregation of different top results

However, imagine that you repeat the same investigation, but you only ask for the top 2 domains with the highest number of visiting devices. In this case, the individual execution on each Engine returns a different list of domains:

| Engine 1 | | Engine 2 | |
|---|---|---|---|
| **Domain** | **Number of Devices** | **Domain** | **Number of Devices** |
| community.nexthink.com | 300 | community.nexthink.com | 350 |
| doc.nexthink.com | 200 | www.nexthink.com | 150 |

Results beyond the second domain are lost. Thus, the aggregation of results ignores anything after the second position and the Cross-Engine investigation returns the following:

| Domain | Number of Devices |
|---|---|
| community.nexthink.com | 650 |
| doc.nexthink.com | 200 |

While we might expect to find the domain `www.nexthink.com` in the second place with 300 devices, as in the previous top 4 investigation, we see instead that

`doc.nexthink.com` takes the second place with 200 devices because the aggregation is ignoring the values beyond the second place in both Engines. Keep in mind this behavior when writing Cross-Engine top investigations whose aggregates are added up.

Related tasks

- Editing the options of an investigation
- Enabling Cross-Engine Finder features

# Tooltips in the user and device views

## Alerts tooltips

### Single alert occurrence

The tooltip displays the name of the alert and its duration. The time range indicates the exact period when the alert was active.

### Multiple alert occurrences

When an alert is triggered multiple times within the selected period, this tooltip shows the start time of each one of the occurrences and their duration. The time range above shows the interval when the two alerts were active, with the granularity of the units of the timeline. Thus, the interval displayed depends on the zoom of the device view.

## Warnings tooltips

### High CPU usage

#### *High device CPU usage*

Applies to platforms:

Applies to platforms:

The tooltip displays the CPU usage on the device during a time interval, when the overall CPU usage is above 70%. The tooltip appears in two forms:

- When multiple applications are the cause of the high CPU load in the device and none is specially responsible for it, then the tooltip only gives a figure of the total CPU usage.
- When a few particular applications have a significant impact in the CPU load, the tooltip details these applications and their CPU usage per thread (this is different from the overall CPU usage, see explanation in the tooltip below), in addition to the overall CPU usage in the device.

The Collector takes a CPU load sample every 30 seconds and reports them every 5 minutes. To report high CPU usage, two or more consecutive samples must exceed 70% of CPU usage. The reported value corresponds to the average value of the samples above 70%.

***High application CPU usage***

Applies to platforms:

The tooltip displays a list of the processes (and their corresponding application) that consumed more than 50% of the CPU processing power per thread (40% for the system process) for at least 30 seconds during the selected time interval. Processes are ordered in the list by their contribution to the CPU load from the most to the least demanding.

A 100% usage corresponds to the capacity of a single-core single-threaded CPU. You can get more than 100% usage in CPUs with multi-core or hyper-threading technologies:

- For each additional physical core in the CPU you get 100% more processing power. That is, the capacity of a dual-core CPU is 200%, for a quad-core is 400%, for an hex-core is 600%, and so on.
- If the CPU uses hyper-threading technology, the processing power of each physical core is doubled. A hyper-threaded dual-core CPU yields thus 400% capacity, a quad-core 800%, a hex-core 1200%, and so forth.

Note that this way of measuring is different from that of high CPU on the device, where the overall capacity of a CPU is 100%, regardless of the number of cores and threads in the CPU. For this reason, the bottom of the tooltip displays the capacity of the CPU in terms of hardware threads, which can be (and it usually is) bigger than 100%.

## High memory

Applies to platforms:

The tooltip displays the usage of physical memory when more than 70% of the total physical memory is in use for at least 5 minutes.

In its lower part, the tooltip lists the top five processes that consume the most memory. Only processes that take more than 1% of the total memory are listed.

## High IO throughput

Applies to platforms:

Applies to platforms:

The warning shows up when IO operations exchange information at a rate higher than 20 MB/s during at least 30 seconds.

The total IO throughput includes all the bytes transferred during read, write or any other kind of IO operation performed on either real or virtual peripherals, such as hard disks, flash drives, network controllers, keyboards, mice, etc.

### High page faults

Applies to platforms:

Applies to platforms:

The warning appears when the device generates more than 5000 page faults per second in memory accesses during at least 5 minutes.

Related concepts

- Event

Related references

- Errors and warnings for devices and executions
- Errors tooltips
- Memory and CPU usage

## Errors tooltips

### Application crash

Applies to platforms:

The tooltip displays the name of the executable that crashed, along with its version and the name of the application to which it belongs. In the case of a single crash, the time in the header of the tooltip is the exact time when the crash was reported.

In the case of several almost simultaneous crashes grouped in the same tooltip, the time in the header of the tooltip displays the interval during which all crashes took place. Each application listed in the tooltip is preceded by its own precise time of crashing.

## Application not responding

Applies to platforms:

The tooltip for non responding applications displays the same information as the tooltip for application crashes, but for applications that hang or freeze instead of for applications that exit unexpectedly.

## System crash

Applies to platforms:

The tooltip displays the stop error code and a brief textual description of the error that caused the device to crash. These are serious hardware or software errors that make your computer halt unexpectedly. For that reason, the time in the header of the tooltip cannot indicate the moment when the error occurred. Instead, it is the time of the first boot after the system crash.

## Hard reset

Applies to platforms:

The tooltip indicates that the device was abruptly stopped and then rebooted. Pressing the reset button, power failures or crashes may be the cause of a hard reset. As in the case of the blue screen tooltip, the time reported here is the time of the first boot of the device after the hard reset.

### SMART disk error

Applies to platforms:

The tooltip signals a disk error detected in a disk drive equipped with the SMART technology. It indicates an increase in the number of disk writing errors or in the count of reallocated sectors.

Related concepts

- Event

Related references

- Errors and warnings for devices and executions
- Warnings tooltips

## Activity tooltips

### Executions

#### *Standard*

The tooltip shows the number of executions during the selected time interval. All the applications executed by user accounts were run at the user privilege level. Note that executions carried out by system accounts, which usually operate at the administrator privilege level, may also contribute to the count.

#### *Privilege warning*

The tooltip displays the number of executions during the selected time interval with an additional privilege warning. The warning indicates that at least one of the executions was carried out by a user account with power user or administrator privilege levels.

### Connections

The connections tooltip displays the overall amount of traffic measured during the selected time interval. This includes the TCP and UDP traffic that the device sent out and the TCP traffic that the device received.

### User interaction

The tooltip displays the name of the user that interacted with the device along with the total duration of the interaction (in parenthesis). The maximum duration is limited by the selected time interval, which is indicated in the header of the tooltip. User interaction is detected as mouse or keyboard activity of the user. The user is considered inactive if the device does not receive any mouse or keyboard event for 15 minutes.

The tooltip can only be displayed if the monitoring of user interaction is enabled.

### Mobile synchronization

At least one successful synchronization with the Exchange ActiveSync server has been detected within a one-hour window.

If **push notifications** are set-up, the device will **automatically synchronize** when a new event arrives on the server (email, calendar update, etc?). However, the user can also **manually synchronize** their device at any time, or disable push notifications and **schedule the synchronization** at a regular interval.

## Services tooltips

## Connection-based services

The tooltip displays summary information on the performance of the connection-based service during the selected time interval. If any of the collected statistics exceeds its normal range, the tooltip may display a warning message as well.

## Web-based services

The tooltip displays summary information on the performance of the web-based service during the selected time interval. If any of the collected statistics exceeds its normal range, the tooltip may display a warning message as well.

# Database information and organization

## Data model

This reference article contains the complete description of Nexthink's data model.

### Objects

Objects represent items recognized by Nexthink.

#### *User*

Users of devices (domain, local or system)

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Activity start time | Activity | Aggregate | | | |
| | Start time of investigated activity | | | | |
| | NXQL ID: | activity_start_time | | | |
| Activity stop time | Activity | Aggregate | | | |
| | Stop time of investigated activity | | | | |
| | NXQL ID: | activity_stop_time | | | |
| Application crash ratio | Errors | Aggregate | | | |
| | Indicates the number of application crashes per 100 executions. | | | | |
| | NXQL ID: | application_crash_ratio | | | |
| Application not responding event ratio | Errors | Aggregate | | | |
| | Indicates the number of application not responding events per 100 executions. | | | | |
| | NXQL ID: | application_not_responding_event_ratio | | | |
| Average incoming network bitrate | Availability | Aggregate | | | |
| | Average incoming network bitrate | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average incoming web bitrate | Availability | Aggregate | | | |
| | Average incoming bitrate of all underlying web requests, consolidated over time | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | average_incoming_bitrate | | | |
| Average memory usage per execution | Activity | Aggregate | | | |
| | Indicates the average memory usage of all underlying executions before aggregation. The value is the average<br><br>memory usage of all executions (calculated with a 5-minute resolution) multiplied by their cardinalities and divided by the total cardinality.<br><br>• Example: if two tabs of the Chrome browser are opened at the same time, two distinct processes of chrome.exe are launched and they are aggregated by the Engine (i.e., event cardinality = 2). The average memory usage will be the average of the two processes before aggregation: it represents the average memory usage of a Chrome tab. | | | | |
| | NXQL ID: | average_memory_usage_per_execution | | | |
| Average network response time | Availability | Aggregate | | | |
| | Indicates the average TCP connection establishment time of all underlying connections. The value is<br><br>the average TCP connection establishment time of all executions weighted by their cardinality. | | | | |
| | NXQL ID: | average_network_response_time | | | |
| Average outgoing network bitrate | Availability | Aggregate | | | |
| | Average outgoing network bitrate | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average outgoing web bitrate | Availability | Aggregate | | | |
| | Average outgoing bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average web request duration | Availability | Aggregate | | | |
| | Average time between request and last response byte | | | | |
| | NXQL ID: | average_request_duration | | | |
| Average web request size | Traffic | Aggregate | | | |
| | Average size of web requests | | | | |
| | NXQL ID: | average_request_size | | | |
| Average web response size | Traffic | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Average size of web responses | | | | |
| | NXQL ID: | average_response_size | | | |
| Binary paths | Activity | Aggregate | | | |
| | List of executed binary paths (max. 50 paths) | | | | |
| CPU usage ratio | Activity | Aggregate | | | |
| | Indicates the sum of the CPU time of all executions on each device in scope over all logical processors divided by their total duration.<br><br>Executions shorter than 30 seconds are ignored.<br><br>• Example: if we consider two executions with the first one taking 50% of a logical processor during 30 minutes and the second one taking 100% of 2 logical processors during 60 minutes, the CPU usage ratio is 150% (= [50% * 30 min + 2 * 100% * 60 min] / [30 min + 60 min]). | | | | |
| | NXQL ID: | cpu_usage_ratio | | | |
| Cumulated execution duration | Activity | Aggregate | | | |
| | Cumulated duration of executions | | | | |
| | NXQL ID: | cumulated_execution_duration | | | |
| Cumulated network connection duration | Activity | Aggregate | | | |
| | Cumulated duration of TCP connections | | | | |
| | NXQL ID: | cumulated_connection_duration | | | |
| Department | Properties | Field | | | |
| | User department as listed in Active Directory | | | | |
| | NXQL ID: | department | | | |
| Distinguished name | Properties | Field | | | |
| | Active Directory distinguished name (DN) | | | | |
| | NXQL ID: | distinguished_name | | | |
| First seen | Properties | Field | | | |
| | First time activity of the user was recorded on any device | | | | |
| | NXQL ID: | first_seen | | | |
| Full name | Properties | Field | | | |
| | Full user name as listed in Active Directory | | | | |
| | NXQL ID: | full_name | | | |
| Highest local privilege level reached | Activity | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Highest local privilege level reached for executions (user, power user, administrator) | | | | |
| | NXQL ID: | highest_local_privilege_reached | | | |
| Incoming network traffic | Traffic | Aggregate | | | |
| | Total network incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming web traffic | Traffic | Aggregate | | | |
| | Total web incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Job title | Properties | Field | | | |
| | Job title as listed in Active Directory | | | | |
| | NXQL ID: | job_title | | | |
| Last seen | Properties | Field | | | |
| | Last time activity of the user was recorded on any device | | | | |
| | NXQL ID: | last_seen | | | |
| Lowest observed web protocol version | Activity | Aggregate | | | |
| | Lowest protocol version observed in web requests (excluding web requests with unknown protocol version) | | | | |
| | NXQL ID: | lowest_protocol_version | | | |
| Name | Properties | Field | | | |
| | User logon name | | | | |
| | NXQL ID: | name | | | |
| Network availability level | Availability | Aggregate | | | |
| | Indicates the ratio of successful TCP connections. The possible values are:<br>• high: the ratio is greater or equal to 98%<br>• medium: the ratio is greater or equal to 90% and less than 98%<br>• low: the ratio is lower than 90% | | | | |
| | NXQL ID: | network_availability_level | | | |
| Number of application crashes | Errors | Aggregate | | | |
| | Number of application crashes | | | | |
| | NXQL ID: | number_of_application_crashes | | | |
| Number of application not responding events | Errors | Aggregate | | | |
| | Number of application not responding events | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | number_of_application_not_responding_events | | | |
| Number of applications | Inventory | Aggregate | | | |
| | Number of applications | | | | |
| | NXQL ID: | number_of_applications | | | |
| Number of binaries | Inventory | Aggregate | | | |
| | Number of binaries | | | | |
| | NXQL ID: | number_of_binaries | | | |
| Number of connections | Activity | Aggregate | | | |
| | Number of connections | | | | |
| | NXQL ID: | number_of_connections | | | |
| Number of days since last seen | Properties | Field | | | |
| | Indicates the number of days since the last time the user was seen by Nexthink. The field is updated every hour. | | | | |
| | NXQL ID: | number_of_days_since_last_seen | | | |
| Number of destinations | Inventory | Aggregate | | | |
| | Number of destinations | | | | |
| | NXQL ID: | number_of_destinations | | | |
| Number of devices | Inventory | Aggregate | | | |
| | Number of devices | | | | |
| | NXQL ID: | number_of_devices | | | |
| Number of domains | Inventory | Aggregate | | | |
| | Number of domains | | | | |
| | NXQL ID: | number_of_domains | | | |
| Number of executables | Inventory | Aggregate | | | |
| | Number of executables | | | | |
| | NXQL ID: | number_of_executables | | | |
| Number of executions | Activity | Aggregate | | | |
| | Number of executions | | | | |
| | NXQL ID: | number_of_executions | | | |
| Number of ports | Inventory | Aggregate | | | |
| | Number of ports | | | | |
| | NXQL ID: | number_of_ports | | | |
| Number of print jobs | Activity | Aggregate | | | |
| | Number of print jobs | | | | |
| | NXQL ID: | number_of_printouts | | | |

| | | | | | |
|---|---|---|---|---|---|
| Number of printed pages | Activity | Aggregate | | | |
| | Number of printed pages | | | | |
| | NXQL ID: | number_of_printed_pages | | | |
| Number of printers | Inventory | Aggregate | | | |
| | Number of printers | | | | |
| | NXQL ID: | number_of_printers | | | |
| Number of web requests | Activity | Aggregate | | | |
| | Number of web requests | | | | |
| | NXQL ID: | number_of_web_requests | | | |
| Outgoing network traffic | Traffic | Aggregate | | | |
| | Total network outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Outgoing web traffic | Traffic | Aggregate | | | |
| | Total web outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Protocols used in web requests | Activity | Aggregate | | | |
| | Protocols used in web requests (HTTP, TLS, HTTP/TLS) | | | | |
| | NXQL ID: | protocols_used_in_requests | | | |
| SID | Properties | Field | | | |
| | Indicates the Windows security identifier for the user.<br>• For Mac 0S: the value is 'S-0-0' if the user is not in Active Directory. | | | | |
| | NXQL ID: | sid | | | |
| Successful HTTP requests ratio | Availability | Aggregate | | | |
| | Percentage of successful HTTP requests (1xx, 2xx and 3xx) | | | | |
| | NXQL ID: | successful_http_requests_ratio | | | |
| Successful network connections ratio | Availability | Aggregate | | | |
| | Percentage of successful TCP connections | | | | |
| | NXQL ID: | successful_connections_ratio | | | |
| Total active days | Activity | Field | | | |
| | Total number of days the user was active | | | | |
| | NXQL ID: | total_active_days | | | |
| Total CPU time | Activity | Aggregate | | | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Indicates the sum of the CPU time of all executions on each device in scope and over all logical processors.<br><br>Executions shorter than 30 seconds are ignored.<br><br>• Example: if we consider two executions with the first one taking 50% of a logical processor during 30 minutes and the second one taking 100% of 2 logical processors during 60 minutes, the total CPU time is 135 minutes (= 50% * 30 min + 2 * 100% * 60 min). | | | | |
| | NXQL ID: | total_cpu_time | | | |
| Total network traffic | Traffic | Aggregate | | | |
| | Total network traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_network_traffic | | | |
| Total web traffic | Traffic | Aggregate | | | |
| | Total web traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_web_traffic | | | |
| Type | Properties | Field | | | |
| | Type of user (local/domain/system) | | | | |
| | NXQL ID: | type | | | |
| UID | Properties | Field | | | |
| | Indicates the universally unique identifier (based on user SID). | | | | |
| | NXQL ID: | user_uid | | | |
| Web interaction time | Activity | Aggregate | | | |
| | Indicates the time during which at least one executable is doing HTTP or TLS traffic. This is counted with a 5-minute resolution. | | | | |
| | NXQL ID: | cumulated_web_interaction_duration | | | |

### *Device*

Devices are Windows, Mac OS or mobile endpoints

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Access state | Exchange | Field | | | |
| | Indicates whether the device can access the Exchange ActiveSync server. The possible states are: | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | • allowed: the device has access<br>• blocked: the device is blocked<br>• discovery: the device is temporarily quarantined while it is being identified by the Exchange ActiveSync server<br>• quarantined: the device is waiting for Exchange ActiveSync administrator approval | | | | |
| | NXQL ID: | eas_access_state | | | |
| Access state reason | Exchange | Field | | | |
| | Indicates the reason for the device access state. The possible values are:<br>• global: caused by the global access settings<br>• device rule: caused by a device access rule<br>• individual: caused by an individual exemption<br>• policy: caused by Exchange ActiveSync policy | | | | |
| | NXQL ID: | eas_access_state_reason | | | |
| Activity start time | Activity | Aggregate | | | |
| | Start time of investigated activity | | | | |
| | NXQL ID: | activity_start_time | | | |
| Activity stop time | Activity | Aggregate | | | |
| | Stop time of investigated activity | | | | |
| | NXQL ID: | activity_stop_time | | | |
| AD site | Properties | Field | | | |
| | Indicates the AD site of the device as configured in the Active Directory.<br>A '-' is displayed if the Collector is older than version 6.19 or if the device is not part of a domain. | | | | |
| | NXQL ID: | directory_service_site | | | |
| Administrator account status | Policy | Field | | | |
| | Determines whether the local Administrator account is enabled or disabled | | | | |
| | NXQL ID: | administrator_account_status | | | |
| All antispyware | Security | Field | | | |
| | Summary information about all the detected antispyware:<br>• unknown: indicates that the information could not be retrieved<br>• N/A: this field is not available on this operating system<br>• '-' : no data, incompatible collector version or the data is not yet available | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | all_antispywares<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| All antiviruses | Security | Field | | | |
| | Summary information about all the detected antiviruses:<br><br>• unknown: indicates that the information could not be retrieved<br>• N/A: this field is not available on this operating system<br>• '-' : no data, incompatible collector version or the data is not yet available | | | | |
| | NXQL ID: | all_antiviruses<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| All firewalls | Security | Field | | | |
| | Summary information about all the detected firewalls:<br><br>• unknown: indicates that the information could not be retrieved<br>• N/A: this field is not available on this operating system<br>• '-' : no data, incompatible collector version or the data is not yet available | | | | |
| | NXQL ID: | all_firewalls<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| Antispyware display name | Security | Field | | | |
| | Name of the main antispyware | | | | |
| | NXQL ID: | antispyware_name<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| Antispyware RTP | Security | Field | | | |
| | Indicates whether the antispyware real time protection (RTP) is active:<br><br>• on: indicates that RTP is active<br>• off: indicates that either RTP is not active or no antispyware has been detected | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | • unknown: indicates that the information could not be retrieved<br>• N/A: this field is not available on this operating system<br>• '-' : no data, incompatible collector version or the data is not yet available | | | | |
| | NXQL ID: | antispyware_rtp<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| Antispyware up-to-date | Security | Field | | | |
| | Indicates whether the antispyware is up-to-date:<br><br>• yes: indicates that antispyware is up-to-date<br>• no: indicates that either the antispyware is not up-to-date or no antispyware has been detected<br>• unknown: indicates that the information could not be retrieved<br>• N/A: this field is not available on this operating system<br>• '-' : no data, incompatible collector version or the data is not yet available | | | | |
| | NXQL ID: | antispyware_up_to_date<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| Antivirus display name | Security | Field | | | |
| | Name of the main antivirus | | | | |
| | NXQL ID: | antivirus_name<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| Antivirus RTP | Security | Field | | | |
| | Indicates whether the antivirus real time protection (RTP) is active:<br><br>• on: indicates that RTP is active<br>• off: indicates that either RTP is not active or no antivirus has been detected<br>• unknown: indicates that the information could not be retrieved<br>• N/A: this field is not available on this operating system<br>• '-' : no data, incompatible collector version or the data is | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | not yet available | | | | |
| | NXQL ID: | antivirus_rtp<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| Antivirus up-to-date | Security | Field | | | |
| | Indicates whether the antivirus is up-to-date:<br><br>• yes: indicates that antivirus is up-to-date<br>• no: indicates that either the antivirus is not up-to-date or no antivirus has been detected<br>• unknown: indicates that the information could not be retrieved<br>• N/A: this field is not available on this operating system<br>• '-' : no data, incompatible collector version or the data is not yet available | | | | |
| | NXQL ID: | antivirus_up_to_date<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| Application crash ratio | Errors | Aggregate | | | |
| | Indicates the number of application crashes per 100 executions. | | | | |
| | NXQL ID: | application_crash_ratio | | | |
| Application not responding event ratio | Errors | Aggregate | | | |
| | Indicates the number of application not responding events per 100 executions. | | | | |
| | NXQL ID: | application_not_responding_event_ratio | | | |
| Audit account logon events | Policy | Field | | | |
| | Determines whether to audit each instance of a user logging on to or logging off from another computer in which this computer is used to validate the account | | | | |
| | NXQL ID: | audit_account_logon_events | | | |
| Audit account management | Policy | Field | | | |
| | Determines whether to audit each event of account management on a computer | | | | |
| | NXQL ID: | audit_account_management | | | |
| Audit directory service access | Policy | Field | | | |
| | Determines whether to audit the event of a user accessing an Active Directory object that has its own system access control list (SACL) specified | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | audit_directory_service_access | | | |
| Audit logon events | Policy | Field | | | |
| | Determines whether to audit each instance of a user logging on to or logging off from a computer | | | | |
| | NXQL ID: | audit_logon_events | | | |
| Audit object access | Policy | Field | | | |
| | Determines whether to audit the event of a user accessing an object, e.g. a file, folder, registry key, printer, and so forth-that has its own system access control list (SACL) specified | | | | |
| | NXQL ID: | audit_object_access | | | |
| Audit policy change | Policy | Field | | | |
| | Determines whether to audit every incident of a change to user rights assignment policies, audit policies, or trust policies | | | | |
| | NXQL ID: | audit_policy_change | | | |
| Audit privilege use | Policy | Field | | | |
| | Determines whether to audit each instance of a user exercising a user right | | | | |
| | NXQL ID: | audit_privilege_use | | | |
| Audit process tracking | Policy | Field | | | |
| | Determines whether to audit detailed tracking information for events such as program activation, process exit, handle duplication, and indirect object access | | | | |
| | NXQL ID: | audit_process_tracking | | | |
| Audit system events | Policy | Field | | | |
| | Determines whether to audit when a user restarts or shuts down the computer or when an event occurs that affects either the system security or the security log | | | | |
| | NXQL ID: | audit_system_events | | | |
| Average extended logon duration | Startup | Aggregate | | | |
| | Indicates the average extended logon duration. | | | | |
| | NXQL ID: | average_extended_logon_duration | | | |
| Average incoming network bitrate | Availability | Aggregate | | | |
| | Average incoming network bitrate | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average incoming web bitrate | Availability | Aggregate | | | |
| | Average incoming bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |

| | | | | | |
|---|---|---|---|---|---|
| Average logon duration | Startup | Aggregate | | | |
| | Indicates the average logon duration. | | | | |
| | NXQL ID: | average_logon_duration | | | |
| Average memory usage per execution | Activity | Aggregate | | | |
| | Indicates the average memory usage of all underlying executions before aggregation. The value is the average<br><br>memory usage of all executions (calculated with a 5-minute resolution) multiplied by their cardinalities and divided by the total cardinality.<br><br>• Example: if two tabs of the Chrome browser are opened at the same time, two distinct processes of chrome.exe are launched and they are aggregated by the Engine (i.e., event cardinality = 2). The average memory usage will be the average of the two processes before aggregation: it represents the average memory usage of a Chrome tab. | | | | |
| | NXQL ID: | average_memory_usage_per_execution | | | |
| Average network response time | Availability | Aggregate | | | |
| | Indicates the average TCP connection establishment time of all underlying connections. The value is<br><br>the average TCP connection establishment time of all executions weighted by their cardinality. | | | | |
| | NXQL ID: | average_network_response_time | | | |
| Average outgoing network bitrate | Availability | Aggregate | | | |
| | Average outgoing network bitrate | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average outgoing web bitrate | Availability | Aggregate | | | |
| | Average outgoing bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average system boot duration | Startup | Aggregate | | | |
| | Indicates the average system boot duration. | | | | |
| | NXQL ID: | average_boot_duration | | | |
| Average web request duration | Availability | Aggregate | | | |
| | Average time between request and last response byte | | | | |
| | NXQL ID: | average_request_duration | | | |

| | | | | | |
|---|---|---|---|---|---|
| Average web request size | Traffic | Aggregate | | | |
| | Average size of web requests | | | | |
| | NXQL ID: | average_request_size | | | |
| Average web response size | Traffic | Aggregate | | | |
| | Average size of web responses | | | | |
| | NXQL ID: | average_response_size | | | |
| Binary paths | Activity | Aggregate | | | |
| | List of executed binary paths (max. 50 paths) | | | | |
| BIOS serial number | Hardware | Field | | | |
| | BIOS serial number | | | | |
| | NXQL ID: | bios_serial_number | | | |
| Chassis serial number | Hardware | Field | | | |
| | Chassis serial number | | | | |
| | NXQL ID: | chassis_serial_number | | | |
| Collector assignment | Nexthink Collector | Field | | | |
| | Indicates whether Collector assignment service is enabled or disabled<br>• disabled: indicates that the Collector feature is disabled<br>• enabled: indicates that the Collector feature is enabled<br>"-" : data not available | | | | |
| | NXQL ID: | cltr_ca_status | | | |
| Collector assignment license UID | Nexthink Collector | Field | | | |
| | Indicates the Collector assignment license UID | | | | |
| | NXQL ID: | cltr_ca_license_uid | | | |
| Collector CrashGuard count | Nexthink Collector | Field | | | |
| | Indicates the number of consecutive hard resets or system crashes of the device | | | | |
| | NXQL ID: | cltr_crash_guard_count | | | |
| Collector CrashGuard limit | Nexthink Collector | Field | | | |
| | Indicates the Collector CrashGuard limit | | | | |
| | NXQL ID: | cltr_crash_guard_limit | | | |

| | | | | | |
|---|---|---|---|---|---|
| Collector CrashGuard protection interval | Nexthink Collector | Field | | | |
| | Indicates the CrashGuard monitoring interval in minutes | | | | |
| | NXQL ID: | cltr_crash_guard_protection_interval | | | |
| Collector CrashGuard reactivation interval | Nexthink Collector | Field | | | |
| | Indicates the Collector CrashGuard reactivation interval in hours | | | | |
| | NXQL ID: | cltr_crash_guard_react_interval | | | |
| Collector installation log | Nexthink Collector | Field | | | |
| | Indicates the link to the last Nexthink Collector installation error log. | | | | |
| | NXQL ID: | collector_installation_log | | | |
| Collector log level | Nexthink Collector | Field | | | |
| | Indicates the Collector log level<br>• off: no logs<br>• critical: only critical logs are enabled<br>• error: only error and critical logs are enabled<br>• warning: only warning, error and critical logs are enabled<br>• info: only info, warning, error and critical logs are enabled<br>• debug: only debug, info, warning, error and critical logs are enabled<br>• trace: only trace, debug, info, warning, error and critical logs are enabled<br>"-" : data not available | | | | |
| | NXQL ID: | cltr_log_level | | | |
| Collector status | Nexthink Collector | Field | | | |
| | Indicates the status of the Nexthink Collector package installed on the device:<br>• unmanaged: the Collector is not automatically updated<br>• up-to-date: the Collector is up-to-date | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | • outdated: a newer Collector version is available. | | | | |
| | NXQL ID: | collector_status | | | |
| Collector tag | Nexthink Collector | Field | | | |
| | Indicates the Collector installation tag. | | | | |
| | NXQL ID: | collector_tag | | | |
| Collector update group | Nexthink Collector | Field | | | |
| | Indicates the update group of Nexthink Collector:<br>• manual: the Collector is manually updated<br>• pilot: the Collector is updated as part of the pilot group<br>• main: the Collector is updated as part of the main group. | | | | |
| | NXQL ID: | upgrade_group | | | |
| Collector update status | Nexthink Collector | Field | | | |
| | Indicates the status of the Nexthink Collector updater. | | | | |
| | NXQL ID: | collector_update_status | | | |
| Collector version | Nexthink Collector | Field | | | |
| | Indicates the version of the Nexthink Collector installed on the device. | | | | |
| | NXQL ID: | collector_version | | | |
| CPU frequency | Hardware | Field | | | |
| | CPU frequency | | | | |
| | NXQL ID: | cpu_frequency | | | |
| CPU model | Hardware | Field | | | |
| | CPU model | | | | |
| | NXQL ID: | cpu_model | | | |
| CPU usage ratio | Activity | Aggregate | | | |
| | Indicates the sum of the CPU time of all executions on each device in scope over all logical processors divided by their total duration.<br>Executions shorter than 30 seconds are ignored.<br>• Example: if we consider two executions with the first one taking 50% of a logical processor during 30 minutes and the second one taking 100% of 2 logical processors during 60 minutes, the CPU usage ratio is 150% (= [50% | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | * 30 min + 2 * 100% * 60 min] / [30 min + 60 min]). | | | | |
| | NXQL ID: | cpu_usage_ratio | | | |
| Cumulated execution duration | Activity | Aggregate | | | |
| | Cumulated duration of executions | | | | |
| | NXQL ID: | cumulated_execution_duration | | | |
| Cumulated interaction time | Activity | Aggregate | | | |
| | Cumulated time with user interaction (mouse or keyboard events) | | | | |
| | NXQL ID: | cumulated_interaction_duration | | | |
| Cumulated network connection duration | Activity | Aggregate | | | |
| | Cumulated duration of TCP connections | | | | |
| | NXQL ID: | cumulated_connection_duration | | | |
| Data transport protocol | Nexthink Collector | Field | | | |
| | Specifies if the Collector data is sent over TCP or UDP<br>• UDP: the Collector data traffic is sent over UDP<br>• TCP: the Collector data traffic is sent over TCP<br>"-" : data not available | | | | |
| | NXQL ID: | cltr_data_channel_protocol | | | |
| Database usage | Properties | Field | | | |
| | Indicates the percentage of the Engine database used by the device. | | | | |
| | NXQL ID: | database_usage | | | |
| Device access rule | Exchange | Field | | | |
| | Indicates the name of the Exchange ActiveSync device access rule and if the rule allows, blocks or quarantines the device. | | | | |
| | NXQL ID: | eas_device_access_rule | | | |
| Device encryption required | Policy | Field | | | |
| | Indicates whether device encryption is required. | | | | |
| | NXQL ID: | device_encryption_required | | | |
| Device identity | Exchange | Field | | | |
| | Indicates the identity of the device in Exchange ActiveSync server. | | | | |
| | NXQL ID: | eas_device_identity | | | |
| Device manufacturer | Hardware | Field | | | |
| | Indicates the device manufacturer. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | device_manufacturer | | | |
| Device model | Hardware | Field | | | |
| | Indicates the model of the device. | | | | |
| | NXQL ID: | device_model | | | |
| Device password required | Policy | Field | | | |
| | Indicates whether a password is required on the device. | | | | |
| | NXQL ID: | device_password_required | | | |
| Device product ID | Hardware | Field | | | |
| | Device product ID | | | | |
| | NXQL ID: | device_product_id | | | |
| Device product version | Hardware | Field | | | |
| | Device product version | | | | |
| | NXQL ID: | device_product_version | | | |
| Device serial number | Hardware | Field | | | |
| | Indicates the device serial number. | | | | |
| | NXQL ID: | device_serial_number | | | |
| Device type | Hardware | Field | | | |
| | Indicates the device type:<br>• desktop<br>• laptop<br>• server<br>• mobile | | | | |
| | NXQL ID: | device_type | | | |
| Device UUID | Properties | Field | | | |
| | Indicates the device universally unique identifier (UUID). | | | | |
| | NXQL ID: | device_uuid | | | |
| Disks S.M.A.R.T. index | Hardware | Field | | | |
| | Lowest S.M.A.R.T. index of installed hard disks (index is based on S.M.A.R.T. attributes) | | | | |
| | NXQL ID: | disks_smart_index | | | |
| Distinguished name | Properties | Field | | | |
| | Indicates the distinguished name (DN) as seen:<br>• For Windows: in Active Directory (AD); if no connection with AD is set up, a '-' is displayed<br>• For Mobile: in the Exchange ActiveSync server | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | distinguished_name | | | |
| Distinguished name reported by Collector | Properties | Field | | | |
| | Indicates the distinguished name as reported by the Collector.<br><br>A '-' is displayed if the device is not part of a domain. | | | | |
| | NXQL ID: | collector_distinguished_name | | | |
| Email attachment enabled | Policy | Field | | | |
| | Indicates whether attachments can be downloaded to the mobile device through the Exchange ActiveSync protocol. | | | | |
| | NXQL ID: | email_attachment_enabled | | | |
| Enforce password history | Policy | Field | | | |
| | Indicates the number of unique password that have to be associated with a user account before an old password can be reused:<br><br>• Windows: as set up in the group policy<br>• Mobile: as set up in security policies | | | | |
| | NXQL ID: | enforce_password_history | | | |
| Engage | Nexthink Collector | Field | | | |
| | Indicates whether Engage is enabled or disabled<br><br>• enabled: indicates that the status of Engage service in Collector is enabled<br><br>• enabled except on server OS: indicates that the status of Engage service in Collector is enabled on all devices except on servers<br><br>• disabled: indicates that the status of Engage service in Collector is disabled<br><br>"-" : data not available | | | | |
| | NXQL ID: | cltr_engage_service_status | | | |
| Entity | Properties | Field | | | |
| | Entity to which the device belongs | | | | |
| | NXQL ID: | entity | | | |
| Exemption | Exchange | Field | | | |
| | Indicates whether a personal exemption is set for the device and its user. Possible values are: | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | • none<br>• allow<br>• block | | | | |
| | NXQL ID: | eas_exemption | | | |
| Extended logon duration baseline | Startup | Field | | | |
| | Indicates the extended logon duration averaged over the last logons. In the calculation, recent logons weigh more than older logons (exponentially<br><br>weighted moving average). | | | | |
| | NXQL ID: | extended_logon_duration_baseline | | | |
| Firewall display name | Security | Field | | | |
| | Name of the main firewall | | | | |
| | NXQL ID: | firewall_name<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| Firewall RTP | Security | Field | | | |
| | Indicates whether the firewall real time protection (RTP) is active:<br><br>• on: indicates that RTP is active<br>• off: indicates that either RTP is not active or no firewall has been detected<br>• unknown: indicates that the information could not be retrieved<br>• N/A: this field is not available on this operating system<br>• '-' : no data, incompatible collector version or the data is not yet available | | | | |
| | NXQL ID: | firewall_rtp<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| First seen | Properties | Field | | | |
| | Indicates the first time when the activity of the device was recorded:<br><br>• For Windows and Mac OS: the first time Collector reported activity<br>• For Mobile: the first time the device was reported with a successful synchronization | | | | |
| | NXQL ID: | first_seen | | | |

