



# **Administrator Handbook**

**NOCTUA® Version**

**Georg Bahlon  
Natalia Hoesl  
DI Dr. Andreas Lang-Nevyjel  
Bernhard Mallinger**

## Administrator Handbook: NOCTUA® Version

by Georg Bahlon, Natalia Hoesl, DI Dr. Andreas Lang-Nevyjel, and Bernhard Mallinger

Release: 1.5 -This Document was generated 2015-05-27

Copyright © 2015 init.at informationstechnologie GmbH

### Abstract

This document is the official documentation for **NOCTUA®**. It explains general concepts of the software, gives an overview of it's components and walks you through various installation and administration tasks, the appendix documents and explains the stable part of the API.

**NOCTUA®** is a registered trademark of **init.at informationstechnologie GmbH**.

Linux® is a registered trademark of Linus Torvalds in the United States and other countries.

MySQL® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Red Hat®, Red Hat Enterprise Linux®, Fedora® and RHCE® are trademarks of Red Hat, Inc., registered in the United States and other countries.

Ubuntu® and Canonical® are registered trademarks of Canonical Ltd.

Debian® is a registered trademark of Software in the Public Interest, Inc.

SUSE® is a registered trademark of Novell, Inc.

openSUSE® is a registered trademark of the openSUSE community.

All trademarks are the property of their respective owners.

Please write to **<noctua@init.at>** to contact our developer and support team.

---

---

1. Introduction .....	1
1.1. Target audience .....	1
1.2. Prerequisite .....	1
1.3. Symbol description .....	1
2. About .....	3
2.1. What is <b>NOCTUA®</b> and what is it for? .....	3
2.2. Much more than just a webfrontend! .....	3
2.3. System Requirements .....	6
2.3.1. Common requirements .....	6
2.3.2. Database requirements .....	7
3. Installation .....	9
3.1. Supported operating systems .....	9
3.1.1. Repository access .....	9
3.1.2. Two different ways to install the software .....	9
3.1.3. Automatically installation with <i>install_icsw.py</i> .....	9
3.1.4. Manually installation .....	10
3.1.5. Repository .....	10
3.1.6. Repository directories .....	10
3.1.7. Debian repositories .....	11
3.1.8. Ubuntu repositories .....	11
3.1.9. OpenSUSE and SLES repositories .....	11
3.1.10. CentOS and RHELrepositories .....	12
3.1.11. Database setup .....	13
3.1.12. Portnumber for accessing the webfrontend .....	15
3.2. Installation .....	16
3.2.1. Install icsw packages .....	16
3.3. Installation on virtual machine .....	16
3.3.1. KVM libvirt/qemu .....	17
3.3.2. VMware .....	17
3.4. Upgrade software packages .....	17
3.5. Server control .....	18
3.5.1. Show server status .....	18
3.5.2. icsw commands .....	18
3.6. License administration .....	20
3.6.1. Login without licenses .....	20
3.6.2. License package structure .....	21
3.6.3. Upload license keyfile .....	21
3.6.4. Show license status .....	22
3.6.5. License time periods .....	23
3.6.6. License parameter limitations .....	23
3.6.7. License violation .....	23
4. Core concepts .....	25
4.1. Services and components .....	25
4.1.1. We do not need to reinvent the wheel .....	25
4.1.2. Common components .....	25
4.1.3. Monitoring components .....	25
4.1.4. HPC components .....	26
4.1.5. Clusterdatabase .....	26
4.1.6. Dataflow between webfrontend and database .....	27
4.2. Run important services .....	27
5. Webfrontend .....	29
5.1. First connection .....	29
5.2. Areas .....	29
5.2.1. Menu area (1) .....	30
5.2.2. Sidebar device tree (2) .....	31
5.2.3. Main area (3) .....	33
5.2.4. Cluster server information .....	34
6. Concept of operation .....	37

---

---

6.1. Working with the web front-end .....	37
6.1.1. Preselection and submenu .....	37
6.1.2. Preselection and homebutton .....	38
6.1.3. Two-stage selection .....	39
6.2. Input field filter/regex, hide/show column, pagination and sorting .....	39
6.2.1. Input field filter/regex .....	39
6.2.2. Hide/show columns .....	40
6.2.3. pagination .....	41
6.2.4. Sort column .....	41
6.3. Reversion .....	42
6.3.1. History overview .....	42
6.3.2. Revert to former version .....	43
7. User and group management .....	45
7.1. Create user or group .....	45
7.2. Create group form .....	45
7.3. Create user form .....	46
7.4. Permission System .....	47
7.4.1. Permission .....	47
7.4.2. Permission level .....	49
8. Device Management .....	51
8.1. Device tree .....	51
8.1.1. Filter function .....	51
8.1.2. Device tree overview .....	52
8.1.3. Create device, create devicegroup, modify selected and delete selected ....	53
8.1.4. Fast device creation .....	55
9. Network Management .....	57
9.1. Networks .....	57
9.1.1. Network device types .....	57
9.1.2. Network types .....	57
9.1.3. Networks .....	57
9.2. Device Network .....	57
9.2.1. Settings .....	57
9.2.2. Network topology .....	62
9.2.3. clusters .....	64
10. Monitoring .....	65
10.1. Three different ways of monitoring .....	65
10.1.1. Basic Monitoring .....	65
10.1.2. Advanced Monitoring SNMP/IPMI .....	65
10.1.3. Advanced Monitoring with host-monitoring .....	66
10.1.4. Setup host-monitoring and SNMP daemon .....	66
10.2. Configuration hierarchy .....	67
10.2.1. Catalog .....	68
10.2.2. Configurations .....	69
10.2.3. Configurations .....	70
10.2.4. Server configurations .....	71
10.3. Basic monitoring for devices .....	72
10.4. Basic monitoring for devicegroups .....	72
10.5. Livestatus .....	73
10.5.1. Livestatus filter options .....	73
10.5.2. Livestatus burst .....	73
10.5.3. Livestatus tables .....	76
10.5.4. Livestatus monitoring categories .....	76
10.6. Monitoring overview .....	76
10.7. Status History .....	78
10.8. Notifications .....	80
10.8.1. Periods .....	80
10.8.2. Notification templates .....	80
10.8.3. Contacts .....	82

---



10.8.4. Service templates .....	83
10.8.5. Device templates .....	85
10.8.6. Host check commands .....	86
10.8.7. Contact groups .....	86
10.8.8. Flapping .....	86
10.9. Parameterizing checks .....	88
10.9.1. Types of checks .....	89
10.9.2. Examples .....	89
10.9.3. Advantages of parameterizing .....	89
10.10. Distributed Monitoring .....	90
10.11. Evaluation of monitored data .....	90
10.12. Overview Configuring Nodes .....	90
10.13. Discovery server .....	90
10.14. Icinga .....	91
11. Graphing .....	93
11.1. Introduction to Graphing .....	93
11.1.1. Principles of RRD .....	94
11.1.2. Data collection and graphing .....	94
11.1.3. How to display RRD graphs? .....	95
11.1.4. RRD frontend .....	95
11.1.5. RRD tree components .....	100
11.1.6. Summarized graphs .....	101
11.1.7. Compound graphs .....	102
12. Device localisations .....	103
12.1. Setup localisation .....	103
12.1.1. Upload and edit user images .....	104
12.1.2. Edit uploaded photos .....	106
12.2. Livestatus integration in maps .....	107
12.2.1. Binding livestatus burst to maps .....	107
12.2.2. Display livestatus burst .....	107
13. SNMP discovery .....	109
13.1. Automatically network discovering with SNMP .....	109
13.1.1. Setup of SNMP discovery .....	109
13.1.2. Auto discover with SNMP .....	110
13.1.3. Auto discover with host-monitoring .....	111
14. Debugging and Error hunting .....	113
14.1. General information .....	113
14.2. Show errors .....	113
14.3. Node information .....	114
14.4. Logging .....	114
14.4.1. Directories for log files .....	114
14.4.2. Automatic log mail delivery system .....	114
14.4.3. icsw logwatch .....	115
14.4.4. Service communication ports .....	115
15. Extensions .....	117
15.1. Shared script directories .....	117
16. Frequently Asked Questions .....	119
16.1. FAQ .....	119
Glossary .....	129



---

# Chapter 1. Introduction

## 1.1. Target audience

Before you start to dive deep into the documentation, we think it is fair to let you know if you get the right information out of the document or not. This documentation is intended for system administrators who want to get into monitoring and cluster management. It is also intended for user who only have to operate with the software but don't do any configurations.

The handbook provides also some background information about **LINUX®** commands in general, i.e. in context with package installation.

It does not deal with general monitoring themes or basic principles of monitoring or cluster management. If you wish to learn something about that, please save your time and look for a more suitable document in the world wide web or in your local specialised bookstore.

## 1.2. Prerequisite











Below list shows you which prerequisite of users and administrators should be fulfilled to get into monitoring or cluster management with software by **init.at ltd.**

- Experience with **LINUX®** in general
- Experience with the **LINUX®** command line, e.g **bash**, **zsh** or other
- Experience with standard html browser
- Experience with network settings

## 1.3. Symbol description

For easier document handling on some place you will notice small icons with the following meanings:

**Table 1.1. Symboltable**



	<b>Link inside the documentation</b>
	<b>Mailto Link</b>
	<b>Internet http link</b>
	<b>Link to the glossary</b>
	<b>Mark some very important statement</b>
	<b>Moving mouse to menu with given name</b>
	<b>Work on this content is in progress</b>
	<b>Left mouse click</b>
	<b>Right mouse click</b>
	<b>Doublemouse click</b>



# Chapter 2. About

## 2.1. What is NOCTUA® and what is it for?

**NOCTUA®** is an extensive Software Package for monitoring devices. Monitoring means to observe, record, collect and display different aspects of hardware and software. This monitored aspects could be close to hardware like CPU Temperature, CPU Voltage, FAN Spin or existing or not existing NET devices but also close to services running on monitored machines like SSH daemons, POSTFIX daemons, HTTP services or simply the availability of devices via ping.


Not only monitoring of devices is possible but also different methods of reaction due to reached predefined limits. And best of all, **NOCTUA®** is  *open source software*, licensed under the  *GNU GPL 2.0 License*.

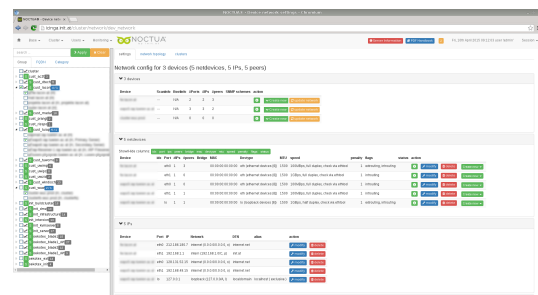
You can find more Information about open source software and it's benefits at <http://www.fsf.org>


## 2.2. Much more than just a webfrontend!

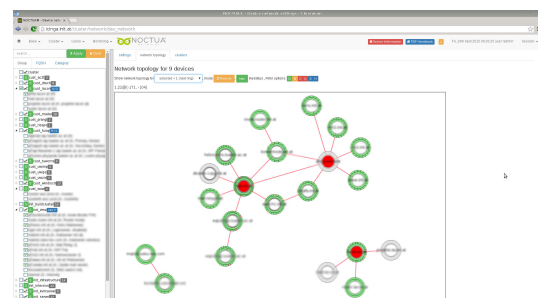
Even if the main task of **Monitoring** is to ease configuration and administration of icinga, there are some special and unique features icinga and other software solutions do not provide.