| | | | | | |
|---|---|---|---|---|---|
| Graphical card RAM | Hardware | Field | | | |
| | Amount of RAM of the graphical card with most RAM | | | | |
| | NXQL ID: | graphical_card_ram | | | |
| Graphical cards | Hardware | Field | | | |
| | Installed graphical cards | | | | |
| | NXQL ID: | graphical_cards | | | |
| Group name | Network | Field | | | |
| | Name of computer domain or workgroup | | | | |
| | NXQL ID: | group_name | | | |
| Guest account status | Policy | Field | | | |
| | Determines if the Guest account is enabled or disabled | | | | |
| | NXQL ID: | guest_account_status | | | |
| Hard disks | Hardware | Field | | | |
| | List of all hard disks | | | | |
| | NXQL ID: | hard_disks | | | |
| Hard disks manufacturers | Hardware | Field | | | |
| | Indicates the list of hard disk manufacturers | | | | |
| | NXQL ID: | disks_manufacturers | | | |
| High device IO throughput time ratio | Warnings | Aggregate | | | |
| | Indicates the ratio between the time the device is in high IO throughput and its uptime. | | | | |
| | NXQL ID: | high_device_io_throughput_time_ratio | | | |
| High device memory time ratio | Warnings | Aggregate | | | |
| | Indicates the ratio between the time the device is in high memory usage and its uptime. | | | | |
| | NXQL ID: | high_device_memory_time_ratio | | | |
| High device overall CPU time ratio | Warnings | Aggregate | | | |
| | Indicates the ratio between the time the device is in high overall CPU usage and its uptime. | | | | |
| | NXQL ID: | high_device_overall_cpu_time_ratio | | | |
| High device page faults time ratio | Warnings | Aggregate | | | |
| | Indicates the ratio between the time the device is in high page faults and its uptime. | | | | |
| | NXQL ID: | high_device_page_faults_time_ratio | | | |
| High device thread CPU time | Warnings | Aggregate | | | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| ratio (deprecated) | Indicates the ratio between the time that the device is in high thread CPU usage and its uptime. | | | | |
| Highest local privilege level reached | Activity | Aggregate | | | |
| | Highest local privilege level reached for executions (user, power user, administrator) | | | | |
| | NXQL ID: | highest_local_privilege_reached | | | |
| Incoming network traffic | Traffic | Aggregate | | | |
| | Total network incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming web traffic | Traffic | Aggregate | | | |
| | Total web incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Interaction time ratio | Activity | Aggregate | | | |
| | Percentage of time with user interaction (mouse or keyboard events) | | | | |
| Internet security settings | Security | Field | | | |
| | Internet security settings (ok, at risk or unknown) | | | | |
| | NXQL ID: | internet_security_settings | | | |
| IP addresses | Network | Field | | | |
| | List of IP addresses for the device | | | | |
| | NXQL ID: | ip_addresses | | | |
| IP protocol DNS resolution | Nexthink Collector | Field | | | |
| | Indicates the DNS resolution preference for Collector in terms of IP protocol version on the device<br>• IPv4: prefer IPv4<br>• IPv6: prefer IPv6<br>"-" : data not available | | | | |
| | NXQL ID: | cltr_dns_res_preference | | | |
| Last extended logon duration | Startup | Field | | | |
| | Indicates the last recorded value for the time between the user logging on and the device is ready. | | | | |
| | NXQL ID: | last_extended_logon_duration | | | |
| Last IP address | Network | Field | | | |
| | Last IP address assigned to the device | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | last_ip_address | | | |
| Last known connection status | Nexthink Collector | Field | | | |
| | Indicates the last known connection status of the device:<br>• 'UDP only': the device is successfully connected via UDP.<br>• 'TCP only': the device is successfully connected via TCP.<br>• 'Fully connected': the device is successfully connected via both UDP and TCP, unless it has been configured to transmit only over TCP, in which case it has successfully connected via TCP. | | | | |
| | NXQL ID: | last_known_connection_status | | | |
| Last logged on user | Startup | Field | | | |
| | Last logged on user | | | | |
| | NXQL ID: | last_logged_on_user | | | |
| Last logged on user's privileges | Startup | Field | | | |
| | Privileges of the last logged on user (user, power user, administrator) | | | | |
| | NXQL ID: | privileges_of_last_logged_on_users | | | |
| Last logon duration | Startup | Field | | | |
| | Indicates the last recorded value for the time between the user logging on and the desktop is displayed. | | | | |
| | NXQL ID: | last_logon_duration | | | |
| Last logon time | Startup | Field | | | |
| | Indicates the time of the last logon. | | | | |
| | NXQL ID: | last_logon_time | | | |
| Last policy update | Exchange | Field | | | |
| | Indicates the last time the Exchange ActiveSync policy was updated on the device. | | | | |
| | NXQL ID: | eas_policy_update | | | |
| Last seen | Properties | Field | | | |
| | Indicates the last time that activity on the device was reported:<br>• For Windows and Mac OS: the last time that the Collector reported activity.<br>• For Mobile: the last time that the device successfully synchronized with the Mobile Bridge. | | | | |
| | NXQL ID: | last_seen | | | |
| | | Field | | | |

| | | | | | |
|---|---|---|---|---|---|
| Last seen on TCP | Nexthink Collector | | | | |
| | Indicates the last time that the device was successfully connected through the TCP channel.<br>• '-': the Collector version is older than V6.6 and does not support TCP; or the Collector has never connected to Engine in TCP. | | | | |
| | NXQL ID: | last_seen_on_tcp | | | |
| Last system boot duration | Startup | Field | | | |
| | Duration of last system boot | | | | |
| | NXQL ID: | last_boot_duration | | | |
| Last system boot time | Startup | Field | | | |
| | Indicates the time of the last system boot. | | | | |
| | NXQL ID: | last_system_boot | | | |
| Last system update | Operating system | Field | | | |
| | Time of last system update | | | | |
| | NXQL ID: | last_windows_update | | | |
| Last update | Nexthink Collector | Field | | | |
| | Indicates the last Collector update time. | | | | |
| | NXQL ID: | last_update | | | |
| Last update status | Nexthink Collector | Field | | | |
| | Indicates the status of the last Collector update:<br>• '-': the Collector was never updated<br>• successful installation: the last Collector installation was successful<br>• package download error: the Collector was not able to download the Collector package from Nexthink Appliance<br>• package digital signature error: the Collector was not able to check the Collector package digital signature<br>• device reboot required: the device needs to be rebooted to complete the Collector installation<br>• package error: the Collector package installation has failed<br>• internal error: the Collector package installation has failed for an unexpected reason. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | last_update_status | | | |
| Last Updater request | Nexthink Collector | Field | | | |
| | Indicates the last time the Nexthink Updater has checked for updates. | | | | |
| | NXQL ID: | last_updater_request | | | |
| Local Administrators | Operating system | Field | | | |
| | Users and groups which are members of the Local Administrators group on the device | | | | |
| | NXQL ID: | local_administrators | | | |
| Local Power Users | Operating system | Field | | | |
| | Users and groups which are members of the Local Powers Users group on the device | | | | |
| | NXQL ID: | local_power_users | | | |
| Logical drives | Local drives | Field | | | |
| | List of all logical drives | | | | |
| | NXQL ID: | logical_drives | | | |
| Logon duration baseline | Startup | Field | | | |
| | Indicates the logon duration averaged over the last logons. In the calculation, recent logons weigh more than older logons (exponentially<br><br>weighted moving average). | | | | |
| | NXQL ID: | average_logon_duration | | | |
| Lowest observed web protocol version | Activity | Aggregate | | | |
| | Lowest protocol version observed in web requests (excluding web requests with unknown protocol version) | | | | |
| | NXQL ID: | lowest_protocol_version | | | |
| MAC addresses | Network | Field | | | |
| | List of MAC addresses for the device | | | | |
| | NXQL ID: | mac_addresses | | | |
| Maximum password age | Policy | Field | | | |
| | Indicates the period in time (in days) during which the password can be used before the system requires the user to change it:<br><br>• Windows: as set up in the group policy<br>• Mobile: as set up in security policies | | | | |
| | NXQL ID: | maximum_password_age | | | |

| | | | | | |
|---|---|---|---|---|---|
| Membership type | Network | Field | | | |
| | Type of computer membership (domain/workgroup) | | | | |
| | NXQL ID: | membership_type | | | |
| Message maximum segment size | Nexthink Collector | Field | | | |
| | Indicates the maximum segment size of packets sent by Collector | | | | |
| | NXQL ID: | cltr_max_segment_size | | | |
| Minimum password age | Policy | Field | | | |
| | Period of time (in days) that a password must be used before the user can change it | | | | |
| | NXQL ID: | minimum_password_age | | | |
| Minimum password length | Policy | Field | | | |
| | Least number of characters that a password for a user account may contain | | | | |
| | NXQL ID: | minimum_password_length | | | |
| Monitor models | Hardware | Field | | | |
| | Models of connected monitors | | | | |
| | NXQL ID: | monitor_models | | | |
| Monitor resolutions | Hardware | Field | | | |
| | Screen resolutions of connected monitors | | | | |
| | NXQL ID: | monitor_resolutions | | | |
| Monitoring of unresponsive applications | Nexthink Collector | Field | | | |
| | Indicates whether the Collector is monitoring for unresponsive applications on the device<br>• disabled: indicates that the Collector feature is disabled<br>• enabled: indicates that the Collector feature is enabled<br>"-" : data not available | | | | |
| | NXQL ID: | cltr_freezes_monitoring | | | |
| Monitors | Hardware | Field | | | |
| | Connected monitors | | | | |
| | NXQL ID: | monitors | | | |
| Monitors serial numbers | Hardware | Field | | | |
| | Serial numbers of connected monitors (ordered as in 'Monitors') | | | | |
| | NXQL ID: | monitors_serial_numbers | | | |

| | | | | | |
|---|---|---|---|---|---|
| Name | Properties | Field | | | |
| | Indicates the name of the device:<br>• For Windows: NetBios Name<br>• For Mac OS: computer name used on the network<br>• For Mobile: composed by mailbox name and device friendly name | | | | |
| | NXQL ID: | name | | | |
| Network availability level | Availability | Aggregate | | | |
| | Indicates the ratio of successful TCP connections. The possible values are:<br>• high: the ratio is greater or equal to 98%<br>• medium: the ratio is greater or equal to 90% and less than 98%<br>• low: the ratio is lower than 90% | | | | |
| | NXQL ID: | network_availability_level | | | |
| Number of antispyware | Security | Field | | | |
| | Number of antispyware detected:<br>• unknown: indicates that the information could not be retrieved<br>• N/A: this field is not available on this operating system<br>• '-' : no data, incompatible collector version or the data is not yet available | | | | |
| | NXQL ID: | number_of_antispyware<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |
| Number of antiviruses | Security | Field | | | |
| | Number of antiviruses detected:<br>• unknown: indicates that the information could not be retrieved<br>• N/A: this field is not available on this operating system<br>• '-' : no data, incompatible collector version or the data is not yet available | | | | |
| | NXQL ID: | number_of_antiviruses<br><br>**Note**: this field is not available for Windows Server operating systems. | | | |

| | | | | | |
|---|---|---|---|---|---|
| Number of application crashes | Errors | Aggregate | | | |
| | Number of application crashes | | | | |
| | NXQL ID: | number_of_application_crashes | | | |
| Number of application not responding events | Errors | Aggregate | | | |
| | Number of application not responding events | | | | |
| | NXQL ID: | number_of_application_not_responding_events | | | |
| Number of applications | Inventory | Aggregate | | | |
| | Number of applications | | | | |
| | NXQL ID: | number_of_applications | | | |
| Number of binaries | Inventory | Aggregate | | | |
| | Number of binaries | | | | |
| | NXQL ID: | number_of_binaries | | | |
| Number of connections | Activity | Aggregate | | | |
| | Number of connections | | | | |
| | NXQL ID: | number_of_connections | | | |
| Number of cores | Hardware | Field | | | |
| | Indicates the number of CPUs multiplied by the number of cores that are available on each CPU. | | | | |
| | NXQL ID: | number_of_cores | | | |
| Number of CPUs | Hardware | Field | | | |
| | Indicates the number of central processing units (CPUs), also known as the number of sockets. | | | | |
| | NXQL ID: | number_of_cpus | | | |
| Number of days since first seen | Properties | Field | | | |
| | Indicates the number of complete days since the device was first seen. The value is updated every hour. | | | | |
| | NXQL ID: | number_of_days_since_first_seen | | | |
| Number of days since last logon | Startup | Field | | | |
| | Number of days since last logon | | | | |
| | NXQL ID: | number_of_days_since_last_logon | | | |
| Number of days since last policy update | Exchange | Field | | | |
| | Indicates the number of days since the last Exchange ActiveSync policy update. | | | | |
| | NXQL ID: | number_of_days_since_last_eas_policy_update | | | |
| Number of days since last seen | Properties | Field | | | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Indicates the number of days since the last time that activity on the device was reported. The field is updated every hour:<br>• For Windows and Mac OS: the last time that the Collector reported activity.<br>• For Mobile: seen the last time that the device successfully synchronized with the Mobile Bridge. | | | | |
| | NXQL ID: | number_of_days_since_last_seen | | | |
| Number of days since last seen on TCP | Nexthink Collector | Field | | | |
| | Indicates the number of days since the last time that the device was successfully connected through the TCP channel. The field is updated every hour:<br>• '-': the Collector version is older than V6.6 and does not support TCP; or the Collector has never connected to Engine in TCP. | | | | |
| | NXQL ID: | number_of_days_since_last_seen_on_tcp | | | |
| Number of days since last system boot | Startup | Field | | | |
| | Number of days since last system boot | | | | |
| | NXQL ID: | number_of_days_since_last_boot | | | |
| Number of days since last system update | Operating system | Field | | | |
| | Number of days since last system update | | | | |
| | NXQL ID: | number_of_days_since_last_windows_update | | | |
| Number of destinations | Inventory | Aggregate | | | |
| | Number of destinations | | | | |
| | NXQL ID: | number_of_destinations | | | |
| Number of domains | Inventory | Aggregate | | | |
| | Number of domains | | | | |
| | NXQL ID: | number_of_domains | | | |
| Number of executables | Inventory | Aggregate | | | |
| | Number of executables | | | | |
| | NXQL ID: | number_of_executables | | | |
| Number of executions | Activity | Aggregate | | | |
| | Number of executions | | | | |
| | NXQL ID: | number_of_executions | | | |
| | Security | Field | | | |

| | | | | | |
|---|---|---|---|---|---|
| Number of firewalls | Number of firewalls detected:<br>• unknown: indicates that the information could not be retrieved<br>• N/A: this field is not available on this operating system<br>• '-' : no data, incompatible collector version or the data is not yet available | | | | |
| | NXQL ID: | number_of_firewalls<br>**Note**: this field is not available for Windows Server operating systems. | | | |
| Number of graphical cards | Hardware | Field | | | |
| | Number of installed graphical cards | | | | |
| | NXQL ID: | number_of_graphical_cards | | | |
| Number of hard resets | Errors | Aggregate | | | |
| | Number of hard resets | | | | |
| Number of installations | Activity | Aggregate | | | |
| | Number of installations | | | | |
| | NXQL ID: | number_of_installations | | | |
| Number of logical processors | Hardware | Field | | | |
| | Indicates the number of cores multiplied by the number of threads that can run on each core through the use of hyperthreading. | | | | |
| | NXQL ID: | logical_cpu_number | | | |
| Number of logons | Startup | Aggregate | | | |
| | Number of logons | | | | |
| | NXQL ID: | number_of_logons | | | |
| Number of monitors | Hardware | Field | | | |
| | Number of connected monitors | | | | |
| | NXQL ID: | number_of_monitors | | | |
| Number of packages | Inventory | Aggregate | | | |
| | Number of packages | | | | |
| | NXQL ID: | number_of_packages | | | |
| Number of ports | Inventory | Aggregate | | | |
| | Number of ports | | | | |
| | NXQL ID: | number_of_ports | | | |
| Number of print jobs | Activity | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Number of print jobs | | | | |
| | NXQL ID: | number_of_printouts | | | |
| Number of printed pages | Activity | Aggregate | | | |
| | Number of printed pages | | | | |
| | NXQL ID: | number_of_printed_pages | | | |
| Number of printers | Inventory | Aggregate | | | |
| | Number of printers | | | | |
| | NXQL ID: | number_of_printers | | | |
| Number of system boots | Startup | Aggregate | | | |
| | Number of system boots | | | | |
| | NXQL ID: | number_of_boots | | | |
| Number of system crashes | Errors | Aggregate | | | |
| | Indicates the number of Windows bluescreens. | | | | |
| Number of users | Inventory | Aggregate | | | |
| | Number of users | | | | |
| | NXQL ID: | number_of_users | | | |
| Number of web requests | Activity | Aggregate | | | |
| | Number of web requests | | | | |
| | NXQL ID: | number_of_web_requests | | | |
| OS architecture | Operating system | Field | | | |
| | Architecture of device operating system (x86/x64) | | | | |
| | NXQL ID: | os_architecture | | | |
| OS build | Operating system | Field | | | |
| | Indicates the build number of the operating system:<br>• '0.0.0.0': incompatible collector version or the data is not yet available | | | | |
| | NXQL ID: | os_build | | | |
| OS version | Operating system (deprecated) | Field | | | |
| | Version of device operating system | | | | |
| OS version and architecture | Operating system | Field | | | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Indicates name, version and architecture (when applicable) of the operating system:<br>• Unknown: the OS version could not be retrieved or it could not be mapped to a recognized value | | | | |
| | NXQL ID: | os_version_and_architecture | | | |
| Outgoing network traffic | Traffic | Aggregate | | | |
| | Total network outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Outgoing web traffic | Traffic | Aggregate | | | |
| | Total web outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Packages and updates scan interval | Nexthink Collector | Field | | | |
| | Indicates the interval, in hours, after which the Collector checks for newly installed packages and updates | | | | |
| | NXQL ID: | cltr_installs_scan_interval | | | |
| Password complexity requirements enabled | Policy | Field | | | |
| | Indicates whether password complexity is required:<br>• Windows: the password must meet complexity requirements as defined in the group policy<br>• Mobile: no simple passwords are allowed or a minimum password length is set, as defined in the security policy | | | | |
| | NXQL ID: | password_complexity_requirements | | | |
| Platform | Properties | Field | | | |
| | Indicates the platform of the device. A platform is a set of operating system families on which the same objects, activities, events and properties can be retrieved. The possible values are:<br>• Windows<br>• Mac OS<br>• Mobile | | | | |
| | NXQL ID: | platform | | | |
| Policy allows non provisionable devices | Exchange | Field | | | |
| | Indicates whether a device which does not fully support the policy is still allowed to connect to the Exchange Exchange ActiveSync server.<br>• If 'yes', the security policy is not guaranteed to be applied, | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | even if the field 'ActiveSync policy application status' value is 'applied in full' | | | | |
| | NXQL ID: | allow_non_provisionable_devices | | | |
| Policy application status | Exchange | Field | | | |
| | Indicates whether the Exchange ActiveSync policy is applied or not. Possible values are:<br>• not applied<br>• applied in full: the policy is applied (unless the field 'Allow non provisionable devices' value is 'yes')<br>• partially applied | | | | |
| | NXQL ID: | eas_policy_application_status | | | |
| Policy name | Exchange | Field | | | |
| | Indicates the name of the Exchange ActiveSync policy applied to the user's mailbox. | | | | |
| | NXQL ID: | eas_policy_name | | | |
| Print monitoring | Nexthink Collector | Field | | | |
| | Indicates whether the Collector printing monitoring is enabled or disabled<br>• disabled: indicates that the Collector feature is disabled<br>• enabled: indicates that the Collector feature is enabled<br>"-" : data not available | | | | |
| | NXQL ID: | collector_print_monitoring_status | | | |
| Protocols used in web requests | Activity | Aggregate | | | |
| | Protocols used in web requests (HTTP, TLS, HTTP/TLS) | | | | |
| | NXQL ID: | protocols_used_in_requests | | | |
| Script execution policy | Nexthink Collector | Field | | | |
| | Indicates the Powershell script execution policy<br>• unrestricted: indicates that Act service in Collector can execute any kind of scripts<br>• signed, trusted: indicates that Act service in Collector can only execute scripts signed by a trusted authority | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | • signed, trusted or nexthink: indicates that Act service in Collector can only execute scripts signed by a trusted authority or by Nexthink<br>• disabled: indicates that Act service in Collector cannot execute scripts<br>"-" : data not available | | | | |
| | NXQL ID: | cltr_ra_execution_policy | | | |
| SD card encryption required | Policy | Field | | | |
| | Indicates whether SD card encryption is required. | | | | |
| | NXQL ID: | sd_card_encryption_required | | | |
| SID | Properties | Field | | | |
| | Windows security identifier for the device | | | | |
| | NXQL ID: | sid | | | |
| SMB print monitoring | Nexthink Collector | Field | | | |
| | Indicates whether SMB printing monitoring is enabled or disabled<br>• disabled: indicates that the Collector feature is disabled<br>• enabled: indicates that the Collector feature is enabled<br>"-" : data not available | | | | |
| | NXQL ID: | cltr_smb_print_mon_status | | | |
| Storage policy | Properties | Field | | | |
| | Indicates the event storage policy for the device. Possible values are:<br>• all: web requests, connections and executions are stored<br>• connections and executions<br>• executions<br>• none: no activity is recorded<br>• remove: the device will be removed from Engine during the next cleanup, as long as it is no longer sending data<br>Note that available events depend on the device platform | | | | |
| | NXQL ID: | storage_policy | | | |
| Successful HTTP requests ratio | Availability | Aggregate | | | |
| | Percentage of successful HTTP requests (1xx, 2xx and 3xx) | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | successful_http_requests_ratio | | | |
| Successful network connections ratio | Availability | Aggregate | | | |
| | Percentage of successful TCP connections | | | | |
| | NXQL ID: | successful_connections_ratio | | | |
| System boot duration baseline | Startup | Field | | | |
| | Indicates the system boot duration averaged over the last boots. In the calculation, recent boots weigh more than older boots (exponentially weighted moving average). | | | | |
| | NXQL ID: | average_boot_duration | | | |
| System drive capacity | Local drives | Field | | | |
| | Total capacity of system drive | | | | |
| | NXQL ID: | system_drive_capacity | | | |
| System drive free space | Local drives | Field | | | |
| | Total available free space on system drive | | | | |
| | NXQL ID: | system_drive_free_space | | | |
| System drive usage | Local drives | Field | | | |
| | Use percentage of system drive | | | | |
| | NXQL ID: | system_drive_usage | | | |
| Target version | Nexthink Collector | Field | | | |
| | Indicates the Collector package version that is targeted. | | | | |
| | NXQL ID: | collector_package_target_version | | | |
| Total active days | Activity | Field | | | |
| | Indicates the total number of days the device has been active. The value is updated every night. | | | | |
| | NXQL ID: | total_active_days | | | |
| Total CPU time | Activity | Aggregate | | | |
| | Indicates the sum of the CPU time of all executions on each device in scope and over all logical processors.<br><br>Executions shorter than 30 seconds are ignored.<br><br>• Example: if we consider two executions with the first one taking 50% of a logical processor during 30 minutes and the second one taking 100% of 2 logical processors during 60 minutes, the total CPU time is 135 minutes (= 50% * 30 min + 2 * 100% * 60 min). | | | | |
| | NXQL ID: | total_cpu_time | | | |

| | | | | | |
|---|---|---|---|---|---|
| Total drive capacity | Local drives | Field | | | |
| | Total capacity of all drives | | | | |
| | NXQL ID: | total_drive_capacity | | | |
| Total drive free space | Local drives | Field | | | |
| | Total free space on all drives | | | | |
| | NXQL ID: | total_drive_free_space | | | |
| Total drive usage | Local drives | Field | | | |
| | Total use percentage of all drives | | | | |
| | NXQL ID: | total_drive_usage | | | |
| Total network traffic | Traffic | Aggregate | | | |
| | Total network traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_network_traffic | | | |
| Total non-system drive capacity | Local drives | Field | | | |
| | Total capacity of all non-system drives | | | | |
| | NXQL ID: | total_nonsystem_drive_capacity | | | |
| Total non-system drive free space | Local drives | Field | | | |
| | Total free space on all non-system drives | | | | |
| | NXQL ID: | total_nonsystem_drive_free_space | | | |
| Total non-system drive usage | Local drives | Field | | | |
| | Total use percentage of all non-system drives | | | | |
| | NXQL ID: | total_nonsystem_drive_usage | | | |
| Total RAM | Hardware | Field | | | |
| | Total amount of RAM | | | | |
| | NXQL ID: | total_ram | | | |
| Total web traffic | Traffic | Aggregate | | | |
| | Total web traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_web_traffic | | | |
| UID | Properties | Field | | | |
| | Indicates the universally unique identifier (based on Engine name and device ID). | | | | |
| | NXQL ID: | device_uid | | | |
| Updater error | Nexthink Collector | Field | | | |
| | Indicates the last Nexthink Collector Updater error. | | | | |
| | NXQL ID: | updater_error | | | |

| | | | | | |
|---|---|---|---|---|---|
| Updater version | Nexthink Collector | Field | | | |
| | Indicates the Nexthink Collector Updater version. | | | | |
| | NXQL ID: | updater_version | | | |
| Uptime | Activity | Aggregate | | | |
| | Amount of time the machine has been running | | | | |
| | NXQL ID: | uptime | | | |
| User account control status | Security | Field | | | |
| | User account control status (ok, at risk or unknown) | | | | |
| | NXQL ID: | user_account_control_status | | | |
| VDI/Kiosk support | Nexthink Collector | Field | | | |
| | Indicates whether the Collector reports user logon events and user interactions in virtualized and embedded (kiosk mode) environments<br>• disabled: indicates that the Collector feature is disabled<br>• enabled: indicates that the Collector feature is enabled<br>"-" : data not available | | | | |
| | NXQL ID: | cltr_custom_shells | | | |
| Visibility from Add or Remove Programs | Nexthink Collector | Field | | | |
| | Indicates whether Collector is hidden in the "Add or Remove Programs"<br>• invisible: indicates that the Collector application is not shown in the "add or remove programs" list<br>• visible: indicates that the Collector application is shown in the "add or remove programs" list<br>"-" : data not available | | | | |
| | NXQL ID: | cltr_is_visible | | | |
| Web & Cloud monitoring | Nexthink Collector | Field | | | |
| | Indicates whether Web & Cloud monitoring is enabled or disabled<br>• disabled: indicates that the Collector feature is disabled<br>• enabled: indicates that the Collector feature is enabled | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | "-" : data not available | | | | |
| | NXQL ID: | cltr_web_mon_status | | | |
| Web interaction time | Activity | Aggregate | | | |
| | Indicates the time during which at least one executable is doing HTTP or TLS traffic. This is counted with a 5-minute resolution. | | | | |
| | NXQL ID: | cumulated_web_interaction_duration | | | |
| Windows license key | Operating system | Field | | | |
| | Indicates the Windows license key:<br>• '-': the data is not yet available<br>• 'BBBBB-BBBBB-BBBBB-BBBBB-BBBBB': Windows is activated, but the license key could not be retrieved<br>• 'BBBBB-BBBBB-BBBBB-BBBBB-?????': the full license key is not present on the machine<br>• 'Windows is not activated': Windows is not activated | | | | |
| | NXQL ID: | windows_license_key | | | |
| Windows Update status | Operating system | Field | | | |
| | Windows Update status (ok, at risk or unknown) | | | | |
| | NXQL ID: | windows_updates_status | | | |
| WMI status | Operating system | Field | | | |
| | Windows WMI service status (ok, failure) | | | | |
| | NXQL ID: | wmi_status | | | |

***Package***

Software packages (programs or updates)

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| First installation | Properties | Field | | | |
| | Date of the first package installation on any device. This field is based on data reported by the operating system and requires devices date and time to be properly set | | | | |
| | NXQL ID: | first_installation | | | |
| Name | Properties | Field | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Package name | | | | |
| | NXQL ID: | name | | | |
| Number of devices | Inventory | Aggregate | | | |
| | Number of devices | | | | |
| | NXQL ID: | number_of_devices | | | |
| Number of updates | Properties | Field | | | |
| | Number of updates (for programs) | | | | |
| | NXQL ID: | number_of_updates | | | |
| Package status | Inventory | Aggregate | | | |
| | Package status (installed/removed) | | | | |
| Platform | Properties | Field | | | |
| | The platform (operating system family) on which the package is installed | | | | |
| | NXQL ID: | platform | | | |
| Program | Properties | Field | | | |
| | Package program | | | | |
| | NXQL ID: | program | | | |
| Publisher | Properties | Field | | | |
| | Package publisher | | | | |
| | NXQL ID: | publisher | | | |
| Status | Properties | Field | | | |
| | Package status (installed/removed) | | | | |
| | NXQL ID: | status | | | |
| Type | Properties | Field | | | |
| | Package type:<br>• program<br>• update (Windows only) | | | | |
| | NXQL ID: | type | | | |
| UID | Properties | Field | | | |
| | Indicates the universally unique identifier (based on package name and package publisher). | | | | |
| Version | Properties | Field | | | |
| | Package version | | | | |
| | NXQL ID: | version | | | |

### *Application*

Sets of executables (e.g. 'Microsoft Office')