Following list contains the exclusive components that makes our software unique and unreachable on software market:

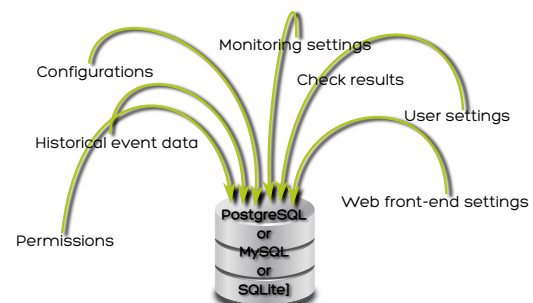
 **Web front-end** - Userfriendly and easy to use web front-end to access to all data, configurations and settings.



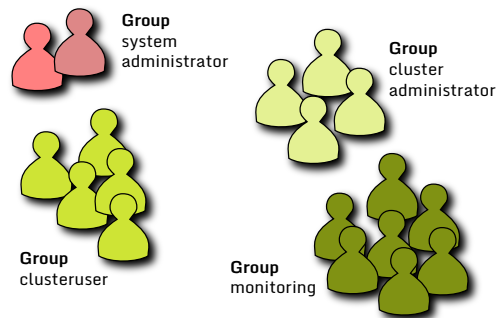
 **Peering** - Possibility to connect monitored devices to peers. Displays whole network topology of connected devices.



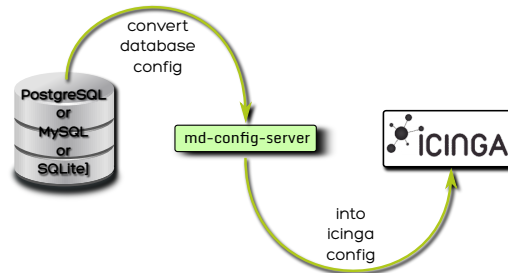
**Central database** - One central storage for all data like configs, settings, logging data user data and much more.



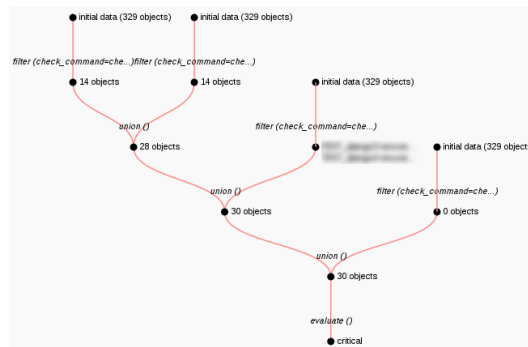
**User Management** - Administrator tool to manage groups, users and corresponding permissions.



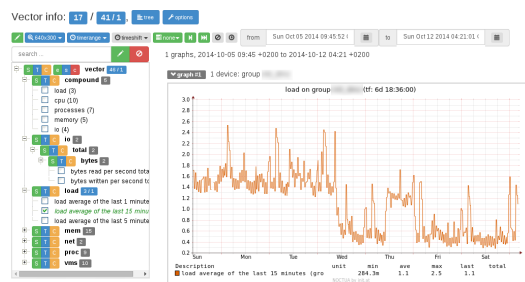
**MD-Config-Server** - md-config-server is responsible for proper configuration conversion into icinga config format



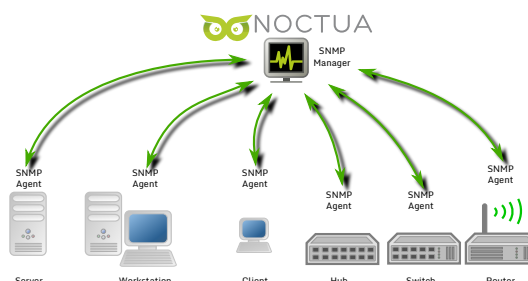
**KPI [coming soon]** - Key Performance Indicator, integration of user defined KPIs. Flexible preselection by device and monitoring categories.




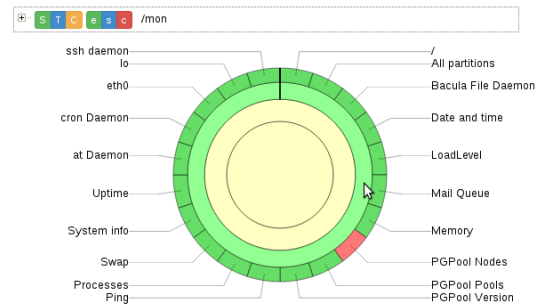
**Graphing** - Beautiful graphing of collected data, compound graphs for devicegroups or whole cluster, aggregation of graphs and many options to control output.




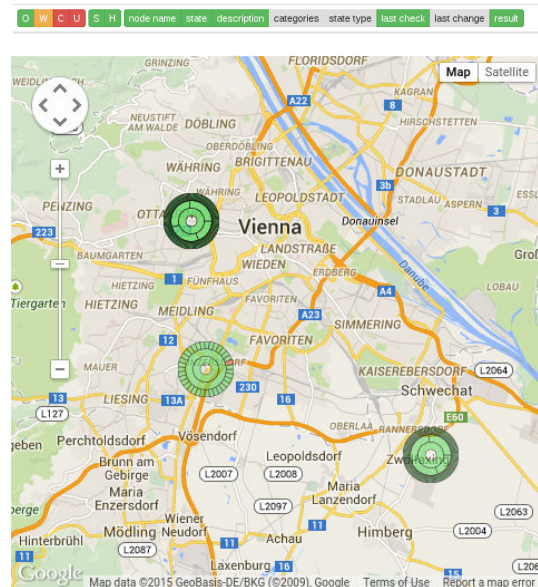
**Discovery Server** - Autodiscovery of network devices. Collect wide range information of devices with help of SNMP




 **Monitoring Dashboard - Livestatus** - Graphical realtime indication tool to display monitored data of devices or cluster. No more need to interpret date.