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Activity start time | Activity | Aggregate | | | |
| | Start time of investigated activity | | | | |
| | NXQL ID: | activity_start_time | | | |
| Activity stop time | Activity | Aggregate | | | |
| | Stop time of investigated activity | | | | |
| | NXQL ID: | activity_stop_time | | | |
| Application crash ratio | Errors | Aggregate | | | |
| | Indicates the number of application crashes per 100 executions. | | | | |
| | NXQL ID: | application_crash_ratio | | | |
| Application not responding event ratio | Errors | Aggregate | | | |
| | Indicates the number of application not responding events per 100 executions. | | | | |
| | NXQL ID: | application_not_responding_event_ratio | | | |
| Average incoming network bitrate | Availability | Aggregate | | | |
| | Average incoming network bitrate | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average incoming web bitrate | Availability | Aggregate | | | |
| | Average incoming bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average memory usage per execution | Activity | Aggregate | | | |
| | Indicates the average memory usage of all underlying executions before aggregation. The value is the average<br><br>memory usage of all executions (calculated with a 5-minute resolution) multiplied by their cardinalities and divided by the total cardinality.<br><br>• Example: if two tabs of the Chrome browser are opened at the same time, two distinct processes of chrome.exe are launched and they are aggregated | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | by the Engine (i.e., event cardinality = 2). The average memory usage will be the average of the two processes before aggregation: it represents the average memory usage of a Chrome tab. | | | | |
| | NXQL ID: | average_memory_usage_per_execution | | | |
| Average network response time | Availability | Aggregate | | | |
| | Indicates the average TCP connection establishment time of all underlying connections. The value is<br><br>the average TCP connection establishment time of all executions weighted by their cardinality. | | | | |
| | NXQL ID: | average_network_response_time | | | |
| Average outgoing network bitrate | Availability | Aggregate | | | |
| | Average outgoing network bitrate | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average outgoing web bitrate | Availability | Aggregate | | | |
| | Average outgoing bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average web request duration | Availability | Aggregate | | | |
| | Average time between request and last response byte | | | | |
| | NXQL ID: | average_request_duration | | | |
| Average web request size | Traffic | Aggregate | | | |
| | Average size of web requests | | | | |
| | NXQL ID: | average_request_size | | | |
| Average web response size | Traffic | Aggregate | | | |
| | Average size of web responses | | | | |
| | NXQL ID: | average_response_size | | | |
| Binary paths | Activity | Aggregate | | | |
| | List of executed binary paths (max. 50 paths) | | | | |
| Company | Properties | Field | | | |
| | Company producing the application | | | | |
| | NXQL ID: | company | | | |
| CPU usage ratio | Activity | Aggregate | | | |
| | Indicates the sum of the CPU time of all executions on each device in scope over all logical processors divided by their total duration. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Executions shorter than 30 seconds are ignored.<br>• Example: if we consider two executions with the first one taking 50% of a logical processor during 30 minutes and the second one taking 100% of 2 logical processors during 60 minutes, the CPU usage ratio is 150% (= [50% * 30 min + 2 * 100% * 60 min] / [30 min + 60 min]). | | | | |
| | NXQL ID: | cpu_usage_ratio | | | |
| Cumulated execution duration | Activity | Aggregate | | | |
| | Cumulated duration of executions | | | | |
| | NXQL ID: | cumulated_execution_duration | | | |
| Cumulated network connection duration | Activity | Aggregate | | | |
| | Cumulated duration of TCP connections | | | | |
| | NXQL ID: | cumulated_connection_duration | | | |
| Database usage | Properties | Field | | | |
| | Indicates the percentage of the Engine database used by the application. | | | | |
| | NXQL ID: | database_usage | | | |
| Description | Properties | Field | | | |
| | Application description | | | | |
| | NXQL ID: | description | | | |
| First seen | Properties | Field | | | |
| | First time activity of the application was recorded on any device | | | | |
| | NXQL ID: | first_seen | | | |
| High application thread CPU time ratio | Warnings | Aggregate | | | |
| | Indicates the ratio between the time that the underlying executions are in high thread CPU usage and their execution duration. | | | | |
| | NXQL ID: | high_application_thread_cpu_time_ratio | | | |
| Highest local privilege level reached | Activity | Aggregate | | | |
| | Highest local privilege level reached for executions (user, power user, administrator) | | | | |
| | NXQL ID: | highest_local_privilege_reached | | | |
| Incoming network traffic | Traffic | Aggregate | | | |
| | Total network incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| | Traffic | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| Incoming network traffic per device | Indicates the incoming network traffic divided by the number of devices. | | | | |
| | NXQL ID: | incoming_network_traffic_per_device | | | |
| Incoming web traffic | Traffic | Aggregate | | | |
| | Total web incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming web traffic per device | Traffic | Aggregate | | | |
| | Indicates the incoming web traffic divided by the number of devices. | | | | |
| | NXQL ID: | incoming_web_traffic_per_device | | | |
| Known packages | Properties | Field | | | |
| | List of packages known to contain the application. This list is not exhaustive: the presence of a package does not necessarily imply that on a given device the application was installed through that package | | | | |
| | NXQL ID: | known_packages | | | |
| Last seen | Properties | Field | | | |
| | Last time activity of the application was recorded on any device | | | | |
| | NXQL ID: | last_seen | | | |
| Lowest observed web protocol version | Activity | Aggregate | | | |
| | Lowest protocol version observed in web requests (excluding web requests with unknown protocol version) | | | | |
| | NXQL ID: | lowest_protocol_version | | | |
| Name | Properties | Field | | | |
| | Application name | | | | |
| | NXQL ID: | name | | | |
| Network availability level | Availability | Aggregate | | | |
| | Indicates the ratio of successful TCP connections. The possible values are:<br>• high: the ratio is greater or equal to 98%<br>• medium: the ratio is greater or equal to 90% and less than 98%<br>• low: the ratio is lower than 90% | | | | |
| | NXQL ID: | network_availability_level | | | |
| Number of application crashes | Errors | Aggregate | | | |
| | Number of application crashes | | | | |
| | NXQL ID: | number_of_application_crashes | | | |
| Number of application not responding events | Errors | Aggregate | | | |
| | Number of application not responding events | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | number_of_application_not_responding_events | | | |
| Number of binaries | Inventory | Aggregate | | | |
| | Number of binaries | | | | |
| | NXQL ID: | number_of_binaries | | | |
| Number of connections | Activity | Aggregate | | | |
| | Number of connections | | | | |
| | NXQL ID: | number_of_connections | | | |
| Number of destinations | Inventory | Aggregate | | | |
| | Number of destinations | | | | |
| | NXQL ID: | number_of_destinations | | | |
| Number of devices | Inventory | Aggregate | | | |
| | Number of devices | | | | |
| | NXQL ID: | number_of_devices | | | |
| Number of domains | Inventory | Aggregate | | | |
| | Number of domains | | | | |
| | NXQL ID: | number_of_domains | | | |
| Number of executables | Inventory | Aggregate | | | |
| | Number of executables | | | | |
| | NXQL ID: | number_of_executables | | | |
| Number of executions | Activity | Aggregate | | | |
| | Number of executions | | | | |
| | NXQL ID: | number_of_executions | | | |
| Number of ports | Inventory | Aggregate | | | |
| | Number of ports | | | | |
| | NXQL ID: | number_of_ports | | | |
| Number of users | Inventory | Aggregate | | | |
| | Number of users | | | | |
| | NXQL ID: | number_of_users | | | |
| Number of web requests | Activity | Aggregate | | | |
| | Number of web requests | | | | |
| | NXQL ID: | number_of_web_requests | | | |
| Outgoing network traffic | Traffic | Aggregate | | | |
| | Total network outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |

| | | | | | |
|---|---|---|---|---|---|
| Outgoing network traffic per device | Traffic | Aggregate | | | |
| | Indicates the outgoing network traffic divided by the number of devices. | | | | |
| | NXQL ID: | outgoing_network_traffic_per_device | | | |
| Outgoing web traffic | Traffic | Aggregate | | | |
| | Total web outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Outgoing web traffic per device | Traffic | Aggregate | | | |
| | Indicates the outgoing web traffic divided by the number of devices. | | | | |
| | NXQL ID: | outgoing_web_traffic_per_device | | | |
| Platform | Properties | Field | | | |
| | The platform (operating system family) on which the application is running | | | | |
| | NXQL ID: | platform | | | |
| Protocols used in web requests | Activity | Aggregate | | | |
| | Protocols used in web requests (HTTP, TLS, HTTP/TLS) | | | | |
| | NXQL ID: | protocols_used_in_requests | | | |
| Storage policy | Properties | Field | | | |
| | Indicates the event storage policy for the application. Possible values are:<br>• all: web requests, connections and executions are stored<br>• connections and executions<br>• executions<br>• none: no activity is recorded | | | | |
| | NXQL ID: | storage_policy | | | |
| Successful HTTP requests ratio | Availability | Aggregate | | | |
| | Percentage of successful HTTP requests (1xx, 2xx and 3xx) | | | | |
| | NXQL ID: | successful_http_requests_ratio | | | |
| Successful network connections ratio | Availability | Aggregate | | | |
| | Percentage of successful TCP connections | | | | |
| | NXQL ID: | successful_connections_ratio | | | |
| Total active days | Activity | Field | | | |
| | Total number of days the application was active | | | | |
| | NXQL ID: | total_active_days | | | |
| Total CPU time | Activity | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Indicates the sum of the CPU time of all executions on each device in scope and over all logical processors.<br><br>Executions shorter than 30 seconds are ignored.<br><br>• Example: if we consider two executions with the first one taking 50% of a logical processor during 30 minutes and the second one taking 100% of 2 logical processors during 60 minutes, the total CPU time is 135 minutes (= 50% * 30 min + 2 * 100% * 60 min). | | | | |
| | NXQL ID: | total_cpu_time | | | |
| Total network traffic | Traffic | Aggregate | | | |
| | Total network traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_network_traffic | | | |
| Total web traffic | Traffic | Aggregate | | | |
| | Total web traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_web_traffic | | | |
| UID | Properties | Field | | | |
| | Indicates the universally unique identifier (based on package name and application company). | | | | |
| Web interaction time | Activity | Aggregate | | | |
| | Indicates the time during which at least one executable is doing HTTP or TLS traffic. This is counted with a 5-minute resolution. | | | | |
| | NXQL ID: | cumulated_web_interaction_duration | | | |

***Executable***

Executable programs (e.g. 'winword.exe')

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Activity start time | Activity | Aggregate | | | |
| | Start time of investigated activity | | | | |
| | NXQL ID: | activity_start_time | | | |
| Activity stop time | Activity | Aggregate | | | |
| | Stop time of investigated activity | | | | |
| | NXQL ID: | activity_stop_time | | | |

| | | | | | |
|---|---|---|---|---|---|
| Application company | Properties | Field | | | |
| | Application company | | | | |
| | NXQL ID: | application_company | | | |
| Application crash ratio | Errors | Aggregate | | | |
| | Indicates the number of application crashes per 100 executions. | | | | |
| | NXQL ID: | application_crash_ratio | | | |
| Application name | Properties | Field | | | |
| | Application name | | | | |
| | NXQL ID: | application_name | | | |
| Application not responding event ratio | Errors | Aggregate | | | |
| | Indicates the number of application not responding events per 100 executions. | | | | |
| | NXQL ID: | application_not_responding_event_ratio | | | |
| Average incoming network bitrate | Availability | Aggregate | | | |
| | Average incoming network bitrate | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average incoming web bitrate | Availability | Aggregate | | | |
| | Average incoming bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average memory usage per execution | Activity | Aggregate | | | |
| | Indicates the average memory usage of all underlying executions before aggregation. The value is the average<br><br>memory usage of all executions (calculated with a 5-minute resolution) multiplied by their cardinalities and divided by the total cardinality.<br><br>• Example: if two tabs of the Chrome browser are opened at the same time, two distinct processes of chrome.exe are launched and they are aggregated by the Engine (i.e., event cardinality = 2). The average memory usage will be the average of the two processes before aggregation: it represents the average memory usage of a Chrome tab. | | | | |
| | NXQL ID: | average_memory_usage_per_execution | | | |
| Average network response time | Availability | Aggregate | | | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Indicates the average TCP connection establishment time of all underlying connections. The value is<br><br>the average TCP connection establishment time of all executions weighted by their cardinality. | | | | |
| | NXQL ID: | average_network_response_time | | | |
| Average outgoing network bitrate | Availability | Aggregate | | | |
| | Average outgoing network bitrate | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average outgoing web bitrate | Availability | Aggregate | | | |
| | Average outgoing bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average web request duration | Availability | Aggregate | | | |
| | Average time between request and last response byte | | | | |
| | NXQL ID: | average_request_duration | | | |
| Average web request size | Traffic | Aggregate | | | |
| | Average size of web requests | | | | |
| | NXQL ID: | average_request_size | | | |
| Average web response size | Traffic | Aggregate | | | |
| | Average size of web responses | | | | |
| | NXQL ID: | average_response_size | | | |
| Binary paths | Activity | Aggregate | | | |
| | List of executed binary paths (max. 50 paths) | | | | |
| CPU usage ratio | Activity | Aggregate | | | |
| | Indicates the sum of the CPU time of all executions on each device in scope over all logical processors divided by their total duration.<br><br>Executions shorter than 30 seconds are ignored.<br><br>• Example: if we consider two executions with the first one taking 50% of a logical processor during 30 minutes and the second one taking 100% of 2 logical processors during 60 minutes, the CPU usage ratio is 150% (= [50% * 30 min + 2 * 100% * 60 min] / [30 min + 60 min]). | | | | |
| | NXQL ID: | cpu_usage_ratio | | | |
| | Activity | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| Cumulated execution duration | Cumulated duration of executions | | | | |
| | NXQL ID: | cumulated_execution_duration | | | |
| Cumulated network connection duration | Activity | Aggregate | | | |
| | Cumulated duration of TCP connections | | | | |
| | NXQL ID: | cumulated_connection_duration | | | |
| Database usage | Properties | Field | | | |
| | Indicates the percentage of the Engine database used by the executable. | | | | |
| | NXQL ID: | database_usage | | | |
| Description | Properties | Field | | | |
| | Executable description | | | | |
| | NXQL ID: | description | | | |
| First seen | Properties | Field | | | |
| | First time activity of the executable was recorded on any device | | | | |
| | NXQL ID: | first_seen | | | |
| High application thread CPU time ratio | Warnings | Aggregate | | | |
| | Indicates the ratio between the time that the underlying executions are in high thread CPU usage and their execution duration. | | | | |
| | NXQL ID: | high_application_thread_cpu_time_ratio | | | |
| Highest local privilege level reached | Activity | Aggregate | | | |
| | Highest local privilege level reached for executions (user, power user, administrator) | | | | |
| | NXQL ID: | highest_local_privilege_reached | | | |
| Incoming network traffic | Traffic | Aggregate | | | |
| | Total network incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming network traffic per device | Traffic | Aggregate | | | |
| | Indicates the incoming network traffic divided by the number of devices. | | | | |
| | NXQL ID: | incoming_network_traffic_per_device | | | |
| Incoming web traffic | Traffic | Aggregate | | | |
| | Total web incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming web traffic per device | Traffic | Aggregate | | | |
| | Indicates the incoming web traffic divided by the number of devices. | | | | |
| | NXQL ID: | incoming_web_traffic_per_device | | | |

| | | | | | |
|---|---|---|---|---|---|
| Known packages | Properties | Field | | | |
| | List of packages known to contain the executable. This list is not exhaustive: the presence of a package does not necessarily imply that on a given device the executable was installed through that package | | | | |
| | NXQL ID: | known_packages | | | |
| Last seen | Properties | Field | | | |
| | Last time activity of the executable was recorded on any device | | | | |
| | NXQL ID: | last_seen | | | |
| Lowest observed web protocol version | Activity | Aggregate | | | |
| | Lowest protocol version observed in web requests (excluding web requests with unknown protocol version) | | | | |
| | NXQL ID: | lowest_protocol_version | | | |
| Name | Properties | Field | | | |
| | Executable name | | | | |
| | NXQL ID: | name | | | |
| Network availability level | Availability | Aggregate | | | |
| | Indicates the ratio of successful TCP connections. The possible values are:<br>• high: the ratio is greater or equal to 98%<br>• medium: the ratio is greater or equal to 90% and less than 98%<br>• low: the ratio is lower than 90% | | | | |
| | NXQL ID: | network_availability_level | | | |
| Number of application crashes | Errors | Aggregate | | | |
| | Number of application crashes | | | | |
| | NXQL ID: | number_of_application_crashes | | | |
| Number of application not responding events | Errors | Aggregate | | | |
| | Number of application not responding events | | | | |
| | NXQL ID: | number_of_application_not_responding_events | | | |
| Number of binaries | Inventory | Aggregate | | | |
| | Number of binaries | | | | |
| | NXQL ID: | number_of_binaries | | | |
| Number of connections | Activity | Aggregate | | | |
| | Number of connections | | | | |
| | NXQL ID: | number_of_connections | | | |
| Number of destinations | Inventory | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Number of destinations | | | | |
| | NXQL ID: | number_of_destinations | | | |
| Number of devices | Inventory | Aggregate | | | |
| | Number of devices | | | | |
| | NXQL ID: | number_of_devices | | | |
| Number of domains | Inventory | Aggregate | | | |
| | Number of domains | | | | |
| | NXQL ID: | number_of_domains | | | |
| Number of executions | Activity | Aggregate | | | |
| | Number of executions | | | | |
| | NXQL ID: | number_of_executions | | | |
| Number of ports | Inventory | Aggregate | | | |
| | Number of ports | | | | |
| | NXQL ID: | number_of_ports | | | |
| Number of users | Inventory | Aggregate | | | |
| | Number of users | | | | |
| | NXQL ID: | number_of_users | | | |
| Number of web requests | Activity | Aggregate | | | |
| | Number of web requests | | | | |
| | NXQL ID: | number_of_web_requests | | | |
| Outgoing network traffic | Traffic | Aggregate | | | |
| | Total network outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Outgoing network traffic per device | Traffic | Aggregate | | | |
| | Indicates the outgoing network traffic divided by the number of devices. | | | | |
| | NXQL ID: | outgoing_network_traffic_per_device | | | |
| Outgoing web traffic | Traffic | Aggregate | | | |
| | Total web outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Outgoing web traffic per device | Traffic | Aggregate | | | |
| | Indicates the outgoing web traffic divided by the number of devices. | | | | |
| | NXQL ID: | outgoing_web_traffic_per_device | | | |
| Platform | Properties | Field | | | |
| | The platform (operating system family) on which the executable is running | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | platform | | | |
| Protocols used in web requests | Activity | Aggregate | | | |
| | Protocols used in web requests (HTTP, TLS, HTTP/TLS) | | | | |
| | NXQL ID: | protocols_used_in_requests | | | |
| Storage policy | Properties | Field | | | |
| | Indicates the event storage policy for the executable. Possible values are:<br>• all: web requests, connections and executions are stored<br>• connections and executions<br>• executions<br>• none: no activity is recorded | | | | |
| | NXQL ID: | storage_policy | | | |
| Successful HTTP requests ratio | Availability | Aggregate | | | |
| | Percentage of successful HTTP requests (1xx, 2xx and 3xx) | | | | |
| | NXQL ID: | successful_http_requests_ratio | | | |
| Successful network connections ratio | Availability | Aggregate | | | |
| | Percentage of successful TCP connections | | | | |
| | NXQL ID: | successful_connections_ratio | | | |
| Total active days | Activity | Field | | | |
| | Total number of days the executable was active | | | | |
| | NXQL ID: | total_active_days | | | |
| Total CPU time | Activity | Aggregate | | | |
| | Indicates the sum of the CPU time of all executions on each device in scope and over all logical processors.<br>Executions shorter than 30 seconds are ignored.<br>• Example: if we consider two executions with the first one taking 50% of a logical processor during 30 minutes and the second one taking 100% of 2 logical processors during 60 minutes, the total CPU time is 135 minutes (= 50% * 30 min + 2 * 100% * 60 min). | | | | |
| | NXQL ID: | total_cpu_time | | | |
| Total network traffic | Traffic | Aggregate | | | |
| | Total network traffic (incoming and outgoing) | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | total_network_traffic | | | |
| Total web traffic | Traffic | Aggregate | | | |
| | Total web traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_web_traffic | | | |
| UID | Properties | Field | | | |
| | Indicates the universally unique identifier (based on application name, application company and executable name). | | | | |
| Web interaction time | Activity | Aggregate | | | |
| | Indicates the time during which at least one executable is doing HTTP or TLS traffic. This is counted with a 5-minute resolution. | | | | |
| | NXQL ID: | cumulated_web_interaction_duration | | | |

***Binary***

Executable binary files (e.g. 'winword.exe - 10.0.6843')

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Activity start time | Activity | Aggregate | | | |
| | Start time of investigated activity | | | | |
| | NXQL ID: | activity_start_time | | | |
| Activity stop time | Activity | Aggregate | | | |
| | Stop time of investigated activity | | | | |
| | NXQL ID: | activity_stop_time | | | |
| Application category | Properties | Field | | | |
| | Indicates the category of the application:<br>• '-' : not yet tagged<br>• Unknown: not categorized by Nexthink Library | | | | |
| | NXQL ID: | application_category | | | |
| Application company | Properties | Field | | | |
| | Application company | | | | |
| | NXQL ID: | application_company | | | |
| Application crash ratio | Errors | Aggregate | | | |
| | Indicates the number of application crashes per 100 executions. | | | | |
| | NXQL ID: | application_crash_ratio | | | |
| Application name | Properties | Field | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Application name | | | | |
| | NXQL ID: | application_name | | | |
| Application not responding event ratio | Errors | Aggregate | | | |
| | Indicates the number of application not responding events per 100 executions. | | | | |
| | NXQL ID: | application_not_responding_event_ratio | | | |
| Average CPU usage (deprecated) | Activity | Field | | | |
| | Indicates the average CPU usage over all logical processors since the first time the binary was seen. The value is the average CPU usage<br><br>sampled every 5 minutes for each execution divided by the number of samples. | | | | |
| | NXQL ID: | average_cpu_usage | | | |
| Average incoming network bitrate | Availability | Aggregate | | | |
| | Average incoming network bitrate | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average incoming web bitrate | Availability | Aggregate | | | |
| | Average incoming bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average memory usage (deprecated) | Activity | Field | | | |
| | Indicates the average memory usage since the first time the binary was seen. The value is the sum of the memory usage<br><br>sampled every 5 minutes for each execution divided by the number of samples. | | | | |
| | NXQL ID: | average_memory_usage | | | |
| Average memory usage per execution | Activity | Aggregate | | | |
| | Indicates the average memory usage of all underlying executions before aggregation. The value is the average<br><br>memory usage of all executions (calculated with a 5-minute resolution) multiplied by their cardinalities and divided by the total cardinality.<br><br>• Example: if two tabs of the Chrome browser are opened at the same time, two distinct processes of chrome.exe are launched and they are aggregated by the Engine (i.e., event cardinality = 2). The | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | | average memory usage will be the average of the two processes before aggregation: it represents the average memory usage of a Chrome tab. | | | |
| | NXQL ID: | average_memory_usage_per_execution | | | |
| Average network response time | Availability | Aggregate | | | |
| | Indicates the average TCP connection establishment time of all underlying connections. The value is<br><br>the average TCP connection establishment time of all executions weighted by their cardinality. | | | | |
| | NXQL ID: | average_network_response_time | | | |
| Average number of graphical handles | Activity | Field | | | |
| | Average number of graphical handles (GDI) | | | | |
| | NXQL ID: | average_number_of_graphical_handles | | | |
| Average outgoing network bitrate | Availability | Aggregate | | | |
| | Average outgoing network bitrate | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average outgoing web bitrate | Availability | Aggregate | | | |
| | Average outgoing bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average web request duration | Availability | Aggregate | | | |
| | Average time between request and last response byte | | | | |
| | NXQL ID: | average_request_duration | | | |
| Average web request size | Traffic | Aggregate | | | |
| | Average size of web requests | | | | |
| | NXQL ID: | average_request_size | | | |
| Average web response size | Traffic | Aggregate | | | |
| | Average size of web responses | | | | |
| | NXQL ID: | average_response_size | | | |
| Binary paths | Activity | Aggregate | | | |
| | List of executed binary paths (max. 50 paths) | | | | |
| CPU usage ratio | Activity | Aggregate | | | |
| | Indicates the sum of the CPU time of all executions on each device in scope over all logical processors divided by their total duration.<br><br>Executions shorter than 30 seconds are ignored. | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | • Example: if we consider two executions with the first one taking 50% of a logical processor during 30 minutes and the second one taking 100% of 2 logical processors during 60 minutes, the CPU usage ratio is 150% (= [50% * 30 min + 2 * 100% * 60 min] / [30 min + 60 min]). | | | | |
| | NXQL ID: | cpu_usage_ratio | | | |
| Cumulated execution duration | Activity | Aggregate | | | |
| | Cumulated duration of executions | | | | |
| | NXQL ID: | cumulated_execution_duration | | | |
| Cumulated network connection duration | Activity | Aggregate | | | |
| | Cumulated duration of TCP connections | | | | |
| | NXQL ID: | cumulated_connection_duration | | | |
| Database usage | Properties | Field | | | |
| | Indicates the percentage of the Engine database used by the binary. | | | | |
| | NXQL ID: | database_usage | | | |
| Description | Properties | Field | | | |
| | Description as it appears in the binary file | | | | |
| | NXQL ID: | description | | | |
| Executable name | Properties | Field | | | |
| | Executable name | | | | |
| | NXQL ID: | executable_name | | | |
| File size | Properties | Field | | | |
| | Binary file size | | | | |
| | NXQL ID: | file_size | | | |
| First seen | Properties | Field | | | |
| | First time activity of the binary was recorded on any device | | | | |
| | NXQL ID: | first_seen | | | |
| High application thread CPU time ratio | Warnings | Aggregate | | | |
| | Indicates the ratio between the time that the underlying executions are in high thread CPU usage and their execution duration. | | | | |
| | NXQL ID: | high_application_thread_cpu_time_ratio | | | |
| Highest local privilege level reached | Activity | Aggregate | | | |
| | Highest local privilege level reached for executions (user, power user, administrator) | | | | |
| | NXQL ID: | highest_local_privilege_reached | | | |

| | | | | | |
|---|---|---|---|---|---|
| Incoming network traffic | Traffic | Aggregate | | | |
| | Total network incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming network traffic per device | Traffic | Aggregate | | | |
| | Indicates the incoming network traffic divided by the number of devices. | | | | |
| | NXQL ID: | incoming_network_traffic_per_device | | | |
| Incoming web traffic | Traffic | Aggregate | | | |
| | Total web incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming web traffic per device | Traffic | Aggregate | | | |
| | Indicates the incoming web traffic divided by the number of devices. | | | | |
| | NXQL ID: | incoming_web_traffic_per_device | | | |
| Last seen | Properties | Field | | | |
| | Last time activity of the binary was recorded on any device | | | | |
| | NXQL ID: | last_seen | | | |
| Lowest observed web protocol version | Activity | Aggregate | | | |
| | Lowest protocol version observed in web requests (excluding web requests with unknown protocol version) | | | | |
| | NXQL ID: | lowest_protocol_version | | | |
| MD5 hash | Properties | Field | | | |
| | Indicates the MD5 hash of the binary. | | | | |
| | NXQL ID: | hash | | | |
| Network availability level | Availability | Aggregate | | | |
| | Indicates the ratio of successful TCP connections. The possible values are:<br>• high: the ratio is greater or equal to 98%<br>• medium: the ratio is greater or equal to 90% and less than 98%<br>• low: the ratio is lower than 90% | | | | |
| | NXQL ID: | network_availability_level | | | |
| Number of application crashes | Errors | Aggregate | | | |
| | Number of application crashes | | | | |
| | NXQL ID: | number_of_application_crashes | | | |
| Number of application not responding events | Errors | Aggregate | | | |
| | Number of application not responding events | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | number_of_application_not_responding_events | | | |
| Number of connections | Activity | Aggregate | | | |
| | Number of connections | | | | |
| | NXQL ID: | number_of_connections | | | |
| Number of destinations | Inventory | Aggregate | | | |
| | Number of destinations | | | | |
| | NXQL ID: | number_of_destinations | | | |
| Number of devices | Inventory | Aggregate | | | |
| | Number of devices | | | | |
| | NXQL ID: | number_of_devices | | | |
| Number of domains | Inventory | Aggregate | | | |
| | Number of domains | | | | |
| | NXQL ID: | number_of_domains | | | |
| Number of executions | Activity | Aggregate | | | |
| | Number of executions | | | | |
| | NXQL ID: | number_of_executions | | | |
| Number of ports | Inventory | Aggregate | | | |
| | Number of ports | | | | |
| | NXQL ID: | number_of_ports | | | |
| Number of users | Inventory | Aggregate | | | |
| | Number of users | | | | |
| | NXQL ID: | number_of_users | | | |
| Number of web requests | Activity | Aggregate | | | |
| | Number of web requests | | | | |
| | NXQL ID: | number_of_web_requests | | | |
| Outgoing network traffic | Traffic | Aggregate | | | |
| | Total network outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Outgoing network traffic per device | Traffic | Aggregate | | | |
| | Indicates the outgoing network traffic divided by the number of devices. | | | | |
| | NXQL ID: | outgoing_network_traffic_per_device | | | |
| Outgoing web traffic | Traffic | Aggregate | | | |
| | Total web outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |

| | | | | | |
|---|---|---|---|---|---|
| Outgoing web traffic per device | Traffic | Aggregate | | | |
| | Indicates the outgoing web traffic divided by the number of devices. | | | | |
| | NXQL ID: | outgoing_web_traffic_per_device | | | |
| Paths | Properties | Field | | | |
| | List of paths of the binary | | | | |
| | NXQL ID: | paths | | | |
| Platform | Properties | Field | | | |
| | The platform (operating system family) on which the binary is running | | | | |
| | NXQL ID: | platform | | | |
| Protocols used in web requests | Activity | Aggregate | | | |
| | Protocols used in web requests (HTTP, TLS, HTTP/TLS) | | | | |
| | NXQL ID: | protocols_used_in_requests | | | |
| SHA-1 hash | Properties | Field | | | |
| | Indicates the SHA-1 hash of the binary. | | | | |
| | NXQL ID: | sha1 | | | |
| SHA-256 hash | Properties | Field | | | |
| | Indicates the SHA-256 hash of the binary. | | | | |
| | NXQL ID: | sha256 | | | |
| Storage policy | Properties | Field | | | |
| | Indicates the event storage policy for the binary. Possible values are:<br>• all: web requests, connections and executions are stored<br>• connections and executions<br>• executions<br>• none: no activity is recorded | | | | |
| | NXQL ID: | storage_policy | | | |
| Successful HTTP requests ratio | Availability | Aggregate | | | |
| | Percentage of successful HTTP requests (1xx, 2xx and 3xx) | | | | |
| | NXQL ID: | successful_http_requests_ratio | | | |
| Successful network connections ratio | Availability | Aggregate | | | |
| | Percentage of successful TCP connections | | | | |
| | NXQL ID: | successful_connections_ratio | | | |
| Threat level | Properties | Field | | | |
| | Indicates the threat level of the binary: | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | • '-' : not yet tagged<br>• none detected: no known threat<br>• low: low threat<br>• intermediate: intermediate threat<br>• high: high threat | | | | |
| | NXQL ID: | threat_level | | | |
| Total active days | Activity | Field | | | |
| | Total number of days the binary was active | | | | |
| | NXQL ID: | total_active_days | | | |
| Total CPU time | Activity | Aggregate | | | |
| | Indicates the sum of the CPU time of all executions on each device in scope and over all logical processors.<br><br>Executions shorter than 30 seconds are ignored.<br><br>• Example: if we consider two executions with the first one taking 50% of a logical processor during 30 minutes and the second one taking 100% of 2 logical processors during 60 minutes, the total CPU time is 135 minutes (= 50% * 30 min + 2 * 100% * 60 min). | | | | |
| | NXQL ID: | total_cpu_time | | | |
| Total network traffic | Traffic | Aggregate | | | |
| | Total network traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_network_traffic | | | |
| Total web traffic | Traffic | Aggregate | | | |
| | Total web traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_web_traffic | | | |
| UID | Properties | Field | | | |
| | Indicates the universally unique identifier (based on binary hash). | | | | |
| User interface | Properties | Field | | | |
| | Application has interactive user interface | | | | |
| | NXQL ID: | user_interface | | | |
| Version | Properties | Field | | | |
| | Version of the binary | | | | |
| | NXQL ID: | version | | | |
| Web interaction time | Activity | Aggregate | | | |

| | | |
|---|---|---|
| | Indicates the time during which at least one executable is doing HTTP or TLS traffic. This is counted with a 5-minute resolution. | |
| | NXQL ID: | cumulated_web_interaction_duration |

***Port***

Connection ports (TCP or UDP)