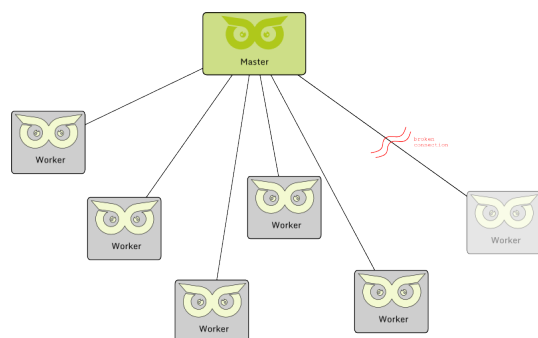
 **Monitoring Dashboard - Geolocation** - Integration of livestatus into google Maps™. Facility to see cluster/device status at a glance.



 **Monitoring Dashboard - Maplocation** - Option to upload user images or photos of e.g. cluster infrastructure, plant layout or server racks. Livestatus of cluster and devices can be displayed as overlay on image map.




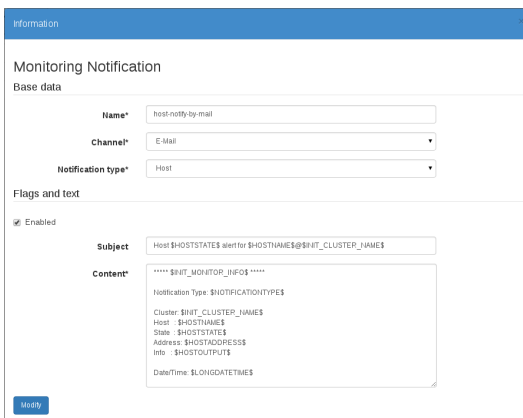
 **Distributed monitoring** - Run distributed instances of NOCTUA® as master/worker instances to increase performance and availability in complex networks.



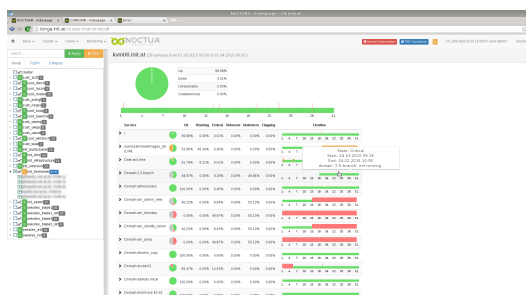
**Database snapshot** - Displays database snapshots at any point, comparison of snapshots. Displays historical status of users, devices, networks and scripts.

no  
illustration  
available  
yet

 **Notifications** - Notifications for status, error and warnings. Notifications by mail and sms. Many settings for periods, intervals and thresholds. Notifications dependent on user and groups.



**Reporting** - Clear and extensive visual display of availability for different hosts and services in defined timerange. Displays all status messages for hosts, services and even devicegroups. Simple handling of time span.



## 2.3. System Requirements


In this section you will find out about what technical requirements **NOCTUA®** has.

### 2.3.1. Common requirements

Like every other software **NOCTUA®** has also certain system requirements. Because **NOCTUA®** is open source software, everybody who has enough programming knowledge could be able to port it to other open source software systems.

The good news: You don't have to port anything if you already use one of the following **LINUX** distributions:


- Debian
- Ubuntu
- CentOS
- Opensuse
- SLES

For exact versions please take a look into  Installation Chapter.



## 2.3.2. Database requirements

Monitoring configurations are stored in databases for faster access and therefore faster reaction time and further more flexible administration of data.

NOCTUA® is using  *Django* as its database interface so every database which is compatible with Django can be used. The recommended Database is PostgreSQL (mostly due to license issues).

MySQL database is not supported any more.



---

# Chapter 3. Installation

## 3.1. Supported operating systems

Software packages are available for following operating systems:

- Debian Squeeze (6.x)
- Debian Wheezy (7.x)
- Ubuntu 12.04
- CentOS 6.5
- openSuSE (12.1, 12.3, 13.1)
- SLES 11 (SP1, SP2, SP3)

It might be very well possible to get the software up and running on different platforms - but this type of operation is not tested and supported at all.

### 3.1.1. Repository access

There is no public access to our repository directories, therefore first, you have to contact us to get a valid **LOG-INNAME** and **PASSWORD**.

Contact us by mail either **support@init.at** or **noctua@init.at** or just call **+43-1-522 53 77** to get your individual access data.

After receiving you access data you are able to use below mentioned repositories.

### 3.1.2. Two different ways to install the software


To install the software on your operating system there are two ways to go.

You can either automatically install the software by downloading and running our **install\_icsw.py** install script or manually by adding the below listed repositories with your individual access data and use your package manager as usual.

The automatically installation via script is well recommended because it is very comfortable and handles most of installation scenarios.

⚠ Please only do a manually installation if you really know what you are doing.

### 3.1.3. Automatically installation with *install\_icsw.py*

In order to use the install script, first you have to contact us to get your individual  access data.


1. Download the script named `install_icsw.py` from our download portal.
2. As user **root** run the script with your repository access data as follows:

```
install_icsw.py [ -u USERNAME ] [ -p PASSWORD ] [ -n CLUSTERNAME ]
```

The script does following:

1. Determine which operating system is running
2. Add the necessary repositories with the required access data in your operating systems repository directory
3. Refresh your package cache
4. Install the software
5. Automatically integration of valid license files to the system

### 3.1.4. Manually installation

After receiving your individual  access data do following steps to install the software manually:

1. Add suitable repositories for your operating system
2. Refresh your package cache with your os package manager
3. Install the software (for details see section installation)
4. Integrate your received or downloaded license files

### 3.1.5. Repository

There are two main repositories you have to deal with to install the software.

Repositories are available for stable (2.5) and for master (devel) versions of above mentioned operating systems. The operating system running on your hardware and the version of the software you want, determine which repository configuration you must use for your package manager.

#### icsw-devel

Repository for **latest** releases of **init cluster soft ware**.

The current developer Version, containing the newest functions and modules. Very fast update and change cycle due to active development. Sometimes a bug could slipped in but usually it works fine. From time to time it will be merged into stable.

#### icsw-2.5

Repository for **stable** releases of **init cluster soft ware**.

The current stable version for productive environment. Most features and functions are included and there are no knowing bugs.