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Activity start time | Activity | Aggregate | | | |
| | Start time of investigated activity | | | | |
| | NXQL ID: | activity_start_time | | | |
| Activity stop time | Activity | Aggregate | | | |
| | Stop time of investigated activity | | | | |
| | NXQL ID: | activity_stop_time | | | |
| Average incoming network bitrate | Availability | Aggregate | | | |
| | Average incoming network bitrate | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average incoming web bitrate | Availability | Aggregate | | | |
| | Average incoming bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average network response time | Availability | Aggregate | | | |
| | Indicates the average TCP connection establishment time of all underlying connections. The value is<br><br>the average TCP connection establishment time of all executions weighted by their cardinality. | | | | |
| | NXQL ID: | average_network_response_time | | | |
| Average outgoing network bitrate | Availability | Aggregate | | | |
| | Average outgoing network bitrate | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average outgoing web bitrate | Availability | Aggregate | | | |
| | Average outgoing bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |

| | | | | | |
|---|---|---|---|---|---|
| Average web request duration | Availability | Aggregate | | | |
| | Average time between request and last response byte | | | | |
| | NXQL ID: | average_request_duration | | | |
| Average web request size | Traffic | Aggregate | | | |
| | Average size of web requests | | | | |
| | NXQL ID: | average_request_size | | | |
| Average web response size | Traffic | Aggregate | | | |
| | Average size of web responses | | | | |
| | NXQL ID: | average_response_size | | | |
| Cumulated network connection duration | Activity | Aggregate | | | |
| | Cumulated duration of TCP connections | | | | |
| | NXQL ID: | cumulated_connection_duration | | | |
| First seen | Properties | Field | | | |
| | First time activity of the port was recorded on any device | | | | |
| | NXQL ID: | first_seen | | | |
| Highest local privilege level reached | Activity | Aggregate | | | |
| | Highest local privilege level reached for executions (user, power user, administrator) | | | | |
| | NXQL ID: | highest_local_privilege_reached | | | |
| Incoming network traffic | Traffic | Aggregate | | | |
| | Total network incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming network traffic per device | Traffic | Aggregate | | | |
| | Indicates the incoming network traffic divided by the number of devices. | | | | |
| | NXQL ID: | incoming_network_traffic_per_device | | | |
| Incoming web traffic | Traffic | Aggregate | | | |
| | Total web incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming web traffic per device | Traffic | Aggregate | | | |
| | Indicates the incoming web traffic divided by the number of devices. | | | | |
| | NXQL ID: | incoming_web_traffic_per_device | | | |
| Last seen | Properties | Field | | | |
| | Last time activity of the port was recorded on any device | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | last_seen | | | |
| Lowest observed web protocol version | Activity | Aggregate | | | |
| | Lowest protocol version observed in web requests (excluding web requests with unknown protocol version) | | | | |
| | NXQL ID: | lowest_protocol_version | | | |
| Network availability level | Availability | Aggregate | | | |
| | Indicates the ratio of successful TCP connections. The possible values are:<br>• high: the ratio is greater or equal to 98%<br>• medium: the ratio is greater or equal to 90% and less than 98%<br>• low: the ratio is lower than 90% | | | | |
| | NXQL ID: | network_availability_level | | | |
| Number of applications | Inventory | Aggregate | | | |
| | Number of applications | | | | |
| | NXQL ID: | number_of_applications | | | |
| Number of binaries | Inventory | Aggregate | | | |
| | Number of binaries | | | | |
| | NXQL ID: | number_of_binaries | | | |
| Number of connections | Activity | Aggregate | | | |
| | Number of connections | | | | |
| | NXQL ID: | number_of_connections | | | |
| Number of destinations | Inventory | Aggregate | | | |
| | Number of destinations | | | | |
| | NXQL ID: | number_of_destinations | | | |
| Number of devices | Inventory | Aggregate | | | |
| | Number of devices | | | | |
| | NXQL ID: | number_of_devices | | | |
| Number of domains | Inventory | Aggregate | | | |
| | Number of domains | | | | |
| | NXQL ID: | number_of_domains | | | |
| Number of executables | Inventory | Aggregate | | | |
| | Number of executables | | | | |
| | NXQL ID: | number_of_executables | | | |
| Number of users | Inventory | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Number of users | | | | |
| | NXQL ID: | number_of_users | | | |
| Number of web requests | Activity | Aggregate | | | |
| | Number of web requests | | | | |
| | NXQL ID: | number_of_web_requests | | | |
| Outgoing network traffic | Traffic | Aggregate | | | |
| | Total network outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Outgoing network traffic per device | Traffic | Aggregate | | | |
| | Indicates the outgoing network traffic divided by the number of devices. | | | | |
| | NXQL ID: | outgoing_network_traffic_per_device | | | |
| Outgoing web traffic | Traffic | Aggregate | | | |
| | Total web outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Outgoing web traffic per device | Traffic | Aggregate | | | |
| | Indicates the outgoing web traffic divided by the number of devices. | | | | |
| | NXQL ID: | outgoing_web_traffic_per_device | | | |
| Port number | Properties | Field | | | |
| | Port number | | | | |
| | NXQL ID: | port_number | | | |
| Port type | Properties | Field | | | |
| | Port type (tcp, udp, tcp port scan, udp port scan) | | | | |
| | NXQL ID: | port_type | | | |
| Port type/Port number | Properties | Field | | | |
| | Port value for tagging | | | | |
| | NXQL ID: | port_value | | | |
| Protocols used in web requests | Activity | Aggregate | | | |
| | Protocols used in web requests (HTTP, TLS, HTTP/TLS) | | | | |
| | NXQL ID: | protocols_used_in_requests | | | |
| Successful HTTP requests ratio | Availability | Aggregate | | | |
| | Percentage of successful HTTP requests (1xx, 2xx and 3xx) | | | | |
| | NXQL ID: | successful_http_requests_ratio | | | |
| | Availability | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| Successful network connections ratio | Percentage of successful TCP connections | | | | |
| | NXQL ID: | successful_connections_ratio | | | |
| Total network traffic | Traffic | Aggregate | | | |
| | Total network traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_network_traffic | | | |
| Total web traffic | Traffic | Aggregate | | | |
| | Total web traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_web_traffic | | | |
| UID | Properties | Field | | | |
| | Indicates the universally unique identifier (based on port number). | | | | |
| Web interaction time | Activity | Aggregate | | | |
| | Indicates the time during which at least one executable is doing HTTP or TLS traffic. This is counted with a 5-minute resolution. | | | | |
| | NXQL ID: | cumulated_web_interaction_duration | | | |

***Destination***

Devices receiving connections

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Activity start time | Activity | Aggregate | | | |
| | Start time of investigated activity | | | | |
| | NXQL ID: | activity_start_time | | | |
| Activity stop time | Activity | Aggregate | | | |
| | Stop time of investigated activity | | | | |
| | NXQL ID: | activity_stop_time | | | |
| Average incoming network bitrate | Availability | Aggregate | | | |
| | Average incoming network bitrate | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average incoming web bitrate | Availability | Aggregate | | | |
| | Average incoming bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average network response time | Availability | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Indicates the average TCP connection establishment time of all underlying connections. The value is<br><br>the average TCP connection establishment time of all executions weighted by their cardinality. | | | | |
| | NXQL ID: | average_network_response_time | | | |
| Average outgoing network bitrate | Availability | Aggregate | | | |
| | Average outgoing network bitrate | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average outgoing web bitrate | Availability | Aggregate | | | |
| | Average outgoing bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average web request duration | Availability | Aggregate | | | |
| | Average time between request and last response byte | | | | |
| | NXQL ID: | average_request_duration | | | |
| Average web request size | Traffic | Aggregate | | | |
| | Average size of web requests | | | | |
| | NXQL ID: | average_request_size | | | |
| Average web response size | Traffic | Aggregate | | | |
| | Average size of web responses | | | | |
| | NXQL ID: | average_response_size | | | |
| Cumulated network connection duration | Activity | Aggregate | | | |
| | Cumulated duration of TCP connections | | | | |
| | NXQL ID: | cumulated_connection_duration | | | |
| Database usage | Properties | Field | | | |
| | Indicates the percentage of the Engine database used by the destination. | | | | |
| | NXQL ID: | database_usage | | | |
| First seen | Properties | Field | | | |
| | First time activity to the destination was recorded on any device | | | | |
| | NXQL ID: | first_seen | | | |
| Highest local privilege level reached | Activity | Aggregate | | | |
| | Highest local privilege level reached for executions (user, power user, administrator) | | | | |
| | NXQL ID: | highest_local_privilege_reached | | | |

| | | | | | |
|---|---|---|---|---|---|
| Incoming network traffic | Traffic | Aggregate | | | |
| | Total network incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming network traffic per device | Traffic | Aggregate | | | |
| | Indicates the incoming network traffic divided by the number of devices. | | | | |
| | NXQL ID: | incoming_network_traffic_per_device | | | |
| Incoming web traffic | Traffic | Aggregate | | | |
| | Total web incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming web traffic per device | Traffic | Aggregate | | | |
| | Indicates the incoming web traffic divided by the number of devices. | | | | |
| | NXQL ID: | incoming_web_traffic_per_device | | | |
| IP address | Properties | Field | | | |
| | IP address for the destination | | | | |
| | NXQL ID: | ip_address | | | |
| Last seen | Properties | Field | | | |
| | Last time activity to the destination was recorded on any device | | | | |
| | NXQL ID: | last_seen | | | |
| Lowest observed web protocol version | Activity | Aggregate | | | |
| | Lowest protocol version observed in web requests (excluding web requests with unknown protocol version) | | | | |
| | NXQL ID: | lowest_protocol_version | | | |
| Name | Properties | Field | | | |
| | Reverse lookup name | | | | |
| | NXQL ID: | name | | | |
| Network availability level | Availability | Aggregate | | | |
| | Indicates the ratio of successful TCP connections. The possible values are:<br>• high: the ratio is greater or equal to 98%<br>• medium: the ratio is greater or equal to 90% and less than 98%<br>• low: the ratio is lower than 90% | | | | |
| | NXQL ID: | network_availability_level | | | |

| | | | | | |
|---|---|---|---|---|---|
| Number of applications | Inventory | Aggregate | | | |
| | Number of applications | | | | |
| | NXQL ID: | number_of_applications | | | |
| Number of binaries | Inventory | Aggregate | | | |
| | Number of binaries | | | | |
| | NXQL ID: | number_of_binaries | | | |
| Number of connections | Activity | Aggregate | | | |
| | Number of connections | | | | |
| | NXQL ID: | number_of_connections | | | |
| Number of devices | Inventory | Aggregate | | | |
| | Number of devices | | | | |
| | NXQL ID: | number_of_devices | | | |
| Number of domains | Inventory | Aggregate | | | |
| | Number of domains | | | | |
| | NXQL ID: | number_of_domains | | | |
| Number of executables | Inventory | Aggregate | | | |
| | Number of executables | | | | |
| | NXQL ID: | number_of_executables | | | |
| Number of ports | Inventory | Aggregate | | | |
| | Number of ports | | | | |
| | NXQL ID: | number_of_ports | | | |
| Number of users | Inventory | Aggregate | | | |
| | Number of users | | | | |
| | NXQL ID: | number_of_users | | | |
| Number of web requests | Activity | Aggregate | | | |
| | Number of web requests | | | | |
| | NXQL ID: | number_of_web_requests | | | |
| Outgoing network traffic | Traffic | Aggregate | | | |
| | Total network outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Outgoing network traffic per device | Traffic | Aggregate | | | |
| | Indicates the outgoing network traffic divided by the number of devices. | | | | |
| | NXQL ID: | outgoing_network_traffic_per_device | | | |
| Outgoing web traffic | Traffic | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Total web outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Outgoing web traffic per device | Traffic | Aggregate | | | |
| | Indicates the outgoing web traffic divided by the number of devices. | | | | |
| | NXQL ID: | outgoing_web_traffic_per_device | | | |
| Protocols used in web requests | Activity | Aggregate | | | |
| | Protocols used in web requests (HTTP, TLS, HTTP/TLS) | | | | |
| | NXQL ID: | protocols_used_in_requests | | | |
| Successful HTTP requests ratio | Availability | Aggregate | | | |
| | Percentage of successful HTTP requests (1xx, 2xx and 3xx) | | | | |
| | NXQL ID: | successful_http_requests_ratio | | | |
| Successful network connections ratio | Availability | Aggregate | | | |
| | Percentage of successful TCP connections | | | | |
| | NXQL ID: | successful_connections_ratio | | | |
| Total network traffic | Traffic | Aggregate | | | |
| | Total network traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_network_traffic | | | |
| Total web traffic | Traffic | Aggregate | | | |
| | Total web traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_web_traffic | | | |
| UID | Properties | Field | | | |
| | Indicates the universally unique identifier (based on destination ip address). | | | | |
| Web interaction time | Activity | Aggregate | | | |
| | Indicates the time during which at least one executable is doing HTTP or TLS traffic. This is counted with a 5-minute resolution. | | | | |
| | NXQL ID: | cumulated_web_interaction_duration | | | |

***Domain***

Domain names

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Activity start time | Activity | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Start time of investigated activity | | | | |
| | NXQL ID: | activity_start_time | | | |
| Activity stop time | Activity | Aggregate | | | |
| | Stop time of investigated activity | | | | |
| | NXQL ID: | activity_stop_time | | | |
| Average incoming web bitrate | Availability | Aggregate | | | |
| | Average incoming bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_incoming_bitrate | | | |
| Average outgoing web bitrate | Availability | Aggregate | | | |
| | Average outgoing bitrate of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | average_outgoing_bitrate | | | |
| Average web request duration | Availability | Aggregate | | | |
| | Average time between request and last response byte | | | | |
| | NXQL ID: | average_request_duration | | | |
| Average web request size | Traffic | Aggregate | | | |
| | Average size of web requests | | | | |
| | NXQL ID: | average_request_size | | | |
| Average web response size | Traffic | Aggregate | | | |
| | Average size of web responses | | | | |
| | NXQL ID: | average_response_size | | | |
| Database usage | Properties | Field | | | |
| | Indicates the percentage of the Engine database used by the domain. | | | | |
| | NXQL ID: | database_usage | | | |
| Domain category | Properties | Field | | | |
| | Indicates the category of the domain:<br>• '-' : not yet tagged or internal domain | | | | |
| | NXQL ID: | domain_category | | | |
| First seen | Properties | Field | | | |
| | The first time the domain has been seen | | | | |
| | NXQL ID: | first_seen | | | |
| Highest local privilege level reached | Activity | Aggregate | | | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Highest local privilege level reached for executions (user, power user, administrator) | | | | |
| | NXQL ID: | highest_local_privilege_reached | | | |
| Hosting country | Properties | Field | | | |
| | Indicates in which country the domain is hosted:<br>• '-' : not yet tagged, internal domain or not known by Nexthink Library | | | | |
| | NXQL ID: | hosting_country | | | |
| Hostname | Properties | Field | | | |
| | The hostname of the fully qualified domain name | | | | |
| | NXQL ID: | hostname | | | |
| Incoming web traffic | Traffic | Aggregate | | | |
| | Total web incoming traffic | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Incoming web traffic per device | Traffic | Aggregate | | | |
| | Indicates the incoming web traffic divided by the number of devices. | | | | |
| | NXQL ID: | incoming_web_traffic_per_device | | | |
| Internal domain | Properties | Field | | | |
| | Indicates whether the domain is considered internal:<br>• yes: the domain is not reported to Nexthink Library and subdomains are not compressed using the '*' pattern<br>• no: the domain is reported to the Nexthink Library (if the license includes the Security module); complex subdomains are compressed using the '*' pattern | | | | |
| | NXQL ID: | internal_domain | | | |
| Last seen | Properties | Field | | | |
| | The last time the domain has been seen | | | | |
| | NXQL ID: | last_seen | | | |
| Lowest observed web protocol version | Activity | Aggregate | | | |
| | Lowest protocol version observed in web requests (excluding web requests with unknown protocol version) | | | | |
| | NXQL ID: | lowest_protocol_version | | | |

| | | | | | |
|---|---|---|---|---|---|
| Name | Properties | Field | | | |
| | The fully qualified domain name | | | | |
| | NXQL ID: | name | | | |
| Number of applications | Inventory | Aggregate | | | |
| | Number of applications | | | | |
| | NXQL ID: | number_of_applications | | | |
| Number of binaries | Inventory | Aggregate | | | |
| | Number of binaries | | | | |
| | NXQL ID: | number_of_binaries | | | |
| Number of destinations | Inventory | Aggregate | | | |
| | Number of destinations | | | | |
| | NXQL ID: | number_of_destinations | | | |
| Number of devices | Inventory | Aggregate | | | |
| | Number of devices | | | | |
| | NXQL ID: | number_of_devices | | | |
| Number of executables | Inventory | Aggregate | | | |
| | Number of executables | | | | |
| | NXQL ID: | number_of_executables | | | |
| Number of ports | Inventory | Aggregate | | | |
| | Number of ports | | | | |
| | NXQL ID: | number_of_ports | | | |
| Number of users | Inventory | Aggregate | | | |
| | Number of users | | | | |
| | NXQL ID: | number_of_users | | | |
| Number of web requests | Activity | Aggregate | | | |
| | Number of web requests | | | | |
| | NXQL ID: | number_of_web_requests | | | |
| Outgoing web traffic | Traffic | Aggregate | | | |
| | Total web outgoing traffic | | | | |
| | NXQL ID: | outgoing_traffic | | | |
| Outgoing web traffic per device | Traffic | Aggregate | | | |
| | Indicates the outgoing web traffic divided by the number of devices. | | | | |
| | NXQL ID: | outgoing_web_traffic_per_device | | | |
| Protocols used in web requests | Activity | Aggregate | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Protocols used in web requests (HTTP, TLS, HTTP/TLS) | | | | |
| | NXQL ID: | protocols_used_in_requests | | | |
| Reputation | Properties | Field | | | |
| | Indicates the reputation of the domain:<br>• '-': internal domain or not yet tagged<br>• 'trustworthy': clean domain which has not been connected to any security risks<br>• 'low risk': benign domain which rarely delivers dangerous content<br>• 'moderate risk': generally benign domain which has exhibited potentially risky behavior<br>• 'high risk': potentially malicious domain which delivers dangerous content | | | | |
| | NXQL ID: | threat_level | | | |
| Storage policy | Properties | Field | | | |
| | Event storage policy for the domain (web request or none) | | | | |
| | NXQL ID: | storage | | | |
| Successful HTTP requests ratio | Availability | Aggregate | | | |
| | Percentage of successful HTTP requests (1xx, 2xx and 3xx) | | | | |
| | NXQL ID: | successful_http_requests_ratio | | | |
| Total web traffic | Traffic | Aggregate | | | |
| | Total web traffic (incoming and outgoing) | | | | |
| | NXQL ID: | total_web_traffic | | | |
| UID | Properties | Field | | | |
| | Indicates the universally unique identifier (based on domain name). | | | | |
| Web interaction time | Activity | Aggregate | | | |
| | Indicates the time during which at least one executable is doing HTTP or TLS traffic. This is counted with a 5-minute resolution. | | | | |
| | NXQL ID: | cumulated_web_interaction_duration | | | |

***Printer***

Installed printers (local, network, shared or virtual)

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Display name | Properties | Field | | | |
| | Most frequently seen display name | | | | |
| | NXQL ID: | real_name | | | |
| First seen | Properties | Field | | | |
| | First time activity of the printer was recorded on any device | | | | |
| | NXQL ID: | first_seen | | | |
| Hostname | Properties | Field | | | |
| | Indicates where the printer is hosted:<br>• for local and smb printers: the hostname of the device the printer is connected to<br>• for tcp/ip and wsd printers: usually the hostname of the printer itself | | | | |
| | NXQL ID: | host_name | | | |
| Last seen | Properties | Field | | | |
| | Last time activity of the printer was recorded on any device | | | | |
| | NXQL ID: | last_seen | | | |
| Location | Properties | Field | | | |
| | Printer location | | | | |
| | NXQL ID: | location | | | |
| Model | Properties | Field | | | |
| | Printer model | | | | |
| | NXQL ID: | model | | | |
| Name | Properties | Field | | | |
| | Unique printer name | | | | |
| | NXQL ID: | name | | | |
| Number of devices | Inventory | Aggregate | | | |
| | Number of devices | | | | |
| | NXQL ID: | number_of_devices | | | |
| Number of print jobs | Activity | Aggregate | | | |
| | Number of print jobs | | | | |
| | NXQL ID: | number_of_printouts | | | |

| | | | | | |
|---|---|---|---|---|---|
| Number of printed pages | Activity | Aggregate | | | |
| | Number of printed pages | | | | |
| | NXQL ID: | number_of_printed_pages | | | |
| Number of users | Inventory | Aggregate | | | |
| | Number of users | | | | |
| | NXQL ID: | number_of_users | | | |
| Type | Properties | Field | | | |
| | The type of the printer:<br>• local: a locally connected or virtual printer<br>• tcp/ip: a printer connected through a TCP/IP port<br>• smb: a printer connected through a SMB (Server Message Block) port<br>• wsd: a printer connected through a WSD (Web Services for Devices) port | | | | |
| | NXQL ID: | type | | | |
| UID | Properties | Field | | | |
| | Indicates the universally unique identifier (based on printer name and model). | | | | |

## Activities

Activities represent actions performed by Objects.

### *Installation*

Installations or uninstallations of software packages

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Device ID | Device | Field | | | |
| | Unique identifier code of the installation target device | | | | |

| | | | | | |
|---|---|---|---|---|---|
| Device name | Device | Field | | | |
| | Indicates the name of the device:<br>• For Windows: NetBios Name<br>• For Mac OS: computer name used on the network<br>• For Mobile: composed by mailbox name and device friendly name | | | | |
| Device SID | Device | Field | | | |
| | Windows security identifier of the installation target device | | | | |
| ID | Properties | Field | | | |
| | Unique installation identifier code | | | | |
| | NXQL ID: | id | | | |
| Operation type | Properties | Field | | | |
| | Type of operation (installation, uninstallation) | | | | |
| | NXQL ID: | type | | | |
| Package ID | Package | Field | | | |
| | Unique identifier code of the installed package | | | | |
| Package name | Package | Field | | | |
| | Name of the installed package | | | | |
| Package program | Package | Field | | | |
| | Program of the installed package | | | | |
| Package publisher | Package | Field | | | |
| | Name of the installed package publisher | | | | |
| Package type | Package | Field | | | |
| | Package type: | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | • program<br>• update (Windows only) | | | | | |
| Package version | Package | Field | | | | |
| | Version of the installed package | | | | | |
| Time of installation | Properties | Field | | | | |
| | Installation start time | | | | | |
| | NXQL ID: | time | | | | |

***Execution***

Executing processes (merged when in close succession)

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Application name | Application | Field | | | |
| | Executed application name | | | | |
| Average memory usage | Activity | Field | | | |
| | Indicates the average memory usage of the underlying executions before aggregation with a sampling resolution of 5 minutes.<br>• Example: if two tabs of the Chrome browser are opened at the same time, two distinct processes of chrome.exe are launched and they are aggregated by the Engine (i.e., event cardinality = 2). The average memory usage will be the average of the two processes before aggregation: it represents the average memory usage of a single Chrome tab. | | | | |
| | NXQL ID: | average_memory_usage | | | |
| Binary path | Application | Field | | | |
| | Executed binary path | | | | |
| | NXQL ID: | binary_path | | | |

| | | | | | |
|---|---|---|---|---|---|
| Binary version | Application | Field | | | |
| | Executed binary version | | | | |
| Cardinality | Properties | Field | | | |
| | Number of underlying processes, consolidated over time | | | | |
| | NXQL ID: | cardinality | | | |
| Device ID | Device | Field | | | |
| | Unique identifier code of the executing device | | | | |
| Device IP addresses | Device | Field | | | |
| | List of IP addresses of the executing device | | | | |
| Device name | Device | Field | | | |
| | Indicates the name of the device:<br>• For Windows: NetBios Name<br>• For Mac OS: computer name used on the network<br>• For Mobile: composed by mailbox name and device friendly name | | | | |
| Device SID | Device | Field | | | |
| | Windows security identifier of the executing device | | | | |
| Duration | Properties | Field | | | |
| | Total execution duration | | | | |
| | NXQL ID: | duration | | | |
| End time | Properties | Field | | | |
| | Execution end time | | | | |
| | NXQL ID: | end_time | | | |
| Executable name | Application | Field | | | |
| | Executed executable name | | | | |
| ID | Properties | Field | | | |
| | Unique execution identifier code | | | | |
| | NXQL ID: | id | | | |
| Incoming TCP traffic | Traffic | Field | | | |
| | Incoming TCP traffic | | | | |
| | NXQL ID: | incoming_tcp_traffic | | | |
| Lifespan | Properties | Field | | | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Execution lifespan in relation to investigation time frame | | | | |
| Outgoing TCP traffic | Traffic | Field | | | |
| | Outgoing TCP traffic | | | | |
| | NXQL ID: | outgoing_tcp_traffic | | | |
| Outgoing UDP traffic | Traffic | Field | | | |
| | Outgoing UDP traffic | | | | |
| | NXQL ID: | outgoing_udp_traffic | | | |
| Privilege level | Properties | Field | | | |
| | Privilege level of the execution (user, power user, administrator) | | | | |
| | NXQL ID: | privilege_level | | | |
| Signature ID | Properties | Field | | | |
| | ID of the related execution signature, i.e. a user executing a certain process on a particular device | | | | |
| | NXQL ID: | usage | | | |
| Start time | Properties | Field | | | |
| | Execution start time | | | | |
| | NXQL ID: | start_time | | | |
| Status | Properties | Field | | | |
| | Status of the execution (started, stopped) | | | | |
| | NXQL ID: | status | | | |
| Total CPU time | Properties | Field | | | |
| | Indicates the sum of the CPU time of all executions (before aggregation by the Engine) over all logical processors.<br><br>Executions shorter than 30 seconds are ignored.<br><br>• Example: if we consider two executions that are launched at the same time (hence aggregated by the Engine), with the first one taking 50% of a logical processor during 30 minutes and the second one taking 100% of 2 logical processors during 60 minutes, the | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | total CPU time is 135 minutes (= 50% * 30 min + 2 * 100% * 60 min). | | | | | |
| | NXQL ID: | total_cpu_time | | | | |
| User ID | User | Field | | | | |
| | Unique identifier code of the executing user | | | | | |
| User name | User | Field | | | | |
| | Name of the executing user | | | | | |
| User SID | User | Field | | | | |
| | Indicates the Windows security identifier for the user.<br>• For Mac 0S: the value is 'S-0-0' if the user is not in Active Directory | | | | | |

***Connection***

TCP or UDP connections (merged when in close succession)

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Application name | Application | Field | | | |
| | Name of the connecting application | | | | |
| Average network response time | Availability | Field | | | |
| | Indicates the average TCP connection establishment time. The value is the average over<br>all underlying connections before aggregation. | | | | |
| | NXQL ID: | network_response_time | | | |
| Binary paths | Application | Field | | | |
| | Paths of the connecting binary | | | | |
| Binary version | Application | Field | | | |
| | Version of the connecting binary | | | | |
| Cardinality | Properties | Field | | | |
| | Number of underlying connections, consolidated over time | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | cardinality | | | |
| Connection type | Properties | Field | | | |
| | Type of the connection (tcp, udp, tcp network scan, tcp port scan, udp network scan, udp port scan) | | | | |
| | NXQL ID: | type | | | |
| Destination IP address | Destination | Field | | | |
| | IP address of the connection destination | | | | |
| | NXQL ID: | destination_ip_address | | | |
| Destination name | Destination | Field | | | |
| | Name of the connection destination | | | | |
| Device ID | Device | Field | | | |
| | Unique identifier code of the connecting device | | | | |
| Device IP address | Device | Field | | | |
| | IP address of the connecting device | | | | |
| | NXQL ID: | device_ip_address | | | |
| Device name | Device | Field | | | |
| | Indicates the name of the device:<br>• For Windows: NetBios Name<br>• For Mac OS: computer name used on the network<br>• For Mobile: composed by mailbox name and device friendly name | | | | |
| Device SID | Device | Field | | | |
| | Windows security identifier for the connecting device | | | | |
| Duration | Properties | Field | | | |
| | The time between the start of the first connection and end of the last underlying connection | | | | |
| | NXQL ID: | duration | | | |
| End time | Properties | Field | | | |
| | Connection end time, corresponding to the moment when the last underlying TCP connection was closed | | | | |
| | NXQL ID: | end_time | | | |

| | | | | | |
|---|---|---|---|---|---|
| Executable name | Application | Field | | | |
| | Name of the connecting executable | | | | |
| ID | Properties | Field | | | |
| | Unique connection identifier code | | | | |
| | NXQL ID: | id | | | |
| Incoming bitrate | Availability | Field | | | |
| | Average incoming bitrate of all underlying connections, consolidated over time | | | | |
| | NXQL ID: | incoming_bitrate | | | |
| Incoming TCP traffic | Traffic | Field | | | |
| | Incoming TCP traffic | | | | |
| Lifespan | Properties | Field | | | |
| | Connection lifespan in relation to investigation time frame | | | | |
| Outgoing bitrate | Availability | Field | | | |
| | Average outgoing bitrate of all underlying connections, consolidated over time | | | | |
| | NXQL ID: | outgoing_bitrate | | | |
| Outgoing TCP traffic | Traffic | Field | | | |
| | Outgoing TCP traffic | | | | |
| Outgoing UDP traffic | Traffic | Field | | | |
| | Outgoing UDP traffic | | | | |
| Port number | Port | Field | | | |
| | Port number of the connection | | | | |
| Signature ID | Properties | Field | | | |
| | ID of the related connection signature, i.e. a user executing a certain process on a particular device which connects to a certain destination/port | | | | |
| | NXQL ID: | signature_id | | | |
| Start time | Properties | Field | | | |
| | Connection start time | | | | |
| | NXQL ID: | start_time | | | |
| Status | Properties | Field | | | |
| | Status of the connection (established, rejected, no service, no host, closed) | | | | |
| | NXQL ID: | status | | | |

| | | | | | |
|---|---|---|---|---|---|
| User ID | User | Field | | | |
| | Unique identifier code of the connecting user | | | | |
| User name | User | Field | | | |
| | Name of connecting user | | | | |
| User SID | User | Field | | | |
| | Indicates the Windows security identifier for the user.<br>• For Mac 0S: the value is 'S-0-0' if the user is not in Active Directory | | | | |

***Web request***

HTTP or TLS requests

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Application name | Application | Field | | | |
| | Name of the application which made the web request | | | | |
| Binary paths | Application | Field | | | |
| | Paths of the binary which made the web request | | | | |
| Binary version | Application | Field | | | |
| | Version of the binary which made the web request | | | | |
| Cardinality | Properties | Field | | | |
| | Number of underlying web requests, consolidated over time | | | | |
| | NXQL ID: | cardinality | | | |
| Connections duration | Properties | Field | | | |
| | The time between start of the first connection and end of the last underlying connection | | | | |
| | NXQL ID: | connections_duration | | | |
| Device ID | Device | Field | | | |
| | Unique identifier code of the web request source | | | | |
| Device name | Device | Field | | | |
| | Indicates the name of the device: | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | • For Windows: NetBios Name<br>• For Mac OS: computer name used on the network<br>• For Mobile: composed by mailbox name and device friendly name | | | | |
| Device SID | Device | Field | | | |
| | Windows security identifier of the web request source | | | | |
| Domain name | Domain | Field | | | |
| | Name of the web request destination domain | | | | |
| End time | Properties | Field | | | |
| | Web request end time, corresponding to the moment when the last underlying TCP connection was closed | | | | |
| | NXQL ID: | end_time | | | |
| Executable name | Application | Field | | | |
| | Name of the executable which made the web request | | | | |
| HTTP status | Properties | Field | | | |
| | HTTP response status code | | | | |
| | NXQL ID: | http_status | | | |
| ID | Properties | Field | | | |
| | Unique request identifier code | | | | |
| | NXQL ID: | id | | | |
| Incoming web traffic | Traffic | Field | | | |
| | Incoming web traffic of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | incoming_traffic | | | |
| Network response time | Availability | Field | | | |
| | Average TCP connection establishment time of all underlying connections, consolidated over time | | | | |
| | NXQL ID: | network_response_time | | | |
| Outgoing web traffic | Properties | Field | | | |
| | Outgoing web traffic of all underlying web requests, consolidated over time | | | | |
| | NXQL ID: | outgoing_traffic | | | |

| | | | | | |
|---|---|---|---|---|---|
| Port number | Port | Field | | | |
| | Port number of the web request | | | | |
| Protocol | Properties | Field | | | |
| | Web request protocol (HTTP, TLS) | | | | |
| | NXQL ID: | protocol | | | |
| Protocol version | Properties | Field | | | |
| | Web request protocol version | | | | |
| | NXQL ID: | protocol_version | | | |
| Service related | Properties | Field | | | |
| | Indicates whether the web request is related to a configured service:<br>• yes: these requests are always visible by all users<br>• no: depending on the privacy settings, requests not related to a service might not be visible by everyone | | | | |
| | NXQL ID: | service_related | | | |
| Signature ID | Properties | Field | | | |
| | ID of the related web request signature, i.e. a user executing a certain process on a particular device which emits requests to a specific domain | | | | |
| | NXQL ID: | signature_id | | | |
| Start time | Properties | Field | | | |
| | Web request start time | | | | |
| | NXQL ID: | start_time | | | |
| URL path | Properties | Field | | | |
| | Indicates the expression used to match the web request against web-based services with URL path: `'-': the web request did not match against any service with URL path` | | | | |
| User ID | User | Field | | | |
| | Unique identifier code of the user who made the web request | | | | |
| User name | User | Field | | | |
| | Name of the user who made the web request | | | | |

| | | | | | |
|---|---|---|---|---|---|
| User SID | User | Field | | | |
| | Indicates the Windows security identifier for the user who made the web request.<br>• For Mac 0S: the value is 'S-0-0' if the user is not in Active Directory | | | | |
| Web request duration | Properties | Field | | | |
| | Average time between request and last response byte of all underlying requests, consolidated over time | | | | |
| | NXQL ID: | web_request_duration | | | |

***Print job***

Print job submissions to printer drivers

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Color enabled | Properties | Field | | | |
| | Indicates whether the print job has the capability to print in color. Color settings defined by the application performing the print job (usually through the application print dialog) are not taken into account. | | | | |
| | NXQL ID: | color_print | | | |
| Device ID | Device | Field | | | |
| | Unique identifier of the print job source | | | | |
| Device name | Device | Field | | | |
| | Indicates the name of the device:<br>• For Windows: NetBios Name<br>• For Mac OS: computer name used on the network<br>• For Mobile: composed by mailbox name and device friendly name | | | | |
| Device SID | Device | Field | | | |
| | Windows security identifier of the print job source | | | | |
| Document type | Properties | Field | | | |
| | Type of printed document | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | NXQL ID: | document_type | | | |
| Duplex print | Properties | Field | | | |
| | Indicates whether the pages are printed on both sides of the sheet. | | | | |
| | NXQL ID: | duplex | | | |
| ID | Properties | Field | | | |
| | Unique print job identifier | | | | |
| | NXQL ID: | id | | | |
| Number of pages | Properties | Field | | | |
| | Number of printed pages | | | | |
| | NXQL ID: | number_of_printed_pages | | | |
| Paper size | Properties | Field | | | |
| | Paper size for printed pages | | | | |
| | NXQL ID: | page_size | | | |
| Print quality | Properties | Field | | | |
| | Print quality | | | | |
| | NXQL ID: | print_quality | | | |
| Printer model | Printer | Field | | | |
| | Model of printer | | | | |
| Printer name | Printer | Field | | | |
| | Name of printer | | | | |
| Size | Properties | Field | | | |
| | Print job size in bytes | | | | |
| | NXQL ID: | size | | | |
| Status | Properties | Field | | | |
| | Status of the print job:<br>• success: the job has been successfully sent to the printer<br>• error: an error was detected during the print; the job might have been partially printed<br>• unknown: the status of the print job could not be detected | | | | |
| | NXQL ID: | status | | | |
| Time | Properties | Field | | | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Print job time | | | | |
| | NXQL ID: | time | | | |
| User ID | User | Field | | | |
| | Unique identifier code of the printing user | | | | |
| User name | User | Field | | | |
| | Name of the printing user | | | | |
| User SID | User | Field | | | |
| | Indicates the Windows security identifier for the user.<br>• For Mac 0S: the value is 'S-0-0' if the user is not in Active Directory | | | | |

***System boot***

System boots (timed between kernel start and launch of 'logonui.exe' process)

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Device ID | Device | Field | | | |
| | Unique device identifier | | | | |
| Device IP addresses | Device | Field | | | |
| | List of IP addresses for the device | | | | |
| Device name | Device | Field | | | |
| | Indicates the name of the device:<br>• For Windows: NetBios Name<br>• For Mac OS: computer name used on the network<br>• For Mobile: composed by mailbox name and device friendly name | | | | |

| | | | | | |
|---|---|---|---|---|---|
| Device SID | Device | Field | | | |
| | Windows security identifier of the device | | | | |
| Duration | Properties | Field | | | |
| | Indicates the time between the kernel start and the launch of the 'logonui.exe' process | | | | |
| ID | Properties | Field | | | |
| | Boot event identifier | | | | |
| Time | Properties | Field | | | |
| | Time of boot | | | | |

***User logon***

User logons

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Device ID | Device | Field | | | |
| | Unique device identifier code | | | | |
| Device IP addresses | Device | Field | | | |
| | List of IP addresses for the device | | | | |
| Device name | Device | Field | | | |
| | Indicates the name of the device:<br>• For Windows: NetBios Name<br>• For Mac OS: computer name used on the network<br>• For Mobile: composed by mailbox name and device friendly name | | | | |
| Device SID | Device | Field | | | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Windows security identifier of the device | | | | |
| Duration | Properties | Field | | | |
| | Indicates the time between the user logging on and the desktop being shown. | | | | |
| Extended duration | Properties | Field | | | |
| | Indicates the time between the user logging on and the device being ready to use. Desktops and laptops are considered fully functional once the CPU<br><br>usage drops below 15% and the disk usage drops below 80%, and servers once the CPU usage of all processes belonging to the corresponding user drops below 15%. | | | | |
| ID | Properties | Field | | | |
| | User logon event identifier code | | | | |
| Time | Properties | Field | | | |
| | Time of user logon | | | | |
| User ID | User | Field | | | |
| | Unique user identifier code | | | | |
| User name | User | Field | | | |
| | Name of user | | | | |
| User SID | User | Field | | | |
| | Indicates the Windows security identifier for the user.<br><br>• For Mac 0S: the value is 'S-0-0' if the user is not in Active Directory | | | | |

## Events

Events are warning or errors.

### *Device warning*

Peaks in system resource usage (CPU, memory or I/O)

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Device ID | Device | Field | | | |
| | Unique device identifier | | | | |
| | NXQL ID: | device | | | |
| Device IP addresses | Device | Field | | | |
| | List of IP addresses for the device | | | | |
| Device name | Device | Field | | | |
| | Indicates the name of the device:<br>• For Windows: NetBios Name<br>• For Mac OS: computer name used on the network<br>• For Mobile: composed by mailbox name and device friendly name | | | | |
| Device SID | Device | Field | | | |
| | Windows security identifier of the device | | | | |
| Duration | Properties | Field | | | |
| | Indicates the duration of the event. | | | | |
| | NXQL ID: | duration | | | |
| End time | Properties | Field | | | |
| | Performance event end time | | | | |
| | NXQL ID: | end_time | | | |
| Event info | Properties | Field | | | |
| | Performance event information | | | | |
| | NXQL ID: | info | | | |
| High io usage | Warnings | Field | | | |

| | | | | | |
|---|---|---|---|---|---|
| | High io usage | | | | |
| High memory usage | Warnings | Field | | | |
| | High memory usage | | | | |
| High overall CPU usage | Warnings | Field | | | |
| | High overall CPU usage. | | | | |
| High page faults | Warnings | Field | | | |
| | High number of page faults | | | | |
| High thread CPU usage (deprecated) | Warnings | Field | | | |
| | High thread CPU usage (deprecated). | | | | |
| ID | Properties | Field | | | |
| | Unique performance event identifier | | | | |
| | NXQL ID: | id | | | |
| Start time | Properties | Field | | | |
| | Performance event start time | | | | |
| | NXQL ID: | start_time | | | |
| Warning duration | Properties | Field | | | |
| | Indicates the effective duration of the warning; it can be shorter than the event duration when the high<br><br>CPU usage is not continuous.<br><br>• Example: a high CPU usage warning started at 08:00 and lasted until 08:15 (event duration is 15 min). During this 15min, the device was effectively in high CPU usage once during 60s, twice during 120s and once during 30s; the warning duration is therefore 5min 30s. | | | | |
| | NXQL ID: | warning_duration | | | |

***Device error***

Critical system errors (system crash, hard reset, or disk failure)

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Device ID | Device | Field | | | |
| | Unique device identifier code | | | | |
| | NXQL ID: | device | | | |
| Device IP addresses | Device | Field | | | |
| | List of IP addresses for the device | | | | |
| Device name | Device | Field | | | |
| | Indicates the name of the device:<br>• For Windows: NetBios Name<br>• For Mac OS: computer name used on the network<br>• For Mobile: composed by mailbox name and device friendly name | | | | |
| Device SID | Device | Field | | | |
| | Windows security identifier of the device | | | | |
| Error code | Properties | Field | | | |
| | Indicates the error code for system crashes (Windows bluescreens). | | | | |
| | NXQL ID: | error_code | | | |
| Error label | Properties | Field | | | |
| | Indicates the error label for system crashes (Windows bluescreens). | | | | |
| | NXQL ID: | error_label | | | |
| ID | Properties | Field | | | |
| | Problem identifier code | | | | |
| | NXQL ID: | id | | | |
| Time | Properties | Field | | | |
| | Time of error | | | | |
| | NXQL ID: | start_time | | | |
| Type | Properties | Field | | | |

| | |
|---|---|
| | Indicates the device error type, with the following possible values:<br>• system crash: a Windows bluescreen<br>• hard reset: the device was abruptly stopped and then rebooted. It might be caused by pressing the reset button, a power failure or a crash<br>• SMART disk failure: a disk error was detected on a disk with SMART technology |
| | NXQL ID: type |

***Execution warning***

Peaks in application resource usage (CPU or memory)

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Application name | Application | Field | | | |
| | Name of application | | | | |
| Binary version | Application | Field | | | |
| | Version of binary | | | | |
| Device ID | Device | Field | | | |
| | Unique device identifier | | | | |
| Device IP addresses | Device | Field | | | |
| | List of IP addresses for the device | | | | |
| Device name | Device | Field | | | |
| | Indicates the name of the device:<br>• For Windows: NetBios Name<br>• For Mac OS: computer name used on the network | | | | |

| | | | | | |
|---|---|---|---|---|---|
| | • For Mobile: composed by mailbox name and device friendly name | | | | |
| Device SID | Device | Field | | | |
| | Windows security identifier of the device | | | | |
| Duration | Properties | Field | | | |
| | Indicates the duration of the event. | | | | |
| | NXQL ID: | duration | | | |
| End time | Properties | Field | | | |
| | Performance event end time | | | | |
| | NXQL ID: | end_time | | | |
| Event info | Properties | Field | | | |
| | Performance event information | | | | |
| Executable name | Application | Field | | | |
| | Name of executable | | | | |
| High memory usage | Warnings | Field | | | |
| | High memory usage | | | | |
| High thread CPU usage | Warnings | Field | | | |
| | High thread CPU usage | | | | |
| ID | Properties | Field | | | |
| | Unique performance event identifier code | | | | |
| | NXQL ID: | id | | | |
| Signature ID | Properties | Field | | | |
| | ID of the related execution event signature, i.e. a user executing a certain process on a particular device | | | | |
| | NXQL ID: | signature_id | | | |
| Start time | Properties | Field | | | |
| | Performance event start time | | | | |
| | NXQL ID: | start_time | | | |
| User ID | User | Field | | | |
| | Unique user identifier code | | | | |
| User name | User | Field | | | |
| | Name of user | | | | |
| User SID | User | Field | | | |

| | | | | | |
|---|---|---|---|---|---|
| | Indicates the Windows security identifier for the user.<br>• For Mac 0S: the value is 'S-0-0' if the user is not in Active Directory | | | | |
| Warning duration | Properties | Field | | | |
| | Indicates the effective duration of the warning; it can be shorter than the event duration when the high<br>CPU usage is not continuous.<br>• Example: a high CPU usage warning started at 08:00 and lasted until 08:15 (event duration is 15 min). During this 15min, the device was effectively in high CPU usage once during 60s, twice during 120s and once during 30s; the warning duration is therefore 5min 30s. | | | | |
| | NXQL ID: | warning_duration | | | |

***Execution error***

Application errors (crash or not responding)

| Field | Group | Type | | | |
|---|---|---|---|---|---|
| Application name | Application | Field | | | |
| | Name of application | | | | |
| Binary version | Application | Field | | | |
| | Version of binary | | | | |
| Device ID | Device | Field | | | |
| | Unique device identifier code | | | | |
| Device IP addresses | Device | Field | | | |
| | List of IP addresses for the device | | | | |

| | | | | | |
|---|---|---|---|---|---|
| Device name | Device | Field | | | |
| | Indicates the name of the device:<br>• For Windows: NetBios Name<br>• For Mac OS: computer name used on the network<br>• For Mobile: composed by mailbox name and device friendly name | | | | |
| Device SID | Device | Field | | | |
| | Windows security identifier of the device | | | | |
| Executable name | Application | Field | | | |
| | Name of executable | | | | |
| ID | Properties | Field | | | |
| | Error identifier code | | | | |
| | NXQL ID: | id | | | |
| Signature ID | Properties | Field | | | |
| | ID of the related execution error signature, i.e. a user executing a certain process on a particular device | | | | |
| | NXQL ID: | signature_id | | | |
| Time | Properties | Field | | | |
| | Time of error | | | | |
| | NXQL ID: | time | | | |
| Type | Properties | Field | | | |
| | Error type | | | | |
| | NXQL ID: | type | | | |
| User ID | User | Field | | | |
| | Unique user identifier code | | | | |
| User name | User | Field | | | |
| | Name of user | | | | |
| User SID | User | Field | | | |
| | Indicates the Windows security identifier for the user. | | | | |

| | |
|---|---|
| | • For Mac 0S: the value is 'S-0-0' if the user is not in Active Directory |

# Maximum supported values

## Overview

To guarantee the correct operation of Nexthink at an adequate level of performance, certain elements and aspects of a setup must be kept below their maximum values, as boundaries of varied nature appear in different contexts within the product.

As a reference, find below a comprehensive list of the maximum values within the platform: the number of objects that an Engine can store; the alerts, services, metrics or actions that can be simultaneously enabled; the hierarchies and the levels that organize your groups of devices; the accounts, profiles and roles that grant access either to the Portal or to the Finder, or to both; the number of results to obtain from a query; etc.

Respect the limits presented in this document to avoid unsupported scenarios.

## Objects and events

Find below the maximum number of allowed objects and events per Engine:

| Object | Default max | Absolute max | Configuration setting |
|---|---|---|---|
| Device | Licensed | 10 k | - |
| Mobile | Licensed | 5 k | - |
| User | - | As many as linked to devices | - |
| Package | - | As many as installed on devices (repeated packages expected) | - |
| Binary | 40 k | 100 k | `max_binaries` |
| Executable | - | As many as linked to binaries | - |
| Application | - | As many as linked to binaries | - |
| Port | - | 2 x 65  535 (TCP / UDP) | - |
| Destination | - | As many as required | |

| | | | |
|---|---|---|---|
| | | | A warning is sent when 50k is reached, there is no limit enforced |
| Domain | 250 k | 250 k | - |
| Printer | 62 printers per device | As many as connected to devices (respecting the limit per device) | - |
| Event | Depends on HW | 200 M | Use the Web Console to modify. |

Modify the limits on objects that allow it by changing the corresponding setting in the configuration file of the Engine. Contact Nexthink Support before modifying the default values:

```
/var/nexthink/engine/01/etc/nxengine.xml
```

In multi-Engine environments, a single Nexthink setup supports all the objects and events held by every Engine that is part of the setup, with the only limit on the total number of devices:

- For setups with more than 150 k devices, contact Customer Success Services to guide you with the deployment, validate your installation and obtain support.

### *Printers*

In contrast with the limits on other objects, the maximum number of printers is not determined by the capacity of the Engine, but by the mechanism through which the Collector handles and reports the printing events of each printer connected to a device.

Thus, the Collector supports up to 62 printers per device. If a device that runs the Collector connects to more than 62 printers, the Collector reports no print jobs for that device. Because the Collector discovers printers as a result of printing events, no printers are visible from that device either.

Nevertheless, the Engine can store as many printers in total as there are printers connected to devices, for as long as each device does not exceed its individual limit.

### *Lifetime of objects*

The lifetime of an object in Nexthink is related to the configured data retention. In the Portal, the metrics computed can extend their lifetime to several years, and the details of count metrics last for as long as there is free disk space allocated for them. As for the Engine, an object is kept in its database until all the events that are linked to that particular object have disappeared from the history of the Engine.

Three types of objects are however not immediately removed from the Engine after losing all their linked events: devices, users and binaries. These objects have an associated maximum inactivity period that may go beyond the history of events in the Engine. Thus, if the maximum inactivity period of an object has not elapsed, the object is not removed from the database of the Engine, even when there is no event linked to the object left. When the Engine records new activity for a device, a user or a binary, the inactivity period associated to the object is reset.

Because application and executable objects are linked to binaries, they are equally stored in the database of the Engine for as long as a related binary exists. Therefore, the lifetime of applications and executables is determined by the maximum inactivity period of the binaries to which they are related.

| Object | Default max inactivity period | Configuration setting |
|---|---|---|
| Device | 7 776 000 s (90 days) | `max_inactivity_period` |
| User | | |
| Binary | 2 592 000 s (30 days) | `max_binary_inactivity_period` |

Modify the `max_inactivity_period` and `max_binary_inactivity_period` settings in the configuration file of the Engine. Contact Nexthink Support before modifying the default values:

```
/var/nexthink/engine/01/etc/nxengine.xml
```

## Categories and keywords

There is no fixed maximum number of categories or keywords that can be applied to tag objects. However, a great number of categories or keywords, especially keywords that specify auto-tagging conditions, may have an impact on the overall performance of a Nexthink setup. Thus, the maximum recommended values to keep your system under an acceptable level of performance are the following:

| Item | Maximum recommended values |
|---|---|
| Category | 25 per type of object (device, user, etc.) |
| Keyword | 800 per category (200 of which with auto-tagging conditions) |
| Auto-tagging condition | 20 per keyword |

## Services

There is no fixed limit on the number of services that you can create in the Finder. However, the maximum number of simultaneously enabled services is 100.

## Alerts

### *Service-based alerts*

Create a maximum of one alert per service. Therefore, a maximum of 100 service-based alerts are permitted.

### *Investigation-based alerts*

There is no fixed limit on the number of alerts that you can create in the Finder. However, the total number of enabled investigation-based alerts per Engine cannot exceed 150 alerts, including global and user specific alerts. The distribution between global and user specific alerts is as follows:

| Alert type | Maximum |
|---|---|
| Global | 50 |
| My alerts + role based | 10 |

## Metrics

There is no fixed limit on the number of metrics that you can create in the Finder. However, there is an actual limit on the number of simultaneously enabled metrics. During the night, the Portal computes the values of enabled metrics only.

Configure the maximum number of metrics in your setup from the Web Console.

| Enabled | Default max | Absolute max |
|---|---|---|
| Metrics | 500 | 1 000 |

## Scores

There is no fixed limit on the number of scores that you can import into the Finder. However, there is an actual limit on the number of simultaneously enabled scores. The maximum number of enabled scores is limited by two factors: storage and computation power required.

| Limiting factor | Max |
|---|---|
| Storage | 500 composite or leaf scores (including subscores in the count) |
| Computation power | 50 scores of the *Computation input* type |

The total maximum number of scores is therefore 500 scores, of which a maximum of 50 scores can be of the *Computation input* type. These are not absolute maximums though. Contact Customer Success Services if you need to exceed these limits.

Starting from V6.21, the Digital Experience Score is embedded in the product. The scores that compose the Digital Experience Score are therefore regarded as part of the base product, such that they consume no part of the maximum number of scores available for the customer.

## Engage and Act

The Engage and Act modules store information on the user and device objects, respectively, by means of *custom fields*; that is, properties or attributes that are dynamically added to the data model of the object. For instance, a user object stores the answer of the user to a campaign in a dedicated custom field; whereas a device object stores the execution status of a remote action on another custom field. There is a limit on the total number of custom fields that campaigns and remote actions can add to the system.

| Custom fields of | Computation | Maximum |
|---|---|---|
| Campaign | • 3 fixed + 1 per question + 1 per optional text<br>• 0, if embedded in a remote action | 500 |
| Remote action | 6 fixed + 1 per output | 500 |

Because of performance reasons, there is also an additional limit on the number of campaigns and remote actions that can be simultaneously enabled. Regarding campaigns, note that the limit on the usable custom fields may be more restrictive than the limit on the number of campaigns themselves. In the case of

remote actions, the limit affects only automatically triggered remote actions and the number depends on the frequency of the action.

| Enabled | Maximum |
|---|---|
| Campaign | • 15 one-off or recurring campaigns<br>• 15 continuous satisfaction measurement campaigns<br>• 100 manual campaigns<br>• 1000 embedded (in remote actions) campaigns |
| Remote action | • 60 automatically triggered<br>• 30 out of the previous 60 with period < 1 hour<br>• 10 out of the previous 30 with period = 1 min |

## Portal content

The maximum number of modules, dashboards and widgets is not specified and mostly dependent on the memory available in the Portal.

## Accounts and hierarchies

The maximum number of user accounts supported by a Nexthink setup is 500 users. There is no specified limit on the maximum number of profiles and roles associated to the user accounts, but they are known to be higher than the following values:

| Items | Maximum value |
|---|---|
| Profile | > 300 |
| Role | > 400 |

Similarly, there is no specified limit on the number of hierarchies and levels that can be created, for as long as they do not exceed the maximum supported complexity in the Portal.

The maximum length of the names that you can specify for user accounts, profiles, and roles is as follows:

| Items | Maximum number of characters |
|---|---|
| Username | 128 characters |

| | |
|---|---|
| Profile | 256 characters |
| Role | 25 characters |

The password length of internally managed accounts (that is, not provisioned from SAML or Active Directory) has the following limits:

| Item | Minimum number of characters | Maximum number of characters |
|---|---|---|
| Password | 8 characters (configurable) | 25 characters |

### *Entities*

The maximum total number of entities in a Nexthink setup is 8 000 entities, which can be distributed among Engines with a maximum of 500 entities per Engine.

The maximum number of rules that can be specified in the CSV file to define entities is 1 000 rules per entity.

## Finder

### *Investigations*

There is no specified limit in the number of investigations that can be created on the Finder. Related to investigations, the maximum number of results shown by a cross-Engine investigation is 10 000 entries (can be overcome by exporting the results of the investigation).

The maximum query time of an investigation is 900 s (15 min). After that, the Engine aborts the query.

### *Network and Web activity views*

The maximum number of connections or web requests that can be visualized in the Network and Web activity views are 10 000 connections or requests per view.

Related references

- Increasing the maximum number of metrics
- Maximum number of remote actions
- Limit on the number of scores
- Maximum number of published campaigns
- Limit on the number of alerts
- Maximum number of users

- Maximum number of binaries
- Limit on the number of printers
- Domain replacement
- Setting the maximum number of events
- Setting the minimum password length for local accounts
- Limit on the number of rules per entity
- Maximum entities per Engine
- Maximum entities per Portal and complexity

# Local and shared content

## Overview

Starting from Nexthink V6.17, almost every item that a user creates in the Finder is centralized, meaning that the item is added to a content manager that resides in the same Appliance as the Portal. In its turn, the content manager synchronizes all connected Engines to hold a copy of every added item. The result is that all Engines share the same user-created content, offering a unified experience to Finder users.

There are a few exceptions and special cases that we detail below.

## Classification

The following table classifies items according to their level of sharing:

| Centralized | | Local to Engine | Local to Finder |
|---|---|---|---|
| **Owned** | **Shared** | | |
| • Investigations<br>• One-click investigations<br>• My alerts | • Categories and keywords<br>• Metrics<br>• Services<br>• Campaigns<br>• Scores<br>• Remote actions<br>• Global alerts | • Tags | • Sessions<br>• Custom actions |

### *Owned content*

Even if most of the content is replicated in all Engines, some types of replicated items belong exclusively to the user that created them. Thus, when connecting to an Engine with the Finder, only the owner of that content is able to see it. Investigations, one-clicks and the alerts in the **My alerts** section fall into this category of items.

### *Shared Content*

Many of the items that are replicated by the content manager are shared by all users. However, these items are usually associated to a configuration option in the profile of the user, so that only the users with the corresponding option ticked are able to manipulate their content. For instance, although all users can see the visible scores of an object, only those whose profile allows *system configuration* can manage the scores. For a list of all the profile options, see how to define user profiles.

Services, for instance, are also visible to all users, although only those with the specific permission in their profile can modify the definition of a service. Similarly, while categories and keywords can be managed by users with the right permission only, their effect on objects (that is, the tags) are visible to everyone. Other types of items such as metrics, campaigns and remote actions are removed from the left-hand panel of the Finder as well when users do not have the permission to modify them.

### *Content local to the Engine*

Although tags do not really qualify as user-created content, they deserve to be mentioned in this section about locality, as tags are the only items which are local to the Engine and that users can modify.

Tags are neither centralized nor shareable. A tag is the result of applying a keyword of a category to an object either manually, automatically, or with the help of text files. Objects (devices, destinations, etc) live in the database of each Engine; therefore, objects are local to an Engine, and so are the tags applied to an object. The locality of tags has some important implications that we explain with the following example.

Suppose that you have a setup with two Engines and one Portal, and in both Engines there is a destination with the same IP address. In fact, even if it is logically the same destination, there are two destination objects: one per Engine. If you create a category and a keyword to automatically tag the destinations with

that IP address as, for instance, a *Mail server*, both Engines will tag the destination identically, though separately. Indeed, categories and keywords are centralized, so the auto-tagging condition on the IP address applies to all Engines.

Now let us imagine that you connect to one of the Engines and that you manually tag the mentioned destination as a *Proxy server*, overriding the auto-tagging rule. Now you find that the same destination is tagged as *Mail server* in one Engine and as *Proxy server* in the other, which is probably not what you want. Therefore, be careful when you apply tags manually (or with text files) to objects, because tags are not shared among Engines. To modify tags, a user must have a profile that allows editing of applications and object tags.

### *Content local to the Finder*

A few types of items are stored in the computer that runs the Finder. These items are not implicitly shared with any other copy of the Finder or with any other Nexthink component:

Sessions
: Store information on how to connect to the Portal (user name, authentication method, etc.).

Custom actions
: Launch external operations based on the data displayed in the Finder. Custom actions can be exported and imported into another Finder (see below).

## The case of alerts

When a user creates an investigation-based alert with the Finder, the alert is stored in the centralized content manager and replicated in all Engines, but it is enabled only in the Engine to which the Finder is connected. All other Engines will remain with the alert disabled until it is explicitly enabled. This mechanism prevents a user from exceeding the limit of enabled alerts in other Engines different from the Engine to which the Finder is connected.

## Exporting and importing content

Even if most of the content created by users with the Finder is centralized, you can still share it manually by exporting items to the clipboard or to a file and, optionally, create a content pack. Because content is shared by all Engines connected to a Portal, exporting content is specially useful in multi-environment setups with more than one Portal. For instance, it is custom to create content in a

pre-production environment and then export it to a production environment only once every item has been thoroughly tested.

Manually export and import your investigations, one-clicks, alerts, categories, metrics, scores, remote actions, and services from the Finder.

***Custom actions***

Although local to the Finder, custom actions are also exportable. Share your custom actions with other copies of the Finder by exporting them to XML files:

1. Click the sprocket icon in the top right part of the Finder window.
2. Select **Custom actions...** to display the list of available custom actions.
3. Select one or more entries in the list by clicking on them. Use **Ctrl+click** to select more than one entry, **Shift+click** to select a group of consecutive entries, or **Ctrl+A** to select them all.
4. Right-click your selection and choose **Export...** from the menu.
5. Type in a name for the XML file.
6. Click **Save**.

To import custom actions into a Finder, click the **Import...** button at the bottom of the list of custom actions and select the file to import. If an imported custom action exists already in the list, it is duplicated.

## Centralizing content via roles

Administrators can assign owned content such as investigations, one-click investigations, investigation-based alerts, and remote actions to other users by making them role-based. Once linked to a role, an item is seen by all the users playing that role and not just by the user that created the item. To add investigations, one-clicks, or alerts to a role, administrators typically use the manual method of exporting items to the clipboard.

Related tasks

- Manually sharing Finder content
- Exporting a content pack
- Tagging objects manually
- Tagging objects automatically
- Importing tags from text files

Related references

- Adding users (defining user roles)
- Adding users (defining profiles)

# Device Identification

## Overview

To update their database with consistent information, Engines must correctly identify the different devices from which they receive Collector data. The Engine is able to distinguish devices from one another thanks precisely to the hardware information and operating system-level data sent by the Collectors.

Because device hardware may get upgraded and the device data stored at the operating system-level may change with time, the Engine uses an algorithm to either recognize a device to be the same as a device seen before, despite possible minor changes, or decide that a new device joined the network. Failing to correctly identify a device may result in a single device being split into two or in two different devices being merged into one in the database of the Engine.

To prevent Engines from misidentifying special groups of devices, such as those in virtualized environments, replace the default identification algorithm by an algorithm exclusively based on the name of the device, as seen by the operating system. Apply this name-based recognition method to groups of devices selected by name patterns.

The methods to identify devices described in this article do not apply to mobile devices.

Applies to platforms:

## Default algorithm to identify a device

To identify a device, the default algorithm considers the following pieces of information:

- The name of the device, as reported by the operating system.
- A hardware identifier that is derived from:
    - The BIOS serial number.
    - The chassis serial number.
    - The motherboard serial number.
- The MAC addresses of the network adapters that are enabled on the device.

- The Machine SID of the device.

***Considerations about the data that identifies a device***

Devices that have not joined a domain may share the same name. For devices in a domain, the name of the device is unique at a given time within a given domain. Name uniqueness is ensured by the domain controller, but two different devices may have the same name at different points in time.

The list of MAC addresses that are enabled by the operating system change whenever a network adapter is added or removed.

The derived hardware identifier is usually unique for branded PCs but it may not be unique for no name or self-assembled PC. In the case of devices being virtual machines, VMWare defines a BIOS serial number that is unique and thus yields a valid hardware id.

The Machine SID of a Windows device is the Security Identifier of the Windows operating system. The SID is generated during the Windows installation process and is supposed to be globally unique. However if Windows is installed using a cloned image which has not been carefully crafted using sysprep, the SID may not be unique. Experience shows that SIDs are rarely unique within corporate network and they appear in bunches of 10 to 50 machines.

***How the device identification algorithm works***

The exact identification algorithm is quite intricate; therefore, it is not described here in detail, but only sketched out. Basically, when the Collector sends to the Engine all the pieces of information about a device mentioned above, the device identification algorithm compares them with the corresponding data of each device that is already present in the database of the Engine:

- If the received information precisely matches that of an existing device, the algorithm concludes that the information belongs to the same device that is already in the database.
- If most of the information at least partially matches that of an existing device, in a majority of cases the algorithm still concludes that the information belongs to the same device. The Engine updates therefore the existing device with the received information. For instance, if the received hardware id, MAC addresses and SID all match those of an existing device, but the received name is different from the name of the device as recorded in the database, the algorithm determines that it is the same device and updates its name in the database.

- If the received information differs significantly from that of any of the existing devices, the Engine adds a new device to its database.

## Identifying devices solely by their name

Starting from the Engine release V5.3.3, it is possible to override the default algorithm to identify devices and instruct the Engine to exclusively identify Windows devices with domain membership by their name. From release V6.8 on, the feature has been extended to support all devices regardless of their platform (Windows or Mac) and membership type. And from release V6.20 on, the option to identify devices exclusively by their name is configurable through the Web Console and applied to all Engines at once. If an Engine was configured to identify devices solely by their name previously to V6.20, the patterns in the configuration file of the Engine take precedence over the configuration specified in the Web Console. To unify the configuration of device identification, manually remove the device identification settings from the configuration file of each Engine (`/var/nexthink/engine/01/etc/nxengine.xml`).

Note that the default device identification algorithm should be preferred in most cases. Use this alternative method only in setups where the default algorithm fails to reliably identify a specific group of devices. A misconfiguration may lead to devices being artificially merged or split, so use the identification of devices by name carefully.

This feature is particularly useful in virtualized environments, where devices are virtual machines (VMs) recreated at every user session. By applying the default algorithm for identifying devices, the Engine regards every new instance of a VM as a new device and ends up with multiple devices that share the same name and that succeed each other over time. By identifying devices on the basis of their name only, the Engine consistently maps a particular VM to a single device time after time, even when its hardware properties change.

To apply the identification by name to a set of devices, specify name patterns in the corresponding configuration page of the Web Console. Only those groups of devices whose names match any of the specified patterns will be identified solely by their name. All other devices follow the usual identification process:

1. Log in to the Web Console as administrator.
2. Select the **APPLIANCE** tab at the top of the Web Console.
3. Click **Collector management** on the left-hand side menu.
4. Under **Collector identification**, type in the desired name patterns inside the box **Device name patterns** and separate each pattern with a new line.

For instance, if the name of all your virtual machines begins with **vm1-ws** or **vm2-ws**, type in:

```
vm1-ws*
vm2-ws*
```

5. Click **SAVE** to apply your changes.

Valid substitution characters in the name patterns are:

- The asterisk ***** to substitute for zero or more characters.
- The question mark **?** to substitute for one single character.

# Timestamping of events

## Overview

Learn how the Engine computes the timestamps of the events in its database by combining the time of reception of the packets sent by the Collectors with the individual time information of each event stored inside every packet.

## Timestamping in the Collector

The Collector reacts to events of interest by recording them into memory. Later, it sends the collected events to the Engine either periodically or when it has accumulated a sufficient quantity of events. To detect activity in the system where it is installed, the Collector employs different techniques, such as intercepting system calls, that allow it to precisely determine the moment at which the event takes place.

This timestamping of the events in the device of the end-user is done according to the time elapsed since the last system boot. Therefore, the Collector is using relative time to timestamp events in the machine of the end-user. The fact that the time used by the Collector is relative is not important for computing a precise timestamp in the Engine, as you will see below.

Once the Collector has gathered enough events, along with their corresponding timestamps, it builds a network packet and sends it to the Engine. Right before sending it, the Collector sets a timestamp in the packet using again the local time relative to the system boot. Therefore, in every packet sent by the Collector we have:

- The timestamps of each individual event sent in the packet.

- One general timestamp for the packet itself.

The difference between the time of the packet and the time of each individual event is used by the Engine to compute the global timestamp of the events.

## Timestamping in the Engine

Once the Engine receives a packet from the Collector, it records the time of reception using the system time. For the recorded time to be correct, the local time of the Engine must be synchronized to an accurate clock. That is one of the reasons why it is recommended to set up NTP in the appliance that hosts the Engine.

For computing the global time of events, the Engine assumes that the transmission time of the Collector packets from the computers of the end-users to the Engine is negligible. In this way, the absolute time at which the Collector sent the packet is considered to be the same absolute time of reception of the packet in the Engine. Therefore, the time that the Collector set in the packet just before sending it is the local time of the end-user machine relative to system boot that is equivalent to the Engine time of reception of the packet. To get the occurrence time of each event, the Engine just has to subtract the difference between the local times of the packet and of each event from the reception time, as shown in the figure below:

Note that the events received in the Engine may not follow a sequential order. The most common case is when you receive two packets in a short interval of time and each packet is coming from a different Collector. Most probably, the two packets have events that overlap in time, but the Engine processes all the events of the second packet after those of the first. The Engine deals well with this situation. Note also that the Engine always inserts events in the past with respect to its current time. This is obvious, for the Engine cannot receive events that have not happened yet. However, for events that lie too far in the past, the Engine will not be able to update the in-memory database, since it would be a too costly operation:

- The Engine rejects events that lie more than **30 minutes** in the past with respect to its present time.

For Collectors in a local network, however, this is a very unlikely case and it often indicates a problem in the device that hosts the Collector.

### The case of TCP connections

The Collector treats TCP connections differently from all other events regarding the setting of their timestamp. All other events have their timestamp set as soon as they begin to do some kind of activity. On the other hand, when the end-user device opens a TCP connection to a server, the Collector waits for the connection to be established to set the timestamp of the TCP connection event.

In versions of the Engine previous to 4.4.3, the Engine does not take into account the time for establishing the connection to compute the start time of the event. From version 4.4.3 on, the Engine substracts the connection establishment time from the timestamp of TCP connections to get the actual initial moment of the event.

## Boot and logon duration

### Overview

The startup time of a device has a direct impact on the productivity and the experience of end-users. Since the first activities that a user performs on a device are to power it on and to log on, users typically have a very negative perception of devices that take too long to start. Indeed, a long boot or logon process are often a symptom of other underlying problems in a device, such as disk failures, network issues, low memory, or general obsolescence. Nexthink provides the following measurements of the boot and logon duration of a device:

Boot duration
: After powering on the device, the boot duration is the time between the start of the OS kernel and the launch of the logon screen.

Logon duration
: The time between user authentication and the desktop being shown.

Extended logon duration
: The time between user authentication and the device being ready.

Because of the techniques employed in the measurement of boot and logon duration, these values apply to Windows devices only.

Applies to platforms:

## Measurement of the boot duration

The measurement of the boot duration begins when the kernel of the operating system loads the Collector driver during its initialization. Once up and running, the Collector notifies the boot of the device and then continuously reports the time elapsed since the kernel started (the system boot, as recorded by the operating system) to the Engine. Any steps in the boot sequence previous to the start of the kernel, such as the BIOS hardware checks and the loading of the kernel itself, are therefore not included in the boot duration. The Engine establishes the absolute boot time of the device according to this information.

In addition to the boot time, the Engine needs to know when the operating system launches the logon screen to compute the boot duration. The launch of the logon screen corresponds to the execution of the system process *logonUI.exe*. Since the Collector successively informs the Engine of the processes being executed in the device, the Engine just needs to wait for the Collector to detect the launch of *logonUI.exe*. The Engine records the interval between the boot time and the start of *logonUI.exe* as the boot duration.