Based on the above mentioned operating system, repository and desired software version, resulting repositories can be added.

### 3.1.6. Repository directories

There are two different ways to add new repositories for monitoring software by **init.at ltd.** to the operating system. It can be added all at once in one central file or in a repository directory. For each operating system there are special repository directories.

**Table 3.1. For debian based systems**

<b>Debian wheezy</b>	/etc/apt/sources.list.d/
<b>Debian squeeze</b>	/etc/apt/sources.list.d/
<b>Ubuntu 12.04</b>	/etc/apt/sources.list.d/

Table 3.2. For suse based systems

OpenSUSE	/etc/zypp/repos.d/
SLES	/etc/zypp/repos.d/

Table 3.3. For red-hat based systems

CentOS	/etc/yum.repos.d/
--------	-------------------

### 3.1.7. Debian repositories

#### One file repository

Following you can see some examples of `source.list` content. This are the lines you must add to your `/etc/apt/sources.list`

##### icsw-devel

The relevant parts for deb based package manager looks like this for **devel** version and **wheezy** :

```
deb      http://LOGINNAME:PASSWORD@www.initat.org/cluster/DEBs/debian_wheezy/
icsw-devel wheezy main
```

##### icsw-2.5

The relevant parts for deb based package manager looks like this for **stable** version and **wheezy** :

```
deb      http://LOGINNAME:PASSWORD@www.initat.org/cluster/DEBs/debian_wheezy/
icsw-2.5 wheezy main
```

### 3.1.8. Ubuntu repositories

For ubuntu 12.04 you can also add repositories either in one file, into `/etc/apt/sources.list` or in repository directory `/etc/apt/sources.list.d`

#### One file repository

##### icsw-devel

```
deb      http://LOGINNAME:PASSWORD@www.initat.org/cluster/DEBs/ubuntu_12.04/
icsw-devel precise main
```

##### icsw-2.5

```
deb      http://LOGINNAME:PASSWORD@www.initat.org/cluster/DEBs/ubuntu_12.04/
icsw-2.5 precise main
```

### 3.1.9. OpenSUSE and SLES repositories

Debian and ubuntu use an other package manager than CentOS, OpenSUSE or SLES. For that reason, on rpm based operating systems `sources.list` does not exist. Rather there are a few files for repository management not only one. All relevant repository files stays in the directory `/etc/zypp/repos.d/`.

## SUSE 13.1 and icsw-devel version

```
[cluster_devel_remote]
name=cluster_devel_remote
enabled=1
autorefresh=0
baseurl=http://LOGINNAME:PASSWORD@www.initat.org/cluster/RPMs/suse_13.1/icsw-devel
type=rpm-md
```

## SUSE 13.1 and icsw-2.5 version

```
[cluster_devel_remote]
name=cluster_devel_remote
enabled=1
autorefresh=0
baseurl=http://LOGINNAME:PASSWORD@www.initat.org/cluster/RPMs/suse_13.1/icsw-2.5
type=rpm-md
```

## Direct links to repositories

Alternative it's possible to download repositories direct from internet instead of editing files manually. There are two URLs you can get repositories from.

- For **deb** repositories look at  [http://www.initat.org/cluster/DEBs/]
- For **rpm** repositories look at  [http://www.initat.org/cluster/RPMs/]

Don't forget to  request for access data to the repository directories otherwise you can not access it.

## 3.1.10. CentOS and RHEL repositories

Repository directory is `/etc/yum.repos.d/`. Place your desired \*.repo files inside this directory, do a **yum check-update** and you are ready to install the software.

## CentOS 6.5 icsw-devel

```
[initat_cluster]
autorefresh=1
enabled=1
type=rpm-md
name=initat_cluster
baseurl=http://LOGINNAME:PASSWORD@www.initat.org/cluster/RPMs/rhel_6.2/icsw-devel
```

## CentOS 6.5 icsw-2.5

```
[initat_cluster]
autorefresh=1
enabled=1
type=rpm-md
name=initat_cluster
baseurl=http://LOGINNAME:PASSWORD@www.initat.org/cluster/RPMs/rhel_6.2/icsw-2.5
```

### 3.1.11. Database setup

Before continuing the server installation it is worth to say something about the database because it is one of most important parts of the software. It contains all settings, configurations, users and much more in one single database. This is the reason why it is easy to migrate or backup data and this is why it lowers the monitoring effort in comparison to a simple icinga installation.

#### POSTGRESQL

After a basic installation of the server, normally only a SQLite database exists. To start with monitoring it suffices completely but it must be mentioned that due to database limitations of SQLite we recommend you to switch to a better database solution e.g. SQL.

So before installing the software, two scenarios are possible.

1. SQL Database already exists
2. SQL Database does not exists

The server can handle both states, in case of an existing database we want the server doing some migrations for us, in the other case we want to do an initial database setup.

#### Note

Please keep in mind that *icsw setup* neither installs a new database nor configure your local database settings.

It links your SQL database with the server environment.

*icsw setup* runs in an interactive mode and is responsible for a couple of basic setting:

- Creates suitable database schemata
- Creates an administrator account and an initial password for first login into the web front-end
- Creates the database config file `/etc/sysconfig/cluster/db.cf`

The script asks you for

1. Install **postgresql-server**
2. Install **python-modules-psycopg2**
3. Either run following command to migrate an former existing database:

**`icsw setup --ignore-existing`**

or run following command to create a completely new database:

**`icsw setup`**

After running *icsw setup*, the script awaits some input from the admin.

```
possible choices for DB engine: psql, sqlite
DB engine (psql)                :
DB host (localhost)              :
DB user (cdbuser)                :
```

```
DB name (cdbase)           :
DB passwd (bTMZPUYMiR)     :
DB port (5432)             :
```

Take the suggested defaults with the "ENTER" key or insert own data. By now you have to have a faultless PostgreSQL installation setup to allow the software to connect to the database.

In case something goes wrong the script displays possible steps have to be done.