Note that Nexthink records boot events only for *full boot* sequences. Waking up the device after being in a standby (sleep) or a hibernation state is not considered a device boot. Moreover, the boot technique known as *Fast Startup* in Windows 8 (and higher) is not a full boot sequence either; therefore, it is not recorded as such.

| **Boot duration** | |
|---|---|
| **Start** | **Stop** |
| • System boot (as recorded by the OS) | • Start of logon screen (launch of *logonUI.exe*) |

## Measurement of the logon and extended logon durations

The moment when the user finishes authenticating, either by typing in their

credentials or by any other identification means, marks the start of the logon process. The Collector has two ways to detect the start of the logon process:

- Look in the Security log for an audit logon event.
- Wait for a session creation event.

The preferred method for the Collector to detect a user logon is to look for audit logon events in the Security log of Windows. For the Security log to include logon information, it is necessary that the system administrator activates the corresponding audit policy option. The logon time detected by the Collector in this case matches thus the time recorded by Windows.

Nevertheless, if the audit policy on the device does not include the audit of logon events, the Collector defaults to detecting user logons by listening to session creation events. Capturing the moment of creation of a session is usually a precise method to determine the time of a user logon. However, in setups with *roaming user profiles*, using this method could yield logon durations that are much shorter than the actual logon duration experienced by users. To avoid sending inaccurate information, if the audit of logon events is not enabled, the Collector does not report the logon duration of users with roaming profiles. For more information on roaming user profiles, see the next section.

Both the logon and the extended logon durations take the start of the logon process as the beginning of their measurement, but they differ from each other in their ending point:

- The appearance of the desktop marks the end of the logon duration.
- After the desktop is shown, the readiness of the device to being used marks the end of the extended logon duration. The device is considered to be *ready to use* when the operating system frees enough resources so that the device becomes responsive again to the commands of the user. Depending on the type of device, the resource consumption for considering the device to be ready is as follows:
    - Desktops and laptops: the CPU usage drops below 15% and the disk usage below 80%.
    - Servers: the CPU usage of all the processes that belong to the logged on user drops below 15%.

If the consumption of resources in the device is still higher than required 25 minutes after user authentication, the Collector stops waiting and reports the logon duration as 25 minutes.

| **Logon duration** | |
|---|---|

| Start | Stop | |
|---|---|---|
| • User authentication | • Desktop is shown | • Device is ready to use |
| **Start** | **Continue** | **Stop** |
| **Extended logon duration** | | |

***Logon duration in devices with roaming user profiles***

A roaming user profile is a concept in Windows that allows users to have a consistent desktop experience across different computers within the same network. Independently of the computer that they choose to work with, the users have access to their personal documents, the applications remember their preferences and the desktop appearance remains the same. In this section, learn how roaming user profiles may impact the measurement of the logon duration.

When roaming users log on a device, the loading of their profile can take a substantial part of the logon time. However, the new session starts only after the profile is completely loaded. If the Collector just waited for the session creation event to compute the logon duration, it would ignore the time spent to load the user profile as part of the logon duration. Because of this omission, the Collector would report much smaller logon durations than the actual values for the logon duration of roaming users. Therefore, the Collector never uses this method for computing the logon duration of roaming users.

The alternative is to get logon information from the Security log of Windows. Logon events in the Security log always report the correct logon time. For this reason, auditing logon events is the preferred method for the Collector to compute the logon duration of all kinds of users and it is mandatory for roaming users. For devices with roaming user profiles, remember to always activate the audit of logon events. Failing to do so results in the Collector not reporting the logon duration of users with roaming profiles.

Related tasks

- Auditing logon events

# Memory and CPU usage

Applies to platforms:

## Overview

Measuring the utilization of the hardware resources within each device in your organization is key to evaluate both the efficiency of devices and the impact of resource consumption on end-user experience. Users that perceive their devices as slow usually suffer from scarcity or misuse of two basic system resources: main memory and CPU processing power.

In this article, learn about the fields and aggregate values that measure the usage of memory and processing power in Nexthink. Based on these figures, assess the amount of resources given to a particular device and find out those applications that are most eager for resources.

## Memory usage

The Collector takes samples of the amount of memory used by each running process with a period of 30 seconds. Every 5 minutes, interval, the Collector calculates the average value of these samples over the past 5 minutes for sending it later to the Engine. Note that if a process allocates and frees memory very quickly, the Collector may miss some peaks of memory consumption when taking its samples every 30 seconds. Therefore, there is always some uncertainty in the values offered by the Collector, but it is usually negligible for well-behaved applications. Moreover, memory issues typically arise because of a sustained high consumption of memory and not because of short-lived allocations.

Based on the data collected, the following fields and aggregates are available for measuring memory usage in Nexthink:

| Name | Type | Applies to | Description |
|---|---|---|---|
| Average memory usage | field | • execution | The average memory usage of the execution before being aggregated. |
| Average memory usage per execution | aggregate | • user<br>• device<br>• application<br>• executable<br>• binary | The average memory usage of all the underlying executions divided by their cardinality. |
| Average memory usage (deprecated) | field | • binary | The average memory usage of the underlying execution with a sampling rate of 5 minutes |

Note that the memory usage is calculated per process before they may be aggregated by into a single execution. A single binary may spawn several identical processes in memory, resulting in a total memory consumption higher than that of the individual processes.

For instance, we can look at the behaviour of two well-known web browsers: Chrome and Firefox. Chrome creates a new process for every tab that the browser opens, while Firefox uses a single process for all tabs. Therefore, Firefox will typically report a higher average memory usage than Chrome for similar use cases when multiple tabs are open, because the memory utilization is reported per process, before processes are aggregated in the Engine.

## CPU usage

In the case of CPU usage, the Collector takes samples of the CPU load of all running processes every 30 seconds. The CPU load is measured as a percentage value from 0 to 100 for each logical processor that is present in the device. Therefore, the CPU load can be higher than 100% for devices with multiple logical processors. For instance, a device with 12 logical processors has a maximum CPU load capacity of 1200%.

Contrary to memory usage samples, CPU samples are not averaged before sending them to the Engine, which lets the Engine know about peaks of CPU utilization. Still, note that the maximum instantaneous load of CPU may not occur simultaneously with the moment when the Collector takes a sample. The Collector sends the CPU samples to the Engine every 5 minutes. For every sample, the Engine calculates the effective CPU utilization of each process during its execution. Retrieve it using the following fields and aggregates:

| Name | Type | Applies to | Description |
|---|---|---|---|
| Total CPU time | field | • execution | The effective utilization time of the CPU during the aggregated execution. Note that the total CPU time can exceed the total duration of the execution if the average CPU load was over 100%. |
| Total CPU time | aggregate | • user<br>• device<br>• application<br>• executable<br>• binary | The sum of the total CPU time of all executions within the scope of the selected object. |
| CPU usage ratio | aggregate | | |

| | | | |
|---|---|---|---|
| | | • user<br>• device<br>• application<br>• executable<br>• binary | The sum of the total CPU time of all executions divided by their duration within the scope of the selected object. |
| Average CPU usage (deprecated) | field | • binary | Average CPU load of a binary over all logical processors, taking into account all its executions since the binary was first seen. Note therefore that this value does not depend on the selected time frame. |

Note that these figures are based on the samples taken directly from running processes. The Collector also takes samples of the total CPU load reported by a device (not broken down by processes), but these are just used to signal high CPU conditions in the device.

Related references

- Errors and warnings for devices and executions
- Warnings tooltips

# Status of TCP connections

The status of a TCP connection can be one of the following:

established
: The TCP connection has been accepted by the remote party and is still active.

closed
: The TCP connection has been closed after being successfully established.

no service
: The remote party acknowledged the initial SYN message by a RST message; i.e. the remote party does exist, but no service is bound to the request port. Note: Most personal computers are protected by a firewall that discards RST messages to prevent effective port scanning.

no host
: The remote party does not acknowledge the SYN message; i.e. the remote party does not exist or it is protected by a firewall.

rejected
: The TCP connection was rejected by the operating system itself; for instance due to security settings.

Related concepts

- Connection

## Status of UDP connections

UDP is a stateless protocol; therefore, a UDP connection has no status in a strict sense. Nevertheless, for keeping them similar to TCP connections, UDP connections also include a *status* field in Nexthink.

Because of the very nature of the UDP protocol, the status of a UDP connection does not indicate success or failure in the delivery. However, the system can deduce whether the connection is still ongoing or if it has expired. Thus, the two possible statuses of a UDP connection are:

established
: While events are being aggregated to the same connection (increasing its cardinality), the system considers that the connection is still open; that is, the device keeps sending datagrams to the same destination via the same UDP port.

closed
: When the aggregation time has passed and there are no new events to add to the connection, the UDP connection is considered closed.

The aggregation time depends on your settings for aggregating events (minimum 5 minutes).

Related tasks

- Establishing a data retention policy in the Engine

Related references

- Status of TCP connections

Related concepts

- Connection

## Network and port scan conditions

Nexthink considers a set of connections to be a network or port scan when the following conditions are met:

- A single process is starting all the connections.
- Each connection in the set is separated from the previous connection by less than 90 seconds; that is, one minute and a half.
- The set of connections contains at least 50 connections.
- The set of connections only contains *failed* connections.

The reason to include the last condition is that a scan operation does not usually complete the vast majority of its connection attempts. Since a scan blindly tests every port or destination, most of the connections are rejected. The way to express this last condition depends on the transport protocol of the connection. In the case of TCP, the status of the connection directly shows whether the connection failed or not. In the case of UDP, however, there is no clear status of the connection. Therefore, Nexthink suspects a UDP scan when many small UDP packets are sent in a short period of time:

TCP
: All connections in the set are unsuccessful.

UDP
: The size of each packet sent is less than 10 KB.
The total duration of the whole scan is less than 15 minutes.

To summarize, this is the list of all the types of network and port scan that you can find:

TCP network scan
: A process launches a burst of unsuccessful TCP connections to the same port of at least 50 destinations.

UDP network scan
: A process sends a burst of small UDP datagrams to the same port of at least 50 destinations within 15 minutes.

TCP port scan
: A process launches a burst of unsuccessful TCP connections to at least 50 ports on the same destination.

UDP port scan
: A process sends a burst of small UDP datagrams to at least 50 ports on the same destination within 15 minutes.

Related concepts

- Connection
- Port

# Incoming traffic measurement

## Overview

Nexthink measures the network traffic that enters a device in two different ways:

Per connection
: The traffic received as response to an outgoing connection.

Per execution
: The amount of traffic received during the execution of a program.

These two ways of measuring the incoming traffic may produce different results if some devices in your network behave as servers.

## Incoming traffic per connection

Since Nexthink records the connections of a device only when the device acts as client, that is, when they are initiated by the device itself, the incoming traffic per connection is exclusively due to the responses received from the corresponding servers through these outgoing connections.

Because the UDP protocol requires a device to act as a server to receive any data, only TCP connections may report incoming traffic.

Rembember that Nexthink does not record the incoming connections to a device; that is, the connections that enter a device when it acts as a server.

## Incoming traffic per execution

Contrary to the measurement of incoming traffic for individual connections, the measurement of incoming traffic during the execution of a program does take into account the incoming connections to the device.

Therefore, if a program accepts connections on a particular port, making the device act like a server, the received data is visible in the amount of traffic associated to the execution of the program, but not as an individual connection.

## Network and Local activity views

Both the Network and the Local activity views in the Finder let you visually examine the incoming network traffic of devices. While the Network activity view aggregates incoming traffic values per connection, the Local activity view collects measurements of the incoming traffic per execution.

Therefore, the Local activity view may report more incoming traffic data than the Network activity view if any of the devices included in the visualization is acting as a server.

Related tasks

- Viewing network connections
- Viewing executions

Related references

- Server support

# Binary paths

## Overview

Nexthink stores the paths from where end-users execute each binary file of their applications, up to a maximum of 20 paths per binary. Binary paths are stored in lowercase letters (converting from uppercase when needed), and they use the forward slash (/) to separate the names of folders in the hierarchy, independently of the convention used by the underlying operating system of the devices.

Typical applications usually install their executable binary files in the same standard locations in the filesystem, independently of the device on which they are run. For example, most software applications are installed under the **Program Files** directory of a Windows device. The execution of binaries from multiple or unusual locations usually indicate a strange behaviour of users or even the presence of malware.

To avoid reporting too many paths for every single binary, Nexthink uses some techniques that are detailed below. Paths that do not fall into any of the special categories shown below are stored in their full form.

## Path aliases

Path aliases replace well-known directories by keywords, using a format similar to that of environment variables in Windows. In this way, binary paths of well-known locations become language neutral and independent of the drive in which the binary is located. For instance, the paths **D:\Program Files** (English version) and **C:\Programme** (German version) become both **%ProgramFiles%** when stored in Nexthink as a binary path.

Contrary to the general rule for binary paths, path aliases may contain uppercase characters. Find below a table with the list of all path aliases, their description, and a few exaples of the folders that they replace:

| Path alias | Description | Example |
|---|---|---|
| %Windows% | Windows directory | DRIVE:\Windows |
| %System% | Windows system directory | DRIVE:\Windows\System32 |
| %ProgramFiles% | Software installation directory | DRIVE:\Program Files<br>DRIVE:\Program Files (x86) |
| %UserProfile% | Directory holding user-specific data | DRIVE:\Documents and Settings\USERNAME<br>DRIVE:\Users\USERNAME |
| %AllProfile% | Directory holding data accessible by all users | DRIVE:\Document and Settings\All users<br>`DRIVE:\Users\Public (Windows Vista and higher)` |
| %ProfileTemp% | Directory holding user-specific temporary files. | DRIVE:\Documents and Settings\USERNAME\Local Settings<br>`DRIVE:\Users\USERNAME\AppData\Local` |
| %WindowsTemp% | Temporary folders in hexadecimal format under the root directory | DRIVE:\c7fa349ced49048e8941a819b264eb8d |
| %NetDrive% | Network shared folder | \\SERVER\shared-dir |
| %RemovableDrive% | Non-permanent storage devices | MEDIA_DRIVE:\ (USB stick, CD / DVD, etc.) |
| %RecycleBin% | Directory holding deleted files | DRIVE:\$RECYCLE.BIN |

## Ellipsis in binary paths

### *Ellipsis in aliased paths*

For privacy reasons and to avoid path explosion, the complete binary path is not recorded for binaries whose working path lies inside some of the aliased locations. Binaries executed from these locations do not have their full path stored:

- %RecylceBin%
- %UserProfile%
- %AllProfile%
- %ProfileTemp%
- %WindowsTemp%
- %RemovableDrive%

Instead, a three dot ellipsis (`/.../`) replaces the part of the path after the alias. For example, the path of a typical binary installer `setup.exe` executed from a temporary Windows folder is recorded as:

```
%WindowsTemp%/.../setup.exe
```

### *Ellipsis for automatically generated folders*

Nexthink is also capable of detecting folders whose names are automatically generated identifiers. These are usually very long alphanumerical names that are meaningless to a human reader. Therefore, the name of those folders is not stored *as is* in binary paths, but replaced by an ellipsis (`/.../`).

The following table contains the types of identifiers recognized by Nexthink and some examples of how each one of them looks like in the filesystem:

| Type of ID | Examples |
|---|---|
| GUID | 4AQlP4lP0xGaDAMF6CwzAQ<br><br>3F2504E0-4F89-11D3-9A0C-0305E82C3301 |
| MD5 | 79054025255fb1a26e4bc422aef54eb4 |
| SHA1 | 2fd4e1c67a2d28fced849ee1bb76e7391b93eb12 |
| Long Hexadecimal strings | Most hexadecimal strings containing 10 or more characters |
| Long numbers | |

| | |
|---|---|
| | Most strings containing at least 10 digits in a row, except if the digits are all the same. |

Related concepts

- Binary

# Maximum number of Binaries

By default, the Engine database supports a maximum of 40 000 simultaneous binaries. When this limit is reached, the Engine sends a system alert to the administrator account. No new binaries are incorporated into the Engine until any of the existing binaries is removed from the system.

The Engine automatically removes a binary from its database when the maximum inactivity period of the binary has elapsed and there is no event related to the binary in the database. By default, the maximum inactivity period of a binary is one month (30 days to be precise).

The Engine does not keep track of any information related to a non-recorded binary, such as executions or connections. Therefore, it is important for the administrator to keep the total number of binaries under control. If necessary, an administrator can increase the maximum number of binaries in an Engine and modify their maximum inactivity period. Beware that modifying the default values may have an impact on the performance of the whole system.

## Modifying the limits on binaries

To modify the maximum number of binaries and their maximum inactivity period:

1. Log in to the CLI of the appliance hosting the Engine.
2. Edit the configuration file of the Engine:
   ```
   sudo vi /var/nexthink/engine/01/etc/nxengine.xml
   ```
3. Inside the **<limit>** tag, add the following lines to increase the maximum number of binaries to 60 000 and decrease the maximum inactivity period of binaries to 20 days, for instance:
   ```
   <max_binaries>60000</max_binaries>
   <max_binary_inactivity_period>1728000</max_binary_inactivity_period>
   ```
4. To save the file and exit, type in:
   ```
   :wq
   ```
5. Restart the Engine:
   ```
   sudo systemctl restart nxengine@1
   ```

Note that the maximum number of binaries has a hard limit of **100  000** binaries and that the maximum inactivity period of binaries is expressed in seconds:

```
20 days * 24 hours/day * 60 min/hour * 60 s/min = 1 728 000 s
```

The operations described in this article should only be performed by a Nexthink Engineer or a Nexthink Certified Partner.

If you need help or assistance, please contact your Nexthink Certified Partner.
Related concepts

- Binary

# Package Executable Mapping

Finding out which package an executable belongs to is not an trivial task and is not 100% accurate, an executable may even belong to no package. To do so, use the heuristic described below.

Let's define an executable as the tuple path, hash and name/size i.e. [PATH,HASH,FILE].

An MSI package contains both an installation and uninstallation scripts linked to embedded resources, usually binaries. Once installed, an MSI is stored on the machine but its resources are striped out to save disk space. However most embedded binaries are listed either by name or by size. In addition, an MSI defines an installation directory.

So for each MSI we have the tuple [{HASH},{FILE},DIR] even if some installed binaries may not be present neither {HASH} nor {FILE}.

Other type of packages are treated as black box and we take only the installation directory if present or by the path of its uninstallation program if not. so we have the tuple [{},{},DIR].

An executable [PATH,HASH,FILE] is associated to a package [{HASH},{FILE},{DIR}] whenever one of those conditions is met:

- HASH is contained in {HASH}

- DIR is equal to {DIR} *
- DIR parent is equal to {DIR} *
- FILE is contained in {FILE}

If no specific package can be associated to a executable, it is associated to the default "unknown" package.

The following directories are excluded:

- WINDOWS e.g. C:\WINDOWS
- SYSTEM e.g. C:\WINDOWS\system32
- PROGRAM_FILES_COMMON e.g. C:\Program Files\Common Files\Common Files
- PROGRAM_FILES e.g. C:\Program Files\Common Files
- COMMON_STARTMENU e.g. C:\Documents and Settings\LeeT\Start Menu
- COMMON_PROGRAMS e.g. C:\Documents and Settings\LeeT\Start Menu\Programs
- COMMON_STARTUP e.g. C:\Documents and Settings\gjaunin\Start Menu\Programs\Startup
- COMMON_MUSIC e.g. C:\Documents and Settings\LeeT\Favorites
- COMMON_FAVORITES e.g. C:\Documents and Settings\LeeT\Favorites
- COMMON_DOCUMENTS e.g. C:\Documents and Settings\LeeT\My Documents
- COMMON_DESKTOPDIRECTORY e.g. C:\Documents and Settings\LeeT\Desktop
- COMMON_APPDATA e.g. C:\Documents and Settings\LeeT\Application Data

# Information on printers and printing

Getting accurate information on the utilization of printers is essential to ensure compliance with the printing policies established inside your organization and to optimize print costs. The Engine records every printing activity of the end-users that is initiated from any device in which the Collector is running.

Because of the technologies involved in the detection of print jobs, only Windows devices are able to send printing information for the moment.

Starting from V6.18, print monitoring is disabled by default. To receive information on printers and printing, enable the functionality in the Collector either during installation or via the Collector Configuration Tool.

Applies to platforms:

## Printer information

The Engine knows about a printer in your organization once a device that is equipped with the Collector tries to print a document on it. Depending on how the printer is connected to the device, Nexthink distinguishes four types of printers:

local
: The printer is directly connected to the device via a serial or parallel port (USB, COM or LPT) and it is visible to all the users of the device under the same name. Virtual printers, that is, software drivers that behave like a printer driver but lack the physical apparatus and typically redirect their output to a file, also fall into the category of local printers.

tcp/ip
: The printer is connected to the network and it is made available to the device via a standard TCP/IP port. All users of the same device see the printer with the same name or IP address.

wsd
: The printer is connected to the network and it is made available to the device by means of a Web Services for Devices port. All users of the same device see the printer with the same name.

smb
: The printer is made available to the device by sharing it from another device where the printer is locally connected via the SMB protocol. The printer is therefore considered as local in the hosting device, and as an SMB printer in the remote device. Each user in the remote device must individually import the SMB printer to be able to use it, so users may see the printer under different names. Note that SMB printer support is disabled by default in Nexthink.

Besides the type, Nexthink records the following information on printers:

Name
: The name of the printer as it appears in the properties dialog.

Hostname
: For local and smb printers, this is the name of the device to which the printer is directly connected.
For tpc/ip and wsd printers, this is usually the DNS name or IP address of the printer itself.

Display name
: Since different users may see the same of printer under different names, the *display name* shows the most frequent name assigned to the printer.

Location
: The place where the printer is found, according to its configuration properties.

***Limit on the number of printers***

The Collector supports up to 62 printers connected to a single device. A Collector that runs on a device with more than 62 connected printers fails to report any print job and, consequently, any printer.

## Print job information

A print job is an activity that puts in relationship a user, a device and a printer. Thus, for a given print job, you can display in the Finder the name of the device that sent the print job, its ID, or its SID, without the need to drill-down to devices. Likewise, you can display the name of the user, its ID, its SID, or the name and model of the printer that took part in the print job.

At the end of the printing process, the print job is added to the Engine with one of the following status:

success
: The print job has been successfully completed.

error
: The print job was not completed because of an error.

unknown
: The Collector could not determine the final status of the print job.

The rest of the fields of the print job reflect the options selected to configure the printer: **Number of pages**, **Paper size**, **Duplex print**, **Color enabled**, **Print quality** and **Size**.

Beware that software may modify the actual printout despite of the settings sent to the printer. In particular, the fact that the **Color enabled** field of the print job is reported as **yes** does not necessarily mean that the output is in color. For example, a user of PowerPoint who decides to print a colorful presentation in black and white may select the **Grayscale** option in the printing dialog of PowerPoint:

In this case, even if the user sets up a color printer to print in color and the Finder reports the **Color enabled** field as **yes**, the actual output is obviously in black and white. As a side note, depending on the printer model, enabling color output can still cause consumption of color ink to get a grey or black tone.

On the other hand, if the user sets up the printer itself to print in black and white, the resulting printout can never be in color and the Finder reports **Color enabled** as **no**.

Something similar happens with applications that are able to group several document pages into one before sending the document to the printer. If an external application does the grouping, the printer knows nothing about it. Therefore, the printer reports the **Number of pages** that it actually printed. When it is the printer that makes the grouping, the **Number of pages** depend on the

model of the printer. Some printers give the number of pages actually printed and some others take into account the grouping and give the number of pages of the original document.

For some print jobs, the Collector is also able to determine the **Document type**, which reflects the file format of the printed document or the application that produced it. Find the list of supported document types below:

| | | |
|---|---|---|
| • Crystal | • Outlook | • Publisher |
| • Excel | • PDF | • Text |
| • Illustrator | • Photoshop | • Visio |
| • Image | • Postscript | • Word |
| • InDesign | • Powerpoint | • Web |
| • InfoPath | • Project | • unknown |

## Print information in virtual environments

The basic principles of print support remain the same for virtual environments. For both VDI and streamed applications, the same information detailed above on printers and print jobs applies. In addition to the standard printing techniques, however, virtual environments introduce the concept of *redirected printers*. Consider for instance a user connecting to a virtual machine from a client device. If the client device is equipped with a local printer, the user can map the printer in the remote session. Thus, printing in the remote computer using the mapped printer effectively redirects the print jobs to the local printer.

In the example above, all the printers that display **(de NXT-PAT-W7)** at the end of their name are redirected to the local printer in the client device **NXT-PAT-W7**.

If the Collector is running on the client device and the user prints on one of these redirected printers, the print job is reported as originating from **NXT-PAT-W7** and not from the virtual machine. This indeed corresponds to reality, since the printer is merely redirected, and not connected to the VM.

On the other hand, if the Collector is also installed in the VM, any print job sent to the redirected printer is reported to the Engine as originating from the VM as well. In this case, even the redirected printer is reported as local to the VM too. By default, these virtual printers and print jobs are discarded by the Engine to avoid duplications that might alter overall statistics. You can nevertheless control what kinds of print jobs are discarded by editing the configuration file of the Engine.

## Print notifications of SMB printers

Devices that do not belong to a domain may fail to receive print notifications from SMB printers in the case that an anonymous user initiated the print job. To receive print notifications of anonymous users from SMB printers, ensure that the registry key **NullSessionPipes** holds the value **spoolss**:

1. Open the registry editor by pressing the **Windows + R** keys and typing **regedit**.
2. Locate the registry key **NullSessionPipes** here:
    - HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\LanmanServer\Para
3. Edit its value and set it to **spoolss**.

An alternative solution is again to join all your devices to a domain. In a domain, all users are authenticated.

Related tasks

- Enabling printing support
- Ignoring specific print ports
- Enabling support for SMB printers

Related references

- Nxtcfg - Collector configuration tool

Related concepts

- Printer
- Print job

# Metro apps

Windows 8 Metro apps behave in a different way than normal Windows applications. Nexthink abstracts those differences in order to provide a comprehensive view of the users' activity.

## Executions of metro apps

There are mainly two kind of architectures for metro apps: compiled and JavaScript. Compiled applications (written in C#, VisualBasic or C++) behave similarly to normal Windows applications and have a .exe file associated with them. JavaScript applications however are not compiled and there is no associated .exe file. Instead the default windows executable WWAHost.exe is used to host them. When analysing the task manager of Windows 8 device we can usually see several executions of WWAHost, each one corresponding to a specific metro app.

Only reporting executions of WWAHost would be confusing as Nexthink users would no longer be able to distinguish between different metro apps. Instead Nexthink Collector abstracts this information and reconstructs name and properties of the metro app being hosted by WWAHost by reporting this as a **Binary**. Similar techniques are used to abstract the corresponding **Executables** and **Applications**.

Executions, connections, web requests, crashes, freezes, etc. are reported as usual. This means that for instance in Nexthink it is possible to distinguish the web requests performed by the *Weather* app with respect to the *Food & Drink* app. The **Description** field of Metro executables is also always set to *Windows 8 Metro Style App*.

## Metro packages

Metro apps are different from a packaging perspective as well. In fact they are not installed like traditional applications and are not visible in Windows **Programs and Features**. As for binaries Nexthink abstracts these differences; all metro apps are reported as a **Package** and the corresponding installations and uninstallations are reported as well.

Related concepts

- Metro app architecture (MSDN)

# Mobile data and ActiveSync

This article or section is in the process of an expansion or major restructuring.

# Investigation with packages

Starting from Nexthink V6.3, those investigations retrieving packages or including a condition on packages have been simplified. The results take into account only those packages that are effectively installed, discarding uninstalled packages.

Nevertheless, it is still possible to find installations or uninstallations of packages by directly querying the installation events.

The following tables summarize the typical uses of investigations related to packages. Download all the investigations and import them into the Finder to try them out.

## Retrieving devices

| Investigation name | Condition | Aggregate condition |
|---|---|---|
| Devices *with* a specific package | Package Name is "abc" | - |
| Devices *without* a specific package | Package Name is "abc" | Number of packages = 0 |
| Devices *with* two specific packages | Package Name is "abc", publisher is "ABC", version is "123", type "program"<br><br>or<br><br>Package Name is "xyz", publisher is "XYZ", version is "789", type "program"<br><br>NOTE: A package is uniquely identified by its name, publisher, version and type. | Number of packages = 2 |

### Retrieving installation events

| Investigation name | Condition | Aggregate condition |
|---|---|---|
| Installation of a specific package | Package Name is "abc" and Operation Type is "installation" | - |
| Uninstallation of a specific package | Package Name is "abc" and Operation Type is "uninstallation" | - |

### Retrieving packages

| Investigation name | Condition | Aggregate condition | Order by |
|---|---|---|---|
| Least used packages | - | - | Number of devices is ascending |
| Most used packages | - | - | Number of devices is descending |
| Packages installed on less than 5 devices (but at least installed on one device) | - | Number of devices is less or equal to 5 | - |

# Portal aggregation and grouping

## Overview

The Portal aggregates metric data along two dimensions: the defined hierarchy and time. To aggregate the data, the Portal also takes into account the *group by* and *aggregate by* options that you set when you create the metric in the Finder (depending on the kind of metric, not all options are available). Learn here how the possible combinations of aggregation and grouping options, as well as the hierarchy node and period selection, produce different results in the dashboards of the Portal.

## The example hierarchy

For demonstration purposes, let us suppose that we have defined a very simple hierarchy based on locations that consists of two levels only:

- Country (top level)
- City (entity level)

Our imaginary organization has offices in two countries: Switzerland (CH) and Spain (ES). For each country, offices are located in two cities: Geneva (GVA) and Zurich (ZRH), in the case of Switzerland; Madrid (MAD) and Barcelone (BCN), in the case of Spain.

Finally, for simplicity, let us suppose that there are just two devices per city, whose names are composed by the initial letter of the city, followed by either the number 1 or the number 2. That makes a total of eight devices:

| Hierarchy | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Country** | CH | | | | ES | | | |
| **City (entity)** | GVA | | ZRH | | MAD | | BCN | |
| **Devices** | G1 | G2 | Z1 | Z2 | M1 | M2 | B1 | B2 |

## Count metrics

Depending on the type of object, count metrics may take into account, for a particular day, only those objects that were active during that day or all the objects regardless of their activity. For some types of objects, you can only count active instances. For packages, you always count all of them. For users and devices, the user may choose whether to count all of them or only those that were active. See the table below as reference:

| Only active | Only all objects | User choice |
|---|---|---|
| • Application<br>• Executable<br>• Binary<br>• Port<br>• Destination<br>• Domain<br>• Printer | • Packages | • Users<br>• Devices |

Except in the case of packages, you are usually interested in counting only those objects which were active during the day. However, in cases such as transformation projects, it is important to know which objects already fulfil the transformation condition, no matter whether they were active during a particular day or not. For instance, if you have a migration project from Windows 8 to Windows 10, you probably want to count every day the devices which have already installed Windows 10, even when they were not active during the

measurement day. In this way, when observing the results of the metric in the Portal, you get a non-decreasing value, which may not be the case if you measure only active devices instead. For instance, in a typical company, the number of active devices decreases dramatically during the weekend, increasing again during weed days.

The value of a metric that counts all objects is thus valid for the particular day when it is computed and cannot be aggregated through time. Therefore, the Portal does not show widgets associated to these metrics when selecting time periods different from one day. In addition, note that the computation of these metrics requires to look through all the history available in the Engine for counting in all possible objects. The conditions that you can specify for these metrics are consequently the same as for full period investigations, namely they cannot depend on activities or events.

When counting active objects only, an extra condition (*count devices that meet conditions on ... in period*) influences how the Portal displays the results of the metrics for periods longer than one day. Use this extra condition to count objects, either:

- *The last day* that the objects were active within the selected period (for inventory purposes), or
- *At least one day* during the selected period (for detecting occurrences).

***Grouping by properties***

As a first example, let us consider a metric that counts the number of devices and groups them both by type and OS (in the **COMPUTE DAILY** section, select **Group by *device type* and *OS version and architecture***).

Since this metric is intended to make an inventory of devices and inventories are usually based on the latest state of the catalogued objets, select the extra condition of count metrics to **last active day** (in the **MATCHING** section, select **Count devices that meet conditions on *last active day* in period**). Leave the rest of the options for creating the metric to their defaults.

The devices in our example are of the following types and have the following operating systems installed:

| Devices | Device Type | OS version and architecture |
|---|---|---|
| G1 | server | Windows 2012 Server Standard (64 bits) |
| G2 | desktop | Windows 8.1 Pro (64 bits) |

| | | |
|---|---|---|
| Z1 | laptop | Windows 7 Enterprise SP1 (64 bits) |
| Z2 | desktop | Windows 8.1 Pro (64 bits) |
| M1 | laptop | Windows 10 Pro (64 bits) |
| M2 | desktop | Windows 7 Enterprise SP1 (64 bits) |
| B1 | laptop | OS X Yosemite 10.10.5 (64 bits) |
| B2 | desktop | Windows 7 Enterprise SP1 (64 bits) |

If the devices and their operating system do not change, the Portal consistently displays the same global results for the metric, regardless of the period selected. For instance, when you display the count metric in a table widget, arranging the rows by operating system and the columns by device type, you always get the following global results:

| | Device type | | |
|---|---|---|---|
| **OS version and architecture** | **desktop** | **laptop** | **server** |
| Windows 8.1 Pro (64 bits) | 2 | 0 | 0 |
| Windows 7 Enterprise SP1 (64 bits) | 2 | 1 | 0 |
| Windows 2012 Server Standard (64 bits) | 0 | 0 | 1 |
| Windows 10 Pro (64 bits) | 0 | 1 | 0 |
| OS X Yosemite 10.10.5 (64 bits) | 0 | 1 | 0 |

Now let us consider the case that B1 (the only Mac OS device in the list) gets its OS upgraded from version *Yosemite* to *El Capitan*. After this upgrade, the choice of the extra condition of count metrics influences the results displayed in the Portal. Differences appear only when viewing periods are longer than one day in the Portal; therefore, let us see the results of our widget when selecting a period of one week (the week when the device got its OS upgraded). If you chose **last active day** as extra condition, you get:

| | Device type | | |
|---|---|---|---|
| **OS version and architecture** | **desktop** | **laptop** | **server** |
| Windows 8.1 Pro (64 bits) | 2 | 0 | 0 |
| Windows 7 Enterprise SP1 (64 bits) | 2 | 1 | 0 |
| Windows 2012 Server Standard (64 bits) | 0 | 0 | 1 |
| Windows 10 Pro (64 bits) | 0 | 1 | 0 |
| OS X El Capitan 10.11.4 (64 bits) | 0 | 1 | 0 |

That is, you get the same results as before, except for the last row in the list, where **OS X Yosemite 10.10.5 (64 bits)** is replaced by **OS X El Capitan 10.11.4 (64 bits)**. That implies that the last day when the Mac device was active during

the selected week, it already had the *El Capitan* version installed. However, if the metric was defined with the extra condition option set to **at least one day** instead of **the last active day**, the widget displays the following results:

| | Device type | | |
|---|---|---|---|
| **OS version and architecture** | **desktop** | **laptop** | **server** |
| Windows 8.1 Pro (64 bits) | 2 | 0 | 0 |
| Windows 7 Enterprise SP1 (64 bits) | 2 | 1 | 0 |
| Windows 2012 Server Standard (64 bits) | 0 | 0 | 1 |
| Windows 10 Pro (64 bits) | 0 | 1 | 0 |
| OS X Yosemite 10.10.5 (64 bits) | 0 | 1 | 0 |
| OS X El Capitan 10.11.4 (64 bits) | 0 | 1 | 0 |

That is, during the week of the upgrade, the Mac computer was seen some days with the *Yosemite* version installed and some other days with the *El Capitan* version. Thus, it appears twice in the table widget. For that reason, the option **at least one day** is not well-adapted to inventorying, which was the original purpose of the metric.

On the other hand, the option **at least one day** is useful to detect occurrences of particular situations. For example, imagine that you want to know whether any device had the antivirus real-time protection turned off any day during the selected period. To that end, create a metric that counts devices and performs any of the following two functions:

- Group the devices by the status of their antivirus real-time protection
- Set a condition to get only those devices with the protection turned off.

Choose **at least one day** as the extra condition of the metric to count those devices that had the protection turned off at any day and not necessarily the last day that they were active.

Note that the Portal verifies the state of the antivirus real-time protection of the devices when it computes the value for the metric. If you switch the antivirus real-time protection of a device off and on in the same day before Portal computation, the situation will go undetected for the previously defined metric. In general, this applies to all metrics that set conditions on the state of objects. On the other hand, metrics with conditions based on events keep a history of occurrences. For instance, if a metric counts the number of devices executing high threat binaries, the Portal will see these executions during the computation of the metric in any case.

Coming back to our inventory example, let us consider now the result of counting all devices instead of counting the active devices only. Since the migration from Yosemite to El Capitan looks like a transformation project, counting all objects is probably more suitable than counting just the active objects. Indeed, imagine that the Mac laptop in the example is turned off for a whole day after being upgraded to El Capitan. While a metric that counts all objects would include the Mac laptop in the results of that day, a metric counting only active devices would leave it out of its results. Think carefully about the choice of counting only active objects or all objects when you create a count metric for devices or users.

### *Grouping by foreign category*

Count metrics let you group results not only by the properties of the counted objects themselves, but also by categories of related objects (*foreign* categories). Let us illustrate this kind of grouping by creating a metric that counts the number of users. The metric groups the users by a foreign category called *Ownership*. We define Ownership as a category of devices that has two keywords:

- **Corporate**, which indicates that a device belongs to the company.
- **BYOD**, which indicates that a device belongs to the user himself.

From the example hierarchy, let us focus on the CH node, that is, the devices in Switzerland, and imagine that we had the following usage pattern during the last day:

- User 1: used his own computer G1 in Geneva and then travelled to Zurich and used the corporate computer Z1.
- User 2: used his own computer G2 in Geneva.
- User 3: used the corporate computer Z2 in Zurich.

When retrieving the data during the nightly computation, the Portal stores the following data internally:

| Users | Entity | Ownership |
|---|---|---|
| User 1 | GVA, ZRH | BYOD, Corporate |
| User 2 | GVA | BYOD |
| User 3 | ZRH | Corporate |

Note that the Portal is unable to deduce from this table whether the Corporate device of User 1 is located in Geneva or Zurich (same for his BYOD device). To be on the safe side, the convention followed by the Portal is to count the user in for all possible combinations, although that may lead to situations where the actual combination did not occur. In our example for User 1, only the

combinations GVA-BYOD (because of the use of G1) and ZRH-Corporate (because of the use of Z1) actually occur. However, the Portal counts as well User 1 for the combinations GVA-Corporate and ZRH-BYOD.

Thus, displaying the metric in a table widget, with the rows organized by hierarchy and the columns by Device-Ownership, yields the following results for the last day (when CH is selected as the Country of the hierarchy):

| | Ownnership | | |
|---|---|---|---|
| **City** | **Overall** | **Corporate** | **BYOD** |
| Overall | 3 | 2 | 2 |
| GVA | 2 | 1 | 2 |
| ZRH | 2 | 2 | 1 |

Note that because of User 1 using computers in both Geneva and Zurich, and because of the additional combinations added by convention, none of the partial results add up to the overall values. The values over all the hierarchies are shown in the Table widget after ticking the option **Display overall value** in the configuration fof the widget for dividing by hierarchy.

***Considerations on ratios and thresholds***

When creating a count metric that includes a ratio computation and thresholds based on the ratio, threshold violations occur more frequently when exploring the lower levels of the hierarchy. The reason is that ratios tend to get more extreme when there are less objects to count. To avoid triggering threshold violations with only a few objects involved, tick the checkbox **Ignore unless at least x objects are impacted in a hierarchy node** and specify the minimum number of objects that must be impacted in order to consider a threshold violation effective.

On the other hand, if you base the thresholds on absolute values instead of ratios, threshold violations occur more frequently in higher levels of the hierarchy, because they involve more objects.

## Quantity metrics

There are two types of quantity metrics: those that measure a countable number of actions (such as the number of executions, the number of printed pages, etc.) and those that measure a continous quantity related to the activity of a device (the memory usage, the boot or logon duration, etc.). In both types of quantity metrics, the measured quantity changes over time. It is therefore necessary to aggregate all the collected values to display a final result in the Portal for each

available time frame. For quantity metrics, there are four ways of aggregating the results:

- **sum over all devices and the whole timeframe**
- **average value per device per day**
- **maximum value per device per day**
- **minimum value per device per day**

Not all of the aggregation strategies are available for all quantity metrics. Only the options that make sense for a particular metric can be selected. Typically, the computation of the average and the maximum values per device per day are available to any quantity metric, whereas the sum or the minimum values only make sense for some kinds of quantity metrics.

Let us examine the different aggregation options for quantity metrics through some examples. For instance, consider a metric that counts the number of executions of the Finder -your favorite real-time analysis application from Nexthink- on each device; that is, a quantity metric that computes the **number of executions** with a condition on the executable name *nxfinder.exe*. Suppose that, for the current week, the devices of our original example located in Spain have the following usage pattern:

| | | Finder executions | |
|---|---|---|---|
| **City** | **Device** | **Monday** | **Tuesday** |
| **MAD** | **M1** | 4 | 2 |
| | **M2** | 1 | 1 |
| **BCN** | **B1** | 0 | 1 |
| | **B2** | 1 | 0 |

Consider first the results of the metric for the different aggregation strategies when looking at the last day:

**Maximum value per device per day: 2**
: The device that executed the Finder the most on Tuesday is M1, which did it twice.

**Sum over all devices and the whole timeframe: 4**
: Results from adding all the executions on Tuesday 2 (M1) + 1 (M2) + 1 (B1) = 4.

**Average value per device per day: 1.3**
: Results from dividing the sum of all executions (the previous value calculated) by the number of devices that participate in the metric 4 / 3 = 1.3. Note that B2 is not counted in for computing the average on Tuesday

because it does not satisfy the condition of executing *nxfinder.exe*.

Now let us consider the results when selecting the last week. Since the week is not over, the Portal has data only for Monday and Tuesday:

**Maximum value per device per day: 4**
Device M1 executed the Finder four times on Monday. That is the maximum for any device during the week.

**Sum over all devices the whole timeframe: 10**
That adds up for the four executions on Tuesday plus six on Monday.

**Average value per device per day: 1.7**
Results from dividing the sum of all executions over the whole week by the sum of devices participating in the metric each day. That is 10 / 6 = 1.7. Note again that neither B1 not counted in on Monday nor B2 is counted in on Tuesday because they do not satisfy the condition of executing *nxfinder.exe*, so we only have three devices each day, making a total of 6 devices for computing the average.

We have just seen an example of a quantity metric that counts individual occurrences. Let us now consider a metric based on continuous value; for instance, a metric that computes the **average memory usage per execution** of the Finder (as in the previous example, we use a condition on the executable *nxfinder.exe*). As you probably know, the memory used by the Finder increases with the number of tabs that are simultaneously open, so we can expect significant differences between the memory used by any two distinct executions of the Finder. Again, for the devices in Spain, suppose that we have the following usage pattern for the current week:

| | | Finder memory | |
|---|---|---|---|
| **City** | **Device** | **Monday** | **Tuesday** |
| **MAD** | **M1** | 200 MB | 100 MB |
| | **M2** | No execution | 200 MB |
| **BCN** | **B1** | 300 MB | 500 MB |
| | **B2** | 400 MB | 400 MB |

The metric yields the following results for Tuesday, according to the chosen aggregation strategy:

**Maximum value per device per day: 500 MB**
Device B1 had the highest memory usage on Tuesday. Note that this is not the absolute maximum value of memory used by the Finder in the device, since the metric is actually measuring the average along the day.

**Minimum value per device per day: 100 MB**

Device M1 had the lowest memory usage on Tuesday. Again, note that this is an average along the day, so it is not the absolute minimum amount of memory that the Finder used.

**Average value per device per day: 300 MB**

Adding the values for all devices makes a total of 100 (M1) + 200 (M2) + 500 (B1) + 400 (B2) = 1200 MB, dividing by four devices gives 300 MB in average.

If you select to see the last week results in the Portal, you get the following results:

**Maximum value per device per day: 500 MB**

No device used more memory for the Finder than B1 on Tuesday, so 500 MB is again the maximum for the week.

**Minimum value per device per day: 100 MB**

The same for the minimum usage of mermory, which corresponds to M1 on Tuesday. Note that the memory usage of M2 on Monday is not counted as 0 because it does not meet the condition of executing the Finder. Thus, the metric does not take M2 into account.

**Average value per device per day: 300 MB**

We have 1200 MB from Tuesday plus a total of 200 (M1) + 300 (B1) + 400 (B2) = 900 MB on Monday, making a grand total of 2100 MB for the week. Dividing by three active devices on Monday (M2 had no executions) plus four devices on Tuesday, that is, a total of seven devices, gives 2100 / 7 = 300 MB in average.

In quantity metrics, as in count metrics, you can group the results by up to two criteria. Since quantity metrics are related to devices only, the criteria for grouping results can be based either on the attributes or on categories of devices (no foreign categories are available in this case). Setting **group by** options lets you break down the results of the metric when they are displayed in a table widget within a dashboard, in the same way as shown for count metrics.

## Top metrics

Top metrics return a list of objects ordered by their contribution to a particular activity. When defining a top metric, you choose:

- The type of object to show in the list.
- The total number of objects in the list (from the top 10 to the top 100).
- An activity that is linked to that type of object.

- The criterion for including the objects in the list: include those with either the highest or the lowest contribution to the activity.
- The *aggregate by* option for the activity, which determines how to compute the contribution of each object.

The available aggregation options are similar to those available in quantity metrics. The difference is that they are not necessarily based on devices, so they can cross device boundaries. For instance, a metric that computes the *top 10 users with the highest number of executions* aggregates into a single value the number of executions carried out by a same user in different devices. Thus, these are the **Aggregate by** options of top metrics:

- **sum over the whole timeframe**
- **average value per day**
- **maximum value per day**
- **minimum value per day**

As in the case of quantity metrics, not all the aggregation possibilities are available for all top metrics, but only those that make sense.

## Grouping by hierarchy

In addition to the **group by** options that you specify for count and quantity metrics, the Portal always aggregates the results of all metrics (including top metrics) by hierarchy.

When selecting one **group by** option in the definition of a count or quantity metric, it is always possible to break down the results by hierarchy and by grouping option, arranging them as the rows and columns (or columns and rows) of a Table widget. If you define two **group by** options and you select one of them to break down the results by rows or columns in a Table widget, you must select the other option for the columns or rows, respectively. In this case, the option to break down by hierarchy is not available. Nevertheless, choosing to display the total value of the metric or metrics added to the widget (i.e. the **Metrics** option), instead one of the *group by* criteria, lets you again choose the hierarchical criterion to arrange the results.

In top metrics, however, there is no **group by** option and it is not possible to arrange the hierarchy in a Table widget. You can still add display fields to the definition of a top metric to see them arranged in a Table widget.

### *Hierarchy navigation in the Portal*

For those widgets that do not explicitly break down the results of a metric by hierarchy (KPI, Line charts, and those Table widgets not arranged by hierarchy), use the hierarchy navigation tool of the Portal that is located in the top blue ribbon to explore the results at different levels in the hierarchy. All the widgets in the dashboard adapt their results to the node selected in the hierarchy navigation tool.

The Table widgets that do break down the results by hierarchy divide in fact their results by the nodes placed at the immediately lower level of the node selected in the hierarchy navigation tool (that is, its child nodes).

For instance, consider a dashboard built around the hierarchy of our original exaple. While viewing the dashboard in the Portal, if you switch from the Global view to the node CH in the hierarchy navigation tool, all the widgets in the dashboard will display the results limited to the values that they got for Switzerland. In addition, Table widgets arranged by hierarchy will divide their results by the nodes GVA and ZRH, which are the children of the CH node.

## Considerations on aggregation along time

As time passes by, the Portal accumulates data day by day. Every night, the Portal collects and computes the values of the past day and aggregates the results to the current week, month, and quarter.

However, when switching from a view of the current week to the current month in the Portal, a counterintuitive situation may occur: the amount of data available for the current month might be less than the amount of data for the current week. This situation may occur when a month boundary has recently been crossed, because a week can overlap over two different months.

Let us explain it with one of our previous examples on quantity metrics. Consider again the metric that counts the number of executions of the Finder and suppose that we chose to aggregate by the **sum over all devices and the whole timeframe**. If you look at the values that we obtained, we had 6 executions of the Finder on Monday and 4 executions on Tuesday. Assume now that Monday was April 30, Tuesday was May 1, and it is May 2. Therefore, a month boundary was crossed yesterday.

| Date | April 30 | May 1 | May 2 |
|---|---|---|---|
| Week day | Monday | Tuesday | Wednesday |

| | | | |
|---|---|---|---|
| **Executions** | 6 | 4 | N/A |

If we navigate to the most recent date available in the Portal, the result depends on the period selected:

- **Day:** 4 executions, corresponding to Tuesday, May 1.
- **Week:** 10 executions, corresponding to the 6 on Monday plus the 4 on Tuesday.
- **Month:** 4 executions, corresponding to Tuesday, May 1.

That is, the week started before the new month and it is still ongoing, so there are more days available for the current week than for the current month. The value for the week may therefore be bigger than the value for the month.

Related tasks

- Creating a metric

# Security

## Access rights and permissions

### Overview

Nexthink users have the right to see and manage content depending on their profile and assigned roles. The definition of a profile includes the account type, view domains, mandatory roles, and other settings that determine the permissions of the users for managing content and performing system administration tasks.

The following tables display the access rights of the different types of users to the features of the product, including all the additional requirements to their profile or roles -when needed.

### System management

| Feature | Main administrator | Central administrator | User |
|---|---|---|---|
| Manage accounts | Ok | Ok | No |
| Manage profiles | Ok | Ok | No |
| Manage roles | Ok | Ok | No |
| Manage hierarchies | Ok | Ok | No |
| Manage entities | Ok | Ok | No |
| Manage engines | Ok | Ok | No |
| Manage appliance | Ok | Ok | No |
| Manage license | Ok | Ok | No |

### Portal content

| Feature | Main administrator | Central administrator | User |
|---|---|---|---|
| Create modules and dashboards | Ok | Ok | Profile |
| View published modules | Ok | Ok | Roles |
| Manage published modules | Ok | Ok | Non-admin |
| Manage service alerts | Ok | Ok | No |

Profile
: Normal users can create modules if the option **Allow creation of personal dashboards** is checked in the definition of their profile. Additionally, normal users can publish their modules if the option **Allow publication of modules** is checked in their profiles.

Roles
: Normal users can see the published modules included in their roles only.

Non-admin
: Normal users can only manage the modules that they can see and have been created by themselves or by other normal (non-admin) users.

## Finder and Engine content

| Feature | Main administrator | Central administrator | User |
|---|---|---|---|
| Access to the Finder | Ok | Profile1 | Profile1 |
| Manage categories, services, metrics, global alerts, import and export content | Ok | Profile2 | Profile2 |
| Manually tag objects | Ok | Profile3 | Profile3 |
| Web API (NXQL) | Ok | Profile4 | Profile4 |
| Management of Collector | Ok | Profile5 | Profile5 |
| Editing (and manual triggering) of campaigns | Ok | Profile6 | Profile6 |
| Editing of remote actions | Ok | Profile7 | Profile7 |
| Execution of remote actions | Ok | Profile8 | Profile8 |

Profile1
: The main administrator has the access to the Finder granted by default. Other users must have the option **Finder access** checked in the definition of their profile.

Profile2
: Users with data privacy disabled (**Data privacy** settings in the profile set to **none (full access)**) are able to manage categories, services, metrics, scores, global alerts, as well as import and export content and manually synchronize users and devices with AD, if they have the suboption **Allow system configuration** checked, in addition to the **Finder access** option, in the definition of their profile.

Profile3
: Users other than the main administrator can tag objects and edit applications if they have the suboption **Allow editing of applications and object tags** checked, in addition to the **Finder access** option, in the

definition of their profile.

Profile4
: Users other than the main administrator can access the Web API V2 (make requests to the Engine written in the NXQL language) if they have their **Data privacy** set to **none (full access)** and the option **Finder access** enabled in the definition of their profile.

Profile5
: Users other than the main administrator are able to supervise the installation of the Collector with the Updater from the Finder if they have the suboption **Allow management of Collectors** checked in their profile.

Profile6
: Users with data privacy disabled (**Data privacy** settings in the profile set to **none (full access)**) are able to edit and publish campaigns, if they have the suboption **Allow editing of campaigns** checked, in addition to the **Finder access** option, in the definition of their profile. For campaigns that target users manually, this profile enables the manual triggering of campaigns.

Profile7
: Users with data privacy disabled (**Data privacy** settings in the profile set to **none (full access)**) are able to edit remote actions, if they have the suboption **Allow editing of remote actions** checked, in addition to the **Finder access** option, in the definition of their profile.

Profile8
: Users with data privacy disabled (**Data privacy** settings in the profile set to **none (full access)**) are able to execute remote actions if, in addition to the **Finder access** option, they have either the suboption **Allow editing of remote actions** checked or the remote actions included as roles in the definition of their profile.

Related tasks

- Adding users

# Active Directory authentication

## Overview

Nexthink supports the authentication of users via Active Directory services. Microsoft Active Directory (AD) is currently used as the authoritative user directory in a vast number of organizations, controlling the authentication and the access rights of users.

The benefits of integrating Nexthink with the AD services include the following:

- Users need only one login and password (no need for a dedicated Nexthink account).
- Administrators can take advantage of the password policy defined in AD.

For a better user experience, Nexthink recommends to combine AD authentication with Windows authentication, so that users can log in to Nexthink without having to retype their Windows credentials.

## How authentication via AD works

To enable AD authentication in Nexthink, provide the user logon in the form `someone@example.com` when creating their account in Nexthink. Make sure that the AD account exists before adding it to Nexthink.

The user logon must be composed of the *sAMAccountName* of the user, followed by the *domain* or *realm*; both separated by the @ character. Note that the previous Windows logon format `DOMAIN\username` is not supported. Note as well that if a user got a *User Principal Name (UPN)* whose user logon name is different from the sAMAccountName, you still need to use the sAMAccountName when manually configuring the user's Nexthink account; otherwise, AD authentication will not work.

For example, if the sAMAccountName of user *John Wick* is `jwick` and he got assigned the user logon name (UPN prefix) `john.wick`, configure his Nexthink account with the first logon only:

- `jwick@example.com` - UPN prefix equal to the *sAMAccountName* - **Use this one**.
- `john.wick@example.com` - UPN prefix different from *sAMAccountName*.

To avoid the error prone manual creation of users and let users log in with their UPN (even if the UPN prefix does not match the sAMAccountName), Nexthink recommends the automatic provisioning of user accounts from Active Directory.

Beware that the account name part of the UPN is case sensitive. Thus, specify exactly the same name in Nexthink as it is registered in the AD, respecting the case. Nexthink uses the suffix part to resolve the name of the AD server (**example.com** in the example above), also known as the *domain controller*.

Once added to Nexthink, users can log in to the Finder or the Portal using their AD accounts (or Windows authentication, if enabled). During Finder login, the AD credentials provided by the user are forwarded to the Portal back-end using an encrypted channel. In the case of Portal login, the browser itself sends the AD credentials provided by the user to the Portal back-end. The Portal back-end is then responsible for contacting the AD server to authenticate the user.

## Requirements for AD authentication

### *Allowed characters in user names*

Use only printable characters in user names. The space and the following symbols are not allowed inside a user name:

```
/\[]:;|=,+*?<>@"&
```

### *Alternate UPN suffixes*

Users provisioned who have an alternate UPN suffix defined, can authenticate in Portal and Finder using both their alternate UPN suffix or the UPN with the fully qualified domain name. To do so, it is required to use a mapping file so that the system knows which alternate UPN suffix is mapped to which fully qualified domain name. A sample mapping file is located in `/var/nexthink/portal/conf/ad_upn_mappings.conf`, and has the following content:

```
acme.com = ch.acme.com
acme.com = fr.acme.com
```

In this sample, the alternate UPN suffix acme.com is mapped both to the standard UPN suffixes `ch.acme.com` and `fr.acme.com`.

The mapping file has the following rules:

- Lines starting with '#' are accepted as comments.
- Empty lines are accepted
- Mapping is defined in the form of `alternate_suffix=standard_suffix`.
    - Separating spaces between suffixes and '=' are accepted, for instance `alternate_suffix = standard_suffix`.
- '=' and space characters are not supported inside the suffixes:
    - `alternate suffix=standard_suffix` is invalid.
    - `alternate=suffix=standard_suffix` is also invalid.
- Duplicate mappings are forbidden (twice the exact same mapping).
- The maximum number of mappings is 50.
- After updating the mapping file the portal needs to be restarted.

Note that if two users have the exact same UPN prefix, they need to use their fully qualified domain names to login. For instance, if ACME has two fully qualified domain names `ch.acme.com` and `fr.acme.com`, an administrator may create an alternate UPN suffix acme.com that is easier to memorize and quicker to type. If there are two users whose UPN are `username@ch.acme.com` and `username@fr.acme.com`, they must login using these UPN. If they use the alternate UPN suffix and try to login using `username@acme.com`, it will not work and the following error message will be displayed: "Authentication failed! The credentials you entered are invalid or cannot be authenticated."

### *Connectivity with AD server*

For the Portal to be able to connect with the AD server, allow outgoing connections from the appliance on which the Portal runs through the following ports:

- UDP port 53, for DNS.
- TCP and UDP port 88, for Kerberos authentication on the AD server.
- TCP ports 389 and 636, for non-secure and secure LDAP connections to the AD server.

### *Time synchronization*

Because of the technique used for authenticating users, the Portal must be synchronized with the clock of the AD server. The configuration of the AD server may nevertheless specify a tolerance regarding clock discrepancy. A difference of at most 5 minutes is generally accepted by default.

***Encryption methods***

Nexthink supports the following encryption methods:

- AES (128 bits)
- RC4-HMAC

On the other hand, DES encryption (legacy for Windows 98) is not supported.

## Finder session saving

The Finder can save sessions and user credentials. This applies to AD credentials as well. If the user chooses to additionally save the password, then the Finder stores only a hash of the password for security reasons.

Related tasks

- Adding users
- Enabling Windows authentication of users
- Provisioning user accounts from Active Directory
- Preventing password saving in the Finder

Related references

- Connectivity requirements

# Canonical domain names for Windows authentication

## Overview

Thanks to Windows authentication (SSO), Nexthink users can conveniently log in either to the Portal or to the Finder without the need to type in their credentials each time.

When configuring Windows authentication for Nexthink, ensure that you set the canonical name of your domain (and not an alias) as the Service Principal Name which associates the service instance to the service logon account of Nexthink (*nxtPortalSSO*). Failing to do so results in users not able to log in through Windows authentication.

### DNS zones and resource records

DNS zones map domain names to IP addresses and other resources. Each resource record in a DNS zone defines a single mapping. We focus our attention on two types of records:

- Type **A** records, which map a domain name to an IP address.
- Type **CNAME** records, which define a domain name that is an alias for another domain name (the *canonical* name).

Let us consider an example of a DNS zone with two resource records. It is a Forward Lookup Zone whose name is `example.com`, which is the suffix of all the hosts in the zone. The DNS snap-in of the **Administrative Tools** of Windows Server shows the resource records as follows:

| Name | Type | Data |
|---|---|---|
| portal | Host (A) | 192.168.1.100 |
| myportal | Alias (CNAME) | portal.example.com. |

The first resource record in the zone, **portal**, is a record of type A. The record maps the name `portal.example.com` to the IP address 192.168.1.100. If this host provides the authentication service, `portal.example.com` is the canonical name that you must set as Service Principal Name to properly configure Windows authentication for Nexthink.

The second resource record in the zone, **myportal**, is a record of type CNAME. The record defines an alias for `portal.example.com` that is called `myportal.example.com`. While the alias `myportal` can replace the canonical name `portal` in many contexts, it is **not** suitable for configuring Windows authentication in Nexthink.

Related tasks

- Enabling Windows authentication of users

# System alerts

System alerts inform you of a special circumstance during system operation. There are four types of system alerts. For each type of system alert, find below its associated warning messages along with the description of the situation that is at the origin of the alert.

## License alerts (Central License Manager)

The Central License Manager sends notifications related to the status of the product license.

The recipients of the notification are:

- To: Customer contact
- Cc: Partner contact
- Cc: Nexthink sales contact

And the following types of notification exist:

### *Activation key*

When a new license is created on the Central License Manager, an activation key is sent. This activation key is used to activate the product.

### *Modification of the license*

When the license is modified in the Central License Management, an automatic notification is sent.

For **online license**, the modification will be automatically applied after a maximum of 6 hours. If necessary, you can force the refresh of the license. For that purpose, go to the Portal, open the view "License Management" and click the refresh button on the top right corner of the page.

For **offline license**, the modified license file will be sent attached to the notification. This file has to be uploaded to the Portal, on the view "License Management" such that changes are applied.

Limitation: notifications are sent only if the license was activated and is not revoked.

### *Connectivity issues*

After three days without connectivity between the Portal and the Central License Manager, an email is sent. The notification is repeated every 7 days.

Limitation: only for commercial license.

## License alerts (Engine)

Besides the Central License Manager, the Engine sends some kinds of license notifications as well.

The recipient of the notifications is the administrator of the Engine

***Maximum number of licensed devices reached***

Triggered when the Engine reaches the maximum number of devices specified in the license and a new source appears in the network.

## Limit alerts

Limit alerts warn you about a possible loss of information related to a technical limitation of the Engine.

Too many processes started on [device IP address]/[device name]

Triggered when more than 10 000 processes have been started by a single user on a device within 15 minutes and the processes are running simultaneously. The Engine does not store information about any other process for that user beyond that limit.

Too many connections generated by executable [executable name] on [device IP address]/[device name]

Triggered when more than 10 000 connections are established by a process on a device within 15 minutes. No more concurrent connections are stored in the Engine for that process beyond this limit.

Engine is about to reach or has reached the limit for the maximum number of ...

Triggered when more than the 95% of the maximum number of objects of a particular type are already stored in the Engine. When the limit is reached, the Engine stores no more objects of the given type. The Engine generates this alert for the following types of objects:

- ◊ Binaries, the maximum allowed are 40 000 binaries.
- ◊ Domains, the maximum allowed are 250 000 web domains.

Engine has detected a large amount of the following objects, which might casuse performance issues ...

Triggered when the number of objects of a particular type reaches an amount that may degrade the performance of the Engine. The Engine generates this alert for the following type of objects:

- ◊ Destinations, when the Engine has recorded more than 50 000 destinations.

The limit alerts for binaries, domains and destinations include additional information on the Engine that generated the alert (Source), and the user concerned (User), which is *admin* for system alerts.

### Internal alerts

Internal alerts provide you with general information on the status of the Engine.

Server started
: Triggered when the Nexthink Engine reboots.

Unable to connect to Nexthink Application Library
: Occurs when the Engine cannot connect to the Application Library to get information on binaries and packages.

### Server Crash alert

The server crash alert is issued on the occurrence of an unrecoverable error in the Engine.

Server crash
: Triggered when the Nexthink Engine finds a minidump file in the database directory while rebooting, meaning that the Engine crashed previously.

Related tasks

- Receiving alerts

Related concepts

- Alert

## Audit trail

### Overview

To trace relevant activities in your Appliances (accesses, configuration modifications, starts, stops, etc), Nexthink components write to the audit log file:

```
/var/log/nexthink/audit.log
```

Find below the complete list of audit events. In the tables, the words displayed in *cursive* in the log messages are replaced by actual values by the log system. For example, the actual username of the account that performed a particular logged action will replace the word *account*.