Most of conflicts at this time are wrong permissions to the database or generally a wrong database installation and setup. If below steps can not solve the problem please take a look into your database manual or ask the database administrator to find out how to setup database with correct permissions.

Login into your database (commonly done with **su postgres** followed by a simple **psql**) and type following commands to create the right user and a new and empty database:

```
CREATE USER cdbuser LOGIN NOCREATEDB UNENCRYPTED PASSWORD 'my_password_123';
CREATE DATABASE cdbase OWNER cdbuser;
```

This will create a new database user with the name **cdbuser**, the desired password **my\_password\_123** and a new, empty database called **cdbase**

After successful creation of database user and database, we have to edit `/var/lib/pgsql/data/pg_hba.conf` (OpenSUSE) to setup correct permissions to the database. Comment out other lines so you have only the below three remaining.

```
local    cdbase          cdbuser                               md5
host     cdbase          cdbuser          127.0.0.1/32          md5
host     cdbase          cdbuser          ::1/128             md5
```

To be on the safe side it is recommended trying to login manually to the database. If you are able to connect manually likely the script is able too.

After everything goes well we get a successful connection message

```
dsn is 'dbname=cdbase user=cdbuser host=localhost password=bTMZPUYMiR
port=5432'
connection successful
```

Once connected successful to the database the script runs the migration for you. Finally you have an installation with a PostgreSQL database.

The database access data for the server is stored in `/etc/sysconfig/cluster/db.cf` created by *icsw setup*, a sample file is provided under `/etc/sysconfig/cluster/db.cf.sample`. If you want to connect via local socket leave `DB_HOST` empty. Either fill in the user and database information manually or run *icsw setup* for an assisted config file creation.

Every daemon and process from the server is using this file to gain access to the database. The File has to be readable for the following system entities:

- The user of the uwsgi-processes (wwwrun on SUSE systems)
- The system group idg



A typical set of rights would look like

```
-rw-r----- 1 wwwrun idg 156 May 7 2013 /etc/sysconfig/cluster/db.cf
```

## Manually database backup

Although the software do periodic backups, it could be necessary to do a database backup by hand. For PostgreSQL there is a special dump command:

```
pg_dump -Fc -U cdbuser cdbase > DATABASE_BACKUP_NAME
```

This single line is enough to copy your whole database to a file.

## Manually database restore

### Important

⚠ ATTENTION: ALL STORED DATA WILL BE LOST AFTER DROPPING YOUR DATABASE

A few more action is needed to restore database backup. First of all, before we are able to restore our database backup cleanly, we have to drop (delete) all database contents. After database contents are dropped we are able to import data into the existing and empty database.

Delete database contents:

```
su postgres -c "psql -c \"DROP DATABASE cdbase; \""
```

Restore database:

```
pg_restore -c -C -F c DATABASE_BACKUP_NAME | psql -U postgres
```

## 3.1.12. Portnumber for accessing the webfrontend

The Webfrontend for your server can be accessed via

```
http://SERVERNAME/cluster or by http://IP_ADDRESS:80/cluster/
```

The Webfrontend for your server can be accessed via

```
http://SERVERNAME/cluster or by http://IP_ADDRESS:80/cluster/
```

In case you run `setup_noctua.sh` script manually, the portnumber will be rewritten to **18080**. So you can access the web front-end by this url:

```
http://SERVERNAME:18080/cluster or http://IP_ADDRESS:18080/cluster
```

Also you can use your server localhost alias for accessing the front-end:

```
http://localhost:18080/cluster or http://localhost:18080/cluster
```

## 3.2. Installation

### 3.2.1. Install icsw packages

After added desired repositories for your operating system it is time to install the software packages itself and configure it. There are three main packages you have to install to get a basic server running:

1. **icsw-server**
2. **icsw-client**
3. **icsw-dependencies**

These packages contains all necessary services, binaries, libraries and dependencies for a clean and proper Installation.

If you also want to get access to the server by web GUI, you additionally need to install the **nginx-init** package and run the nginx-init http server.

For SUSE operating systems a installation command should look like following one:

**zypper ref; zypper install icsw-server icsw-client icsw-dependencies**

While accessing the repositories you will be prompted for valid username and password. Type in your received access data on the terminal and continue installation.

### Postinstall

To guarantee the maximum possible flexibility, we decided to involve the system administrator into the installation procedure. After installation you will get a note on **stdout** how to create a new database configuration.

*icsw setup* command

### Components of icsw-server

This is the basic package you have to install to run a server

### Components of icsw-client

This is the basic package you have to install to run a client

### Components of icsw-dependencies

This package provides system wide dependencies the other both can not provide.

## 3.3. Installation on virtual machine

Alternative to usual installation of binary packages via repositories and the operating system package manager like **zypper** , **apt-get** or **yum**, you can use a virtual machine with an ready to go installation. We distribute two popular VM image file formats running with *libvirt/qemu* and *vmware* . For information how to set up your VM environment, please take a look at the corresponding documentation of your VM vendor.

### 3.3.1. KVM libvirt/qemu

Following steps have to be done to run a KVM libvirt/qemu virtual machine with preinstalled **NESTOR®**

1. Download the KVM/libvirt image and move it into the right image directory e.g. `/usr/local/share/images/`.
2. Copy an existing \*.xml or create a new one
3. Edit your new \*.xml file
4. Define your new virtual machine

For virtual machine installation it could be necessary to do individual network setup (bridge) of your VM host.

Finally if your machine is setup correct, only you have to do is to start the virtual machine and have fun with monitoring.

### 3.3.2. VMware

You have to convert our virtual machine image for *kvm/qemu* into an suitable format for VMware.

## 3.4. Upgrade software packages

From time to time, new software packages were built and can be downloaded. Especially for the master development branch there are frequent updates which can be applied to get new functions or features or simply fixing some bugs. Update period for master is about every second day.

The stable branch gets less frequent updates than the master version. Because it is the stable branch, most updates for stable affected security issues und bugfixes. Really big updates are done only if the master is stable enough for productive environment. The update period time is about 4-6 month.