## Appliance

See how to configure the system log for the Appliance to record the following events:

- Logon with the SSH Nexthink account
- Commands launched with super-user privileges

## Web Console

| Code | Description and format |
|---|---|
| 50000 | User logged in<br>`[Console\|Login\|50000\|`*`account`*`] Login successful` |
| 50001 | User login failed<br>`[Console\|Login\|50001\|`*`account`*`] Login failed` |
| 50002 | User logged out<br>`[Console\|Login\|50002\|`*`account`*`] User logout` |
| 51000 | Web Console password updated<br>`[Console\|Appliance\|51000\|`*`account`*`] Console password updated` |
| 51010 | Portal remote management account password updated<br>`[Console\|Appliance\|51010\|`*`account`*`] Remote password updated` |
| 51011 | Portal remote management account enabled<br>`[Console\|Appliance\|51011\|`*`account`*`] Remote access enabled` |
| 51012 | Portal remote management account disabled |

| | |
|---|---|
| | [Console\|Appliance\|51012\|*account*] Remote access disabled |
| 51020 | SSH Nexthink account password updated<br>[Console\|Appliance\|51020\|*account*] SSH Nexthink account password updated |
| 51021 | SSH Nexthink account enabled<br>[Console\|Appliance\|51021\|*account*] SSH Nexthink account enabled |
| 51022 | SSH Nexthink account disabled<br>[Console\|Appliance\|51022\|*account*] SSH Nexthink account disabled |
| 51100 | Appliance hostname updated<br>[Console\|Appliance\|51100\|*account*] Appliance hostname updated |
| 51101 | Appliance static route updated<br>[Console\|Appliance\|51101\|*account*] Appliance static route updated |
| 51102 | Appliance static route deleted<br>[Console\|Appliance\|51102\|*account*] Appliance static route deleted |
| 51103 | Appliance DNS server updated<br>[Console\|Appliance\|51103\|*account*] Appliance dns server updated |
| 51104 | Appliance default gateway updated<br>[Console\|Appliance\|51104\|*account*] Appliance default gateway updated |
| 51106 | Appliance NTP servers updated<br>[Console\|Appliance\|51106\|*account*] Appliance NTP servers updated |

| | |
|---|---|
| 51107 | Appliance NTP service enabled<br>`[Console\|Appliance\|51107\|`*`account`*`] Appliance NTP service enabled` |
| 51108 | Appliance NTP service disabled<br>`[Console\|Appliance\|51108\|`*`account`*`] Appliance NTP service disabled` |
| 51109 | Appliance network interface updated<br>`[Console\|Appliance\|51109\|`*`account`*`] Appliance network insterface updated` |
| 51111 | rsyslog service restarted<br>`[Console\|Appliance\|51111\|`*`account`*`] rsyslog service restarted` |
| 51112 | crond service restarted<br>`[Console\|Appliance\|51112\|`*`account`*`] crond service restarted` |
| 51603 | Automatic updates enabled / disabled<br>`[Console\|Appliance\|51603\|`*`account`*`] Automatic updates enabled`<br>`[Console\|Appliance\|51603\|`*`account`*`] Automatic updates disabled` |
| 51609 | Updates email recipient updated<br>`[Console\|Appliance\|51609\|`*`account`*`] Updates email recipient updated` |
| 51610 | Check for updates triggered<br>`[Console\|Appliance\|51610\|`*`account`*`] Check for updates triggered` |
| 51611 | Start updates triggered<br>`[Console\|Appliance\|51611\|`*`account`*`] Start updates triggered` |
| 51800 | Appliance reboot triggered |

| | |
|---|---|
| | [Console\|Appliance\|51800\|*account*] Appliance reboot triggered |
| 52000 | Portal parameters updated<br>[Console\|Portal\|52000\|*account*] Portal parameters updated |
| 52001 | Engine name updated<br>[Console\|Engine-01\|52001\|*account*] Engine name updated |
| 52007 | Maximum stored events updated<br>[Console\|Engine-01\|52007\|*account*] Maximum stored events updated |
| 52010 | Portal server address updated<br>[Console\|Engine-01\|52010\|*account*] Portal server address updated |
| 52010 | Portal admin account reset<br>[Console\|Portal\|52010\|*account*] Portal admin account reset |
| 52011 | Aggregation policy updated<br>[Console\|Engine-01\|52011\|*account*] Aggregation policy updated |
| 52012 | Domain compression updated<br>[Console\|Engine-01\|52012\|*account*] Domain compression updated |
| 52090 | Engine stopped<br>[Console\|Engine-01\|52090\|*account*] Engine stopped |
| 52091 | Engine started<br>[Console\|Engine-01\|52091\|*account*] Engine started |
| 52100 | Internal network removed |

| | |
|---|---|
| | [Console\|Engine-01\|52100\|*account*] Internal network removed |
| 52100 | Internal network added<br>[Console\|Engine-01\|52100\|*account*] Internal network added |
| 52105 | Engine internal domains configuration updated<br>[Console\|Engine-01\|52105\|*account*] Engine internal domains configuration updated |
| 52200 | Active directory added<br>[Console\|Engine-01\|52200\|*account*] Active directory added |
| 52201 | Active directory removed<br>[Console\|Engine-01\|52201\|*account*] Active directory removed |
| 52550 | Engine Mobile Bridge parameters updated<br>[Console\|Engine-01\|52550\|*account*] Engine Mobile Bridge parameters updated |
| 53090 | Portal stopped<br>[Console\|Portal\|53090\|*account*] Portal stopped |
| 53091 | Portal started<br>[Console\|Portal\|53091\|*account*] Portal started |
| 53092 | LLM started<br>[Console\|Portal\|53092\|*account*] LLM started |
| 53093 | LLM stopped<br>[Console\|Portal\|53093\|*account*] LLM stopped |
| 53094 | Nginx started<br>[Console\|Portal\|53094\|*account*] nginx started |
| 53095 | |

| | |
|---|---|
| | Nginx stopped<br>`[Console\|Portal\|53095\|`*`account`*`] nginx stopped` |

## Portal

| Code | Description |
|---|---|
| 20001 | Portal is starting<br>`[Portal\|SYSTEM\|20001\|*system] Portal is starting` |
| 20002 | Portal is up and running<br>`[Portal\|SYSTEM\|20002\|*system] Portal is up and running` |
| 20004 | Portal is stopped<br>`[Portal\|SYSTEM\|20004\|*system] Portal is stopped` |
| 20101 | User logged in<br>`[Portal\|LOGIN\|20101\|`*`account`*`] User `*`account`*` logged with session id `*`session id`* |
| 20102 | User logged out<br>`[Portal\|LOGIN\|20102\|`*`account`*`] User `*`account`*` logout for session id `*`session id`* |
| 20103 | User login failed<br>`[Portal\|LOGIN\|20103\|*system] User `*`account`*` failed login attempts – `*`reason`* |
| 20201 | User created<br>`[Portal\|USER\|20201\|`*`account`*`] User `*`created account`*` is created` |
| 20202 | User removed<br>`[Portal\|USER\|20202\|`*`account`*`] User `*`deleted account`*` is removed` |
| 20203 | User updated |

| | |
|---|---|
| | [Portal\|USER\|20203\|*account*] User *updated account* is created |
| 20204 | User profile updated<br>[Portal\|USER\|20204\|*account*] Updated profile of *n* users |
| 20205 | User domain ownership updated<br>[Portal\|USER\|20204\|*account*] Updated account ownership of *n* users |
| 20206 | Role added<br>[Portal\|USER\|20206\|*account*] Role *name* is added |
| 20207 | Role updated<br>[Portal\|USER\|20207\|*account*] Role *name* is updated |
| 20208 | Role removed<br>[Portal\|USER\|20208\|*account*] Role *name* is removed |
| 20209 | Profile added (with roles)<br>[Portal\|USER\|20209\|*account*] Added profile *name* roles: *roles names* |
| 20210 | Profile updated (with roles)<br>[Portal\|USER\|20210\|*account*] Updated profile *name* roles: *roles names* |
| 20211 | Profile removed<br>[Portal\|USER\|20211\|*account*] Removed profile *name* |
| 20501 | Hierarchy added<br>[Portal\|HIERARCHY\|20501\|*account*] Hierarchy *name* is added |
| 20502 | Hierarchy removed<br>[Portal\|HIERARCHY\|20502\|*account*] Hierarchy *name* is removed |

| | |
|---|---|
| 20503 | Hierarchy updated<br>[Portal\|HIERARCHY\|20503\|*account*] Hierarchy *name* is updated |
| 20504 | Definition of entities updated<br>[Portal\|HIERARCHY\|20504\|*account*] CSV of entities category is updated |
| 20701 | Engine added<br>[Portal\|ENGINE\|20701\|*account*] Engine *name* of IP *IP address or DNS name* Port *port number* is added |
| 20702 | Engine removed<br>[Portal\|ENGINE\|20702\|*account*] Engine *name* of IP *IP address or DNS name* Port *port number* is removed |
| 20703 | Engine connected<br>[Portal\|ENGINE\|20703\|*account*] Engine *name* of IP *IP address or DNS name* Port *port number* is connected |
| 20704 | Engine disconnected<br>[Portal\|ENGINE\|20704\|*account*] Engine *name* of IP *IP address or DNS name* Port *port number* is disconnected |
| 20801 | Finder user logged in<br>[Portal\|FINDER\|20801\|*account*] User *account* logged in (finder) |
| 20803 | Finder user login failed<br>[Portal\|FINDER\|20801\|*account*] User *account* login failed |
| 20804 | Library pack import request (only issued for big packs)<br>[Portal\|FINDER\|20804\|*account*] Finder import req uid=*pack uid* |
| 20901 | |

| | |
|---|---|
| | Remote action updated<br>[Portal\|CONTENTMANAGER\|20901\|*account*] Updated remote action in content manager, uid=*remote action uid*, name=*remote action name* |
| 20902 | Remote action created<br>[Portal\|CONTENTMANAGER\|20902\|*account*] Created remote action in content manager, uid=*remote action uid*, name=*remote action name* |
| 20903 | Remote action deleted<br>[Portal\|CONTENTMANAGER\|20902\|*account*] Deleted remote action in content manager, uid=*remote action uid* |
| 20911 | Metric updated<br>[Portal\|CONTENTMANAGER\|20911\|*account*] Updated metric in content manager, uid=*metric uid*, status=*enabled\|disabled* |
| 20912 | Metric created<br>[Portal\|CONTENTMANAGER\|20912\|*account*] Created metric in content manager, uid=*metric uid* |
| 20913 | Metric deleted<br>[Portal\|CONTENTMANAGER\|20913\|*account*] Deleted metric in content manager, uid=*metric uid* |
| 20921 | Service updated<br>[Portal\|CONTENTMANAGER\|20921\|*account*] Updated service in content manager, uid=*service uid*, status=*enabled\|disabled* |
| 20922 | Service created<br>[Portal\|CONTENTMANAGER\|20922\|*account*] Created service in content manager, uid=*service uid* |
| 20923 | Service deleted |

| | |
|---|---|
| | [Portal\|CONTENTMANAGER\|20923\|*account*] Deleted service in content manager, uid=*service uid* |
| 20931 | Campaign updated<br>[Portal\|CONTENTMANAGER\|20931\|*account*] Updated campaign in content manager, uid=*campaign uid*, name=*campaign name*, status=*draft\|published\|retired* |
| 20932 | Campaign created<br>[Portal\|CONTENTMANAGER\|20932\|*account*] Created campaign in content manager, uid=*campaign uid*, name=*campaign name* |
| 20933 | Campaign deleted<br>[Portal\|CONTENTMANAGER\|20933\|*account*] Deleted campaign in content manager, uid=*campaign uid* |
| 21001 | Manual execution of a remote action through the Finder<br>[Portal\|REMOTEACTION\|21001\|*account*] Finder request manual execution of remote action, uid=*remote action uid* on *n* devices with uids *devices uids* |
| 21002 | External execution of a remote action through the API<br>[Portal\|REMOTEACTION\|21002\|*account*] API request manual execution of remote action, uid=*remote action uid* on *n* devices with uids *devices uids* |
| 21101 | Metric compute triggered from the Finder<br>[Portal\|METRICS\|21101\|*account*] Compute metric from finder uid=*metric uid* |
| 21102 | Metric clear history triggered by query<br>[Portal\|METRICS\|21102\|*account*] Clear metric from query uid=*metric uid* |
| 21103 | Metric clear triggered from the Finder |

| | |
|---|---|
| | [Portal\|METRICS\|21103\|*account*] Clear metric from finder uid=*metric uid* |
| 21104 | Metric compute triggered by query<br>[Portal\|METRICS\|21104\|*account*] Compute metric from query uid=*metric uid* |
| 21201 | Module published<br>[Portal\|MODULES\|21201\|*account*] Published module uid=*module uid*, name=*module name*\|- |
| 21202 | Module deleted<br>[Portal\|MODULES\|21202\|*account*] Deleted module uid=*module uid* |
| 21203 | Module replaced<br>[Portal\|MODULES\|21203\|*account*] Replaced published module uid=*module uid*, replaced uid=*module uid* |
| 21501 | Dashboard deleted<br>[Portal\|DASHBOARDS\|21501\|*account*] Deleted dashboard, uid=*dashboard uid* |
| 21301 | Software metering metric updated<br>[Portal\|SOFTWARE_METERING_METRIC\|21301\|*account*] Updated software metering metric, uid=*metric uid* |
| 21302 | Software metering metric deleted<br>[Portal\|SOFTWARE_METERING_METRIC\|21302\|*account*] Deleted software metering metric, uid=*metric uid* |
| 21303 | Software metering metric enabled<br>[Portal\|SOFTWARE_METERING_METRIC\|21303\|*account*] Enabled software metering metric, uid=*metric uid* |
| 21304 | Software metering metric disabled<br>[Portal\|SOFTWARE_METERING_METRIC\|21304\|*account*] Disabled software metering metric, uid=*metric* |

| | |
|---|---|
| | *uid* |
| 21401 | Software metering module updated<br>[Portal\|SOFTWARE_METERING_MODULE\|21401\|*account*] Updated software metering module, uid=*module uid* |
| 21402 | Software metering module created<br>[Portal\|SOFTWARE_METERING_MODULE\|21402\|*account*] Created software metering module, uid=*module uid* |

## Engine

| Code | Description |
|---|---|
| 10001 | Engine is up and running<br>[Engine-01\|General\|10001\|nxengine] Engine is up and running |
| 10002 | Engine stopped with error<br>[Engine-01\|General\|10002\|nxengine] Engine abnormally stopped |
| 10003 | Engine stopped gracefully<br>[Engine-01\|MAIN\|10003\|nxengine] Engine gracefuly stopped |
| 10004 | Engine stopped forcefully<br>[Engine-01\|General\|10004\|nxengine] Engine stopped |
| 10005 | Database created<br>[Engine-01\|Database\|10005\|nxengine] Engine database creation:new database created |
| 10006 | Finder user logged in<br>[Engine-01\|Communication\|10006\|*account*] Finder user logged in:[*milliseconds*] |
| 10007 | Finder user logged out |

| | |
|---|---|
| | [Engine-01\|Communication\|10007\|*account*] Finder logged out |
| 10008 | Finder user login attempt<br>[Engine-01\|Communication\|10008\|*account*] Finder log-in attempt |
| 10009 | Finder account created<br>[Engine-01\|Database\|10009\|portal] Finder account creation:[*created account*] |
| 10010 | Finder account deleted<br>[Engine-01\|Database\|10010\|portal] Finder account destruction:[*deleted account*] |
| 10011 | Finder account updated<br>[Engine-01\|Database\|10011\|portal] Finder account update:[*updated account*] |
| 10012 | Finder account password changed<br>[Engine-01\|Database\|10012\|portal] Finder password change:[*changed account*] |
| 10017 | Global alert created<br>[Engine-01\|Database\|10017\|portal] Global alert creation:[*alert name*] |
| 10018 | Global alert deleted<br>[Engine-01\|Database\|10018\|portal] Global alert destruction:[*alert name*] |
| 10019 | Global alert updated<br>[Engine-01\|Database\|10019\|portal] Global alert update:[*alert name*] |
| 10026 | LDAP synchronization request<br>[Engine-01\|Communication\|10026\|*account*] LDAP synchronization |

| | |
|---|---|
| 10028 | Object manually tagged<br>[Engine-01\|DBMGR\|10028\|*account*] Manual tagging:[*object type*\|*object name*] |
| 10029 | Binary filtering rule (storage policy) updated<br>[Engine-01\|DBMGR\|10029\|*account*] Binary filtering rule update:[binary\|*executable name*] |
| 10030 | Executable filtering rule (storage policy) updated<br>[Engine-01\|DBMGR\|10030\|*account*] Application filtering rule update:[application\|*executable name*] |
| 10031 | Application filtering rule (storage policy) updated<br>[Engine-01\|DBMGR\|10031\|*account*] Product or source filtering rule update:[product\|*application name*] |
| 10032 | Device filtering rule (storage policy) updated<br>[Engine-01\|DBMGR\|10032\|*account*] Source filtering rule update:[source\|*device name*] |
| 10034 | Finder request execution<br>[Engine-01\|Communication\|10034\|*account*] Request execution:[*request type*\|*request details*] |
| 10035 | Alert execution<br>[Engine-01\|Alert\|10035\|*account*] Alert execution:[*alert name*\|*alert frequency*\|*number of impacted objects*\|*selector*] |
| 10038 | License updated<br>[Engine-01\|License\|10038\|nxengine] License updated: *D* licensed sources, *S* licensed servers, *M* licensed mobile devices with *enabled* features |
| 10039 | NXQL request execution<br>[Engine-01\|WebAPI\|10039\|*account*] NXQL V2 execution:[*duration ms*\|*wait ms*\|*computation ms*\| |

| | | |
|---|---|---|
| | | *dump ms\|NXQL query*] |

The start and stop commands for the Engine that are executed from the CLI are logged in journalctl. Use the following command to retrieve them:

```
sudo journalctl –u nxengine@*.service | grep systemd
```

Related tasks

- Configuring the system log

# GDPR script

## Deprecated

The functions of the GDPR script are now available from the Web Console.

Please use the graphical interface provided by the Web Console instead of running the GDPR script from the command line.

# Appliance Hardening

## Overview

Starting from 6.17, Nexthink Appliances are better protected against unauthorized accesses and malicious attacks by default. To comply with eventual security audits, the measures described in the Security Hardening Guide are now automatically applied to every fresh installation of the Appliance. By their very nature, two of the steps in the Security Hardening Guide cannot be automated and still demand manual intervention:

- Changing the default passwords.
- Replacing the default certificates.

The Appliance requires additional communication ports to be open depending on the Nexthink server component (if any) that is is installed along with the system packages. The automatic hardening procedure opens the ports needed by the Portal, the Engine or both when they are installed on top of the Nexthink Appliance.

In the last sections of this article, learn how to open additional ports in the Appliance that you may need for your specific setup and how to enforce security hardening in existing Appliances. Because the hardening procedure is only automatic for fresh installations of the Appliance, you may find these sections useful if you are upgrading your Nexthink Appliances to V6.17 or higher.

## Hardening measures

### *ISO hardening*

The following measures are applied to every new installation of the Appliance:

- Disable ICMP redirection (kernel parameter).
- Enable the strongest SSH and TLS ciphers only.
- Disable HTTP communication and allow secure HTTP (HTTPS) only.
- By default, the only open ports for listening are TCP 99 and TCP 22.
- Umask configuration is less permissive (umask is set to 0027 for all users).
- Ensure that the partitions mountpoints cannot be mis-used.

### *Portal*

In addition to TCP ports 99 and 22, the following ports are open by default when installing the Portal on the Appliance:

- TCP 443 and 80.

After federation, these additional communication channels with the Engine are open as well, but they are only accessible to the host names or IPs of the federated Engines:

- TCP 7000, 7001, 7002 and 7003.

Therefore, federation is mandatory in hardened Appliances to enable the real-time communication between the Portal and the Engines. Because of this same reason, it is not possible to work in compatibility mode.

### *Engine*

In addition to TCP ports 99 and 22, the following ports are open by default when installing the Engine on the Appliance:

- TCP 99, 22, 999, 8443 and 1671.

- UDP 999.

## Enabling additional ports

The automatic hardening only enables the default ports or, for those Engine ports that are configurable, it enables the ports for which you have changed the default number.

Third-party applications other than Nexthink that you install in the Appliance may require additional communication ports. To enable additional ports in the Engine or the Portal Appliances, even when hardening is turned on:

1. Log in to the Web Console of either the Portal or the Engine Appliance.
2. Select the **APPLIANCE** tab at the top of the Web Console.
3. Click **Security** on the left-hand side menu.
4. Under **Custom ports**:
    - Type in the additional UDP ports required inside the **UDP ports** box. Separate each port number by a new line.
    - Type in the additional TCP ports required inside the **TCP ports** box. Separate each port number by a new line.
5. Click **SAVE**.

## Enforce hardening from the Web Console

Only fresh installations of a V6.17 or higher Appliance are hardened. Starting from V6.18, you can protect upgraded Appliances with the same security settings of a fresh V6.17 or higher Appliance from the Web Console. Keep in mind that the Appliances must be federated before enforcing their hardening.

To harden your upgraded Appliances from the Web Console:

1. Log in to the Web Console of either the Portal or the Engine Appliance.
2. Select the **APPLIANCE** tab at the top of the Web Console.
3. Click **Security** on the left-hand side menu.
4. Under **Security hardening**, tick the option **Keep appliance secure**.

5. Click **SAVE**.

Related tasks

- Federating your Appliances
- Managing Appliance accounts
- Importing and replacing Certificates
- Changing the default ports in the Engine
- Installing third-party software in the Appliance

Related references

- Connectivity requirements
- Compatibility mode
- Security Hardening Guide (Community)

# References

## Components of the Collector

### Overview

The Collector is composed of a set of services and libraries that gather information about the devices in your corporate network and their activity. The Collector periodically sends all the gathered information to an Engine, where it is processed and stored. Additional components of the Collector deal with the features provided by optional Nexthink modules. Finally, other components help you with the installation and configuration of the Collector.

Find in this document the description of all the different components of the Collector and the filesystem paths where to find them in the devices of the end users after installation. This article details as well the registry keys and the additional files created or modified during the installation of the Collector.

Applies to platforms:

### Windows Collector

The Windows version of the Collector includes the following set of components.

Applies to platforms:

***Windows Collector binaries***

For all versions of Windows, the following components are installed:

**Main driver**
: A kernel mode driver that gathers valuable information from the device of the end-user.

**Network specific driver**
: A kernel mode driver that detects network connections.

**Helper service**
: A Windows service that complements the main driver by collecting additional information.

**Printing info library**
: A dynamic link library that is responsible for detecting printing activity.

Optional **Command line configuration tool**
: A tool to configure the Collector from the command line.

Optional **Control Panel extension**
: A tool to control the behaviour of the Collector that is added to the Control Panel of Windows.

**Automatic updates**
: A component of the Collector that is responsible for downloading new versions and updating the installed components.

**Coordinator**
: Coordination of the Collector with the Appliance to detect new updates, engage with end-users, and execute remote actions.

**Nexthink Engage**
: Components for presenting the questions of campaigns and getting answers from the end-users.

**Nexthink Act**
: Components that manage the execution of remote actions.

**Nexthink Reporter**
: A troubleshooting tool that creates debug reports for specific support cases.

**Nexthink Event Log Provider**
: A component for logging events in the Windows Event Log.

| Component | File | Path |
|---|---|---|
| Main driver | nxtrdrv.sys | %Windows%\System32\drivers |
| Network specific driver | nxtrdrv5.sys | |
| Command line configuration tool | nxtcfg.exe | %Windows%\System32 |
| Control Panel extension | nxtpanel.cpl | |
| Helper service | nxtsvc.exe | %ProgramFiles%\Nexthink\Collector\Collector |
| Printing info helper library | nxtdll.dll | |
| Nexthink Event Log Provider | nxteventprovider.dll | |
| Immersive apps | nxtwrt.dll | |
| Application start time | nxtwpm.dll | |
| Application start time (32 bit) | • nxtwpm32.dll<br>• nxtusm.exe | %Windows%\SysWOW64 |

| | | |
|---|---|---|
| Application start time | nxtwpm.dll | |
| Coordinator service | nxtcoordinator.exe | %ProgramFiles%\Nexthink\Collector\Coordinator |
| Engage coordinator | nxteufb.exe | |
| Act coordinator | nxtcod.exe | |
| Updates coordinator | nxtupdater.exe | |
| OpenSSL (64 bit) | • libcrypto-1_1-x64.dll<br>• libssl-1_1-x64.dll | |
| OpenSSL (32 bit) | • libcrypto-1_1.dll<br>• libssl-1_1.dll | |
| Nexthink Engage | • nxtray.exe<br>• nxtray.exe.config | %ProgramFiles%\Nexthink\Collector\Engage |
| Nexthink Act | • Google.Protobuf.dll<br>• nxtcampaignaction.dll<br>• nxtremoteactions.dll | %ProgramFiles%\Nexthink\Collector\RemoteActions |
| Nexthink Reporter | nxtreporter.exe | %ProgramFiles%\Nexthink\Collector\Reporter |

***Registry keys of the Windows Collector***

On installation, the Collector creates the following keys in the Registry of Windows:

```
HKEY_CLASSES_ROOT\nxtrayproto
HKEY_LOCAL_MACHINE\SOFTWARE\Nexthink\Collector
HKEY_LOCAL_MACHINE\SOFTWARE\Nexthink\Collector\AppStartTime
HKEY_LOCAL_MACHINE\SOFTWARE\Nexthink\DN
HKEY_LOCAL_MACHINE\SOFTWARE\Nexthink\RebootMarker
HKEY_LOCAL_MACHINE\SOFTWARE\Nexthink\RemoteActions
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\EventLog\Application\Nexthink
Collector
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Nexthink
Coordinator
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Nexthink
Coordinator\params
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Nexthink
Coordinator\Modules\COD
```

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Nexthink
Coordinator\Modules\EndUserFeedback
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Nexthink
Coordinator\Modules\Updater
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\Nexthink Service
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\Nexthink
Service\runtime_stats
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\nxtrdrv
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\nxtrdrv\params
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\nxtrdrv5
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\nxtrdrv5\Parameters\Wdf
HKEY_LOCAL_MACHINE\SYSTEM\Nexthink\Updater
HKEY_USERS\S-1-5-21-[X-X-X-X]\SOFTWARE\NEXThink\NxTray
```

### *Additional files of the Windows Collector*

Find the log files of the Collector here:

- %windir%\nxtsvc.log
- %windir%\nxtsvc.1.log
- %windir%\nxtsvc.2.log
- %windir%\nxtupdater.log
- %windir%\nxtupdater.1.log
- %windir%\nxtupdater.2.log
- %windir%\nxtcoordinator.log
- %windir%\nxtcoordinator.1.log
- %windir%\nxtcoordinator.2.log
- %windir%\nxteufb.log
- %windir%\nxteufb.1.log
- %windir%\nxteufb.2.log
- %windir%\nxtcod.log
- %windir%\nxtcod.1.log
- %windir%\nxtcod.2.log
- %temp%\nxtray.log
- %temp%\nxtray.log.<timestamp>

Finally, Windows creates a cached copy of the kernel drivers in two folders whose names start with the name of the drivers (**nxtrdrv** and **nxtrdrv5**, respectively) followed by an unique identifier that depends on the version of the driver itself. Find the folders here:

- %windir%\System32\DRVSTORE

The Nexthink Reporter tool creates its logs and reports here:

- %temp%\nxtreporter[reportID].log
- %temp%\nxtreport-[hostname]-[reportID].zip

## Mac Collector

The Mac version of the Collector includes the following set of components.

Applies to platforms:
***Mac Collector binaries***

- **Main service**: A Mac daemon that gathers valuable information from the device of the end-user.
- **Coordination service**: A Mac daemon that synchronizes with the appliances to provide services such as automatic updates, end-user engagement and execution of remote actions in the near future.

| Component | File | Path |
|---|---|---|
| Main service | nxtsvc | /Library/Application Support/Nexthink |
| Coordination service | nxtcoordinator | |

***Configuration files of the Mac Collector***

Starting from V6.21, there is only one configuration file for the Mac Collector:

| Component | File | Path |
|---|---|---|
| Configuration file | config.json | /Library/Application Support/Nexthink |

At the end of the file `config.json`, find the exact version of the installed Collector and the status of the TCP connection.

***Additional files of the Mac Collector***

Find the log files of the Mac Collector here:

- /Library/Logs/nxtsvcgen.log
- /Library/Logs/nxtsvcgen.*n*.log (*n* positive, when previous log is rotated)
- /Library/Logs/nxtcoordinator.log

Related tasks

- Installing the Collector on Windows
- Installing the Collector on macOS
- Updating the Collector

Related references

- Collector MSI parameters reference table
- Nxtcfg - Collector configuration tool
- Collector (Product Overview)

# Operating systems supported by the Collector

## Windows

Nexthink supports the installation of the Collector on those versions of the Windows operating system that are currently supported by Microsoft.

| Windows OS |
| --- |
| Windows 7 (SP1) |
| Windows 8.1 |
| Windows 10 |

The Collector supports both the Long-Term Servicing Channel and Semi-Annual Channel releases of Windows 10, in accordance with Microsoft Modern Lifecycle Policy.

The latest version of Windows 10 on which the Collector has been successfully tested is

- **Windows 10, version 1903**

## Windows Server

The Collector supports the installation on Windows Server for those versions currently supported by Microsoft, except for the Nano Server edition.

| Windows Server OS |
| --- |
| Windows Server 2008 R2 |
| Windows Server 2012 |
| Windows Server 2012 R2 |
| Windows Server 2016 (except Nano Server) |
| Windows Server 2019 (except Nano Server) |

The Collector supports both the Long-Term Servicing Channel and Semi-Annual

Channel releases of Windows Server, in accordance with Microsoft Modern Lifecycle Policy.

The latest version of Windows Server on which the Collector has been successfully tested is

- **Windows Server, version 1809**

### Mac

Nexthink supports the installation of the Collector on those versions of the macOS operating systems that are currently supported by Apple.

| macOS |
| --- |
| macOS 10.12 Sierra |
| macOS 10.13 High Sierra |
| macOS 10.14 Mojave |
| macOS 10.15 Catalina |

The latest version of macOS on which the Collector has been successfully tested is

- **macOS 10.15 Catalina Beta 5**

Related references

- Server support

## Server support

### Overview

Although Nexthink is a solution designed for monitoring the devices of the end-users, the same monitoring techniques may be applied to servers to some extent. In Nexthink V5, it is possible to install the Collector in Citrix / RDS servers, where each server is roughly equivalent to having a group of end-user devices. Starting from V6, Nexthink also supports the installation of the Collector in other types of Windows servers (note that V5.3 offers server support for specific Windows Server versions as well). Therefore, a new type of device is available in Nexthink: the **server**.

The Collector reports basically the same analytics from a server as from the device of an end-user, except for some security-related information. Indeed, the technology used by the Collector to retrieve security information is not available on servers, so data on the following areas is missing:

- Antivirus
- Antispyware
- Firewall

Also keep in mind that, as for normal devices, the Collector does not report incoming connections for servers. Only outgoing connections are recorded.

## Sizing your installation

Servers usually generate much more activity than end-user devices. As a rule of thumb for sizing your installation, consider that a server is equivalent to 20 end-user devices. Calculate the number of Engines that you need according to the following formula, where **S** is the number of servers and **D** is the number of end-user devices in your setup:

- Number of Engines = (20 * S + D) / 6000

The hardware requirements for the Engines apply.

## Traffic reduction

The typically higher network activity of servers with respect to end-user devices often generates lots of connections and events that might saturate the Engine. Therefore, the Engine has set in place a strategy to reduce the traffic of servers, although it applies to the traffic of all types of devices. When a device connects to too many destinations or opens too many ports in a burst, the Engine can automatically decide to aggregate these connections into a single connection, setting its port or its destination value to **multiple**. In the case of a device launching a burst of web connections to many domains, the Engine aggregates the connections as one connection where the value of the domain is **multiple**.

In this manner, individual information about the connections is lost, but the amount of traffic information stored in the Engine is kept to a reasonable level. Otherwise, the explosion of connections could drastically reduce the history available in the Engine. Even with the traffic reduction policy in place, you should expect a slight reduction on the history available in the Engine if you install the Collector on servers.

The strategy for reducing traffic is configurable (see below). Choose between **normal** and **aggressive**, depending on whether you prefer to aggregate connections gently or almost right away. An aggressive policy lets you keep a longer history in the Engine at the expense of losing the individual information of more connections.

## Taxonomy of servers

Find below a classification of servers according to their function. Depending on the function of the server, you should be aware of the chances that the Engine reduces server traffic.

Client-like (Citrix, RDS)
: Supported from V5. Traffic reduction rare.

Application (Mail, SQL Database)
: Traffic reduction depending on load.

Agent manager (SCCM)
: Traffic reduction probable.

UDP server (DNS)
: Traffic reduction certain.

Proxy (Web proxy)
: Expect to have web traffic reduction and bigger Collector usage of your network. Installation on a web proxy is therefore not recommended.

Bot (Scanner, Stress test)
: Not supported, since just one server may behave as thousands of end-user devices. Do not install the Collector on servers of the bot class.

## Engine configuration

Set the traffic reduction policy by editing the configuration file of the Engine. In addition, configure whether you want the outgoing traffic of servers to be used in the computation of services. Usually, this only makes sense for client-like (Citrix / RDS) servers.

1. Log in to the CLI of the Engine.
2. Open the configuration file for editing:
   ```
   sudo vi /var/nexthink/engine/01/etc/nxengine.xml
   ```
3. Write the following settings in the correspoding section of the configuration file:
   - In the **aggregation** section, set **destination_reduction_policy** to **normal** or **aggressive**.
   - In the **service** section, set **compute_service_on_server** to **true** or **false**.

4. Save your changes and exit:

```
:wq
```

Alternatively, use the nxinfo command to change the configuration settings of the Engine. For instance, to set the destination reduction policy to normal and turn on the computation of services on servers, type in:

- `sudo nxinfo config -s aggregation.destination_reduction_policy=normal`
- `sudo nxinfo config -s service.compute_service_on_server=true`

After modifying the settings, find the following lines with the provided values in the configuration file:

```
<aggregation>
  <destination_reduction_policy>normal<destination_reduction_policy>
</aggregation>
<service>
  <compute_service_on_server>true<compute_service_on_server>
</service>
```

Depending on the types of servers that you have, use the settings described below.

***Only client-like (Citrix / RDS) servers***

- Destination reduction policy: **normal**.
- Compute service on servers: **true**.

Whenever possible, assign the Citrix / RDS servers to the same Engines of the end-user devices that they serve.

***Only non-client servers***

- Destination reduction policy: **aggressive**.
- Compute service on servers: **false**.

If possible, assign non-client servers to an Engine separate from those used by end-user devices.

***Mixed setup***

- Destination reduction policy: **aggressive**.
- Compute service on servers: **true**.

In the case of a mixed setup with both client-like and non-client servers, you may want to compute a service for client-like servers and actual clients (end-user devices) only.

To compute services selectively, manually tag those devices that you want to include in the computation and set a condition on the services for taking into account only those tagged devices. For instance:

1. Create a category called, for example, **Compute services**.
2. Add two keywords to the categores without auto-tagging rules: **yes** and **no**.
3. Manually tag all your end-user devices and client-like servers with the keyword **yes** and the non-client servers with the keyword **no**.
4. Add a condition on devices in the definition of each service to include in the service only those devices tagged with the **yes** keyword:

Whenever possible, assign Citrix / RDS servers to the same Engines of the end-user devices that they serve and group non-client servers in separate Engines.

## Incompatibility of Collector with Receive Segment Coalescing (RSC)

Currently, the Collector is incompatible with Receive Segment Coalescing, a technology present in Windows Server 2012 and later that improves the reception of network traffic by offloading network processing from the CPU to the network interface. Windows desktop operating systems also support RSC since Windows 8, but the use of RSC is usually limited to Windows servers with high input traffic.

The driver of a network interface that is compatible with RSC can coalesce multiple TCP segments received and present them as a single larger segment to the networking layer of the operating system.

The monitoring capabilities of the Windows Collector at the kernel level effectively disable RSC on supported network interfaces.

Related references

- Hardware requirements
- Operating systems supported by the Collector
- Receive Segment Coalescing (external)

# Compatibility mode

*Compatibility mode* is a temporary state of Appliances V6.6 (or higher) that is previous to federation. In compatibility mode, Appliances work in much the same way as in V6.5 (or previous), hence its name. Starting from V6.17, federation of the Appliances is mandatory for proper communication between Portal and Engines.

Appliances enter compatibility mode right after their installation or after being updated from V6.5 (or previous). In this mode, there are no centralized configuration settings or coordinated updates. Unless you have a particular problem when federating your Appliances, it is recommended to switch from compatibility mode to federated mode as soon as possible.

An Appliance in master / slave configuration, that is, an Appliance with both the Portal and the Engine installed, is automatically federated. Therefore, it can never enter compatibility mode.

On the other hand, a slave Appliance which runs in compatibility mode shows two configuration settings in the Web Console that disappear when the Appliance is federated:

- The **Update** section
- The **Portal address** field

The **Update** section is found in the left-hand side menu of the Web Console, when the **Appliance** tab is selected. It allows for slave Appliances to be updated individually. After federation, the **Update** section is only available in the Web Console of the master Appliance.

## Real-time information

When working in compatibility mode, the Engine is able to send real-time information to the Portal only if you specify the address of the Portal in the Web Console of the slave Appliance. The **Portal address** setting is found by selecting the **Engine** tab, the **General** section of the left-hand side menu, and looking under **Parameters**. After federation, the address of the Portal is known to the Engine and this parameter is no longer needed; thus, it is removed from the Web Console.

Since V6.17, the automatic hardening of new installations of the Appliance prevents the real-time communication of data between the Portal and the Engines if they are not federated. Therefore, federation becomes mandatory from V6.17 on.

Related tasks

- Federating your Appliances