The update procedure is very comfortable, it based on the system integrated package manager, for example **zypper** in OpenSUSE or **apt-get** in debian.

Comands for updating/upgrading **all** installed software by package manager are:

**zypper ref; zypper dup** Refresh repositories and do whole system upgrade in OpenSUSE

**apt-get update; apt-get dist-upgrade** Refresh repositories and do whole system upgrade in debian

Of course, you are also able to only update single packages, for example the package **handbook-init** . The command looks similar to the command used to update all packages:

**zypper ref; zypper up handbook-init** Refresh repositories and do single package upgrade, in this case upgrade of package **handbook-init** in OpenSUSE

**apt-get update; apt-get upgrade handbook-init** Refresh repositories and do single package upgrade, in this case upgrade of package **handbook-init** in debian

⚠ It is not recommended to update only one single package except you know exactly what you are doing.

For other distributions please look into your distributors package management description.

## 3.5. Server control

After successful installation of **NESTOR®** first of all you have to check if all necessary services are running. Type following command into the terminal:

### 3.5.1. Show server status

#### icsw service status

Get more information about possible flags with: **icsw --help**

One of the most common flag is **-v**. This shows additionally the version number of each package like shown below.

**Figure 3.1. icsw status command**

```
Wed, 20. May 2015 20 15:21:28
Name      type    source License status
-----
hoststatus      client simple -    running
logging-server  client meta -    running
meta-server     client meta -    running
host-monitoring client meta -    running
package-client  client meta -    running
logcheck-server server meta -    running
package-server  server meta valid not configured
mother          server meta -    running
collectd        server meta valid running
rrd-grapner     server meta valid running
rms-server      server N/A  valid not configured
cluster-server  server meta -    running
discovery-server server meta valid running
cluster-config-server server meta -    running
host-relay      server meta -    running
snmp-relay      server N/A  -    error
md-config-server server meta valid running
memcached       system simple -    running
nginx           system simple -    running
icinga          system simple -    running
uwsgi-init      system simple -    running
rrdcached       system pid  -    running
```

An other common flag is **-a**. With this flag, the script shows additional information:

- Thread info
- pids
- runlevels
- Memory


Take a look into our  **command reference** to learn more about *icsw* command.

### 3.5.2. icsw commands

The main command, which can be used to manage the cluster components, is the **icsw** command. The following table present the frequently used examples of the *icsw* tool:

**Table 3.4. icsw - command overview**

icsw-command	Functionality
<code>icsw service { status, start, stop, re-start, debug, state }</code>	Show status or control icsw services

icsw-command	Functionality
<code>icsw state { overview,enable,disable }</code>	<p>Show the state overview or enable/disable services</p> <p>These are the service states the which are managed by the meta-server</p>
<code>icsw logwatch [ -f ] [ --system-filter ] system-name</code>	<p>The logwatch command is intended to show logging messages to stdout. The <b>-f</b> option is used to append data while the file grows. The <b>--system-filter</b> option limits the logging output to a specific service. If used without any arguments it displays logging messages for all running services.</p>
<code>icsw license { show_cluster_id,register_cluster,lock,unlock,show_locks }</code>	<p>With the icsw license command administrators are able to lock, unlock or show locked licenses and devices from the license system.</p> <p>You can also show your cluster ID or register a cluster.</p> <p>For more information about the lock or unlock command take a look the section called “ Lock command to fall below the parameter limitation ”</p>
<code>icsw setup [ service-name ]</code>	<p>Create database and perform initial setup. There are many options and arguments for this command, please take a look into the  for further information.</p>

Short overview about icsw commands

**Table 3.5. icsw - service**

icsw-command	Functionality
<code>icsw service status [ service-name ]</code>	<p>Displays the status of the server and all of its services</p> <p>With the "service-name" it displays only status of given "service-name"</p>
<code>icsw service start [ service-name ]</code>	<p>Starts the service with "service-name". Without service-name option the command initiates the start of all cluster components.</p> <p>Warning!: if the service is disabled in the meta-server, then in some minutes the meta-server will stop the service again.</p>
<code>icsw service stop [ service-name ]</code>	<p>This command stops the service with the "service-name" or all services of cluster instance.</p> <div style="border: 2px solid #90EE90; padding: 10px; margin-top: 10px;"> <p><b>⚠Warning!:</b> Be sure, that the service is disabled in meta-server. Otherwise the service will be start by meta-server in few minutes again.</p> </div>

icsw-command	Functionality
<code>icsw service restart [ service-name ]</code>	This command restarts the service with the given "service-name" or, if no service-name is used, it restarts all services of cluster instance.
<code>icsw service debug { service-name }</code>	Using the debug option is like to start a service in foreground. In contrast with services started as daemon (background), the service probably displays some stdout messages.
<code>icsw service state overview [ service-name ]</code>	Provides the state/status info about all services or specific service, if there is the "service-name"
<code>icsw service state enable { service-name }</code>	Enables the service in meta-server settings(database). That means, the meta-server is responsible for running of the service. If the service is not active, in few minutes the meta-server will start the service again.
<code>icsw service state disable { service-name }</code>	Disables the services in meta-server settings (database). Henceforward the meta-server controls, if the service is not running.

The icsw **service** command

## 3.6. License administration

All core features, also called basis features, of the software are licensed under an open source license and for free but some enterprise features of **NESTOR®** are not. There are licenses for this enterprise features and you have to buy licenses in order to get the features working.


This section guides you through the process of getting licenses, understanding the concept of license limitation and managing licenses.

### 3.6.1. Login without licenses

If you never have applied licenses to your server then you are running an unlicensed software version. To remind you of this fact you will see notification messages on some pages, for example on the login page and also on the dashboard after succesfull login.

#### Figure 3.2. Unlicensed notification

You are running an **unlicensed version**.  
Please contact **support@init.at** in order to obtain a license.

Also you can check your license state by navigating to **Session  License**.

There you will see three different drop down windows:

- Your licenses for this cluster
- License packages
- Upload license file

**Figure 3.3. License overview**License overview (ClusterID is **8E9YKSLA-2W446**)

Your licenses for this cluster

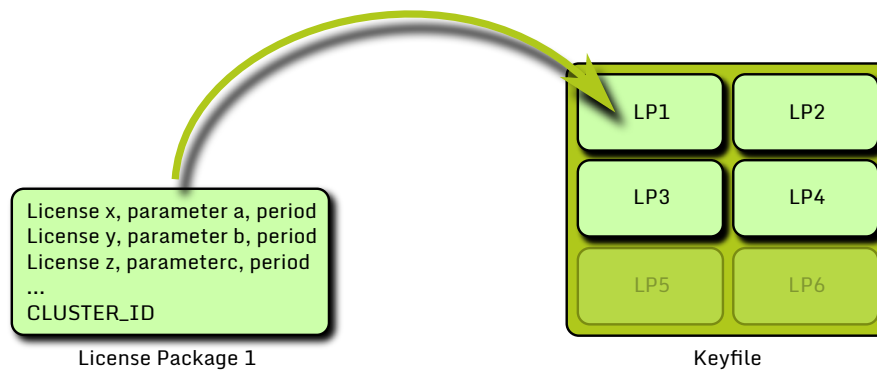
License packages

Upload license file

No file chosen

### 3.6.2. License package structure


Shipped Licenses are keyfiles containing information about licensed features, license period and license parameter. Each license is associated to one specific Cluster ID. A keyfile can contain one or more license packages assigned with one or more cluster IDs.

**Figure 3.4. License package**

The license package dropdown window shows you an content overview of uploaded keyfiles.

Keyfiles containing more than one single license package will be displayed in separate tabs inside of the **License package** dropdown window.

### 3.6.3. Upload license keyfile

You can't expand any dropdown menu by  left clicking on the arrow beside the dropdown except the **Upload license file** one. Use the **Choose File** button to select your valid license keyfile. After selecting your keyfile its name will be displayed on the right side of the button:

**Figure 3.5. License overview**License overview (ClusterID is **8E9YKSLA-2W446**)

Your licenses for this cluster


License packages

Upload license file

2015-05-13.keyfile

Push the **upload** button to integrate your valid license keyfile into the server and activate acquired enterprise features.

### 3.6.4. Show license status

After uploading your valid license keyfile to the server, your license overview immediately will be updated and shows your purchased licenses. Generally, if you want to see your license overview navigate to **Session**  **License** to display the license status.

**Figure 3.6. icsw status command**

License overview (ClusterID is `19PT22G-SMR-V`)

Your licenses for this cluster						
License	Description	Parameters	Status		Type	License package
MD Config Server	Monitoring Daemon Configuration Writer	Device: 100 (used: 0) Service: 1000 (used: 0)	✓ Valid	until 2015-06-21	Test	Test package NOCTUA 21.05.2015
Key Performance Indicators	Calculate key figures to measure the performance of your cluster					
Graphing	Comprehensive graphical evaluation using RRDs	Device: 100 (used: 3)	✓ Valid	until 2015-06-21	Test	Test package NOCTUA 21.05.2015
Discovery Server	Automatical configuration using SNMP	Device: 100 (used: 1)	✓ Valid	until 2015-06-21	Test	Test package NOCTUA 21.05.2015
Monitoring dashboard	Central monitoring unit consisting of livestatus, geolocation and maplocation	Device: 100 (used: 9) Service: 1000 (used: 81)	✓ Valid	until 2015-06-21	Test	Test package NOCTUA 21.05.2015
Distributed Monitoring	Distribute monitoring load to multiple workers					
Database snapshot	Keep track of the configuration changes	Device: 100 (used: 13)	✓ Valid	until 2015-06-21	Test	Test package NOCTUA 21.05.2015
Notifications	Status notifications via mail and text messages	Device: 100 (used: 0) Service: 1000 (used: 1)	✓ Valid	until 2015-06-21	Test	Test package NOCTUA 21.05.2015
Network Weathermap	Overview of relevant network utilization data					
Rootkit Hunter	Security scan for installations					
Reporting	Generate graphs to view the state of your cluster					
License Optimization Management	Interactive graphic license utilization evaluation					
Virtual Desktop	Manage virtual desktop sessions on your cluster					
HPC Workbench	Convenient interface for job management					
Package Install	Configure repositories and install packages on your nodes					
Resource Management System	Overview over your job system					
Netboot	Manage the boot process of your nodes					

#### License

License name

#### Description

Description of the specific license


#### Parameter

Parameter value is the limitation of licenses in context of

- Device
- Service
- User
- External license

Used licenses and amount will be displayed as info window

**Figure 3.7. icsw license service used**

 Service: used: 18

#### Status

**Valid in future**, license will be valid from point of time in future

**Valid**, license is active and valid until displayed date

**Grace**, license is in grace time. It is still active until the grace timeperiod of **2 weeks** is over.



**Expired**, license is out of grace time period.

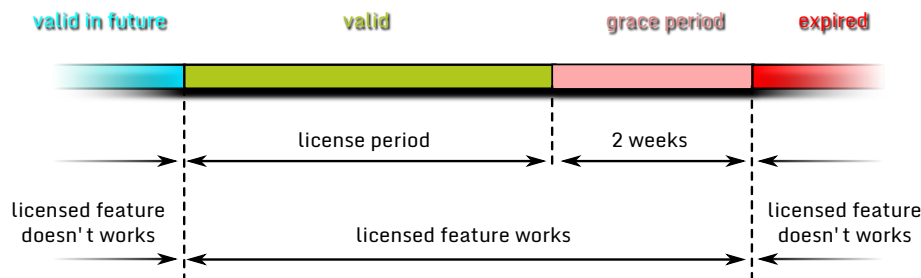
License package

Name of used license package

### 3.6.5. License time periods

There are four different periods or states of licenses. Dependent of the period or the state a licensed feature is working or not.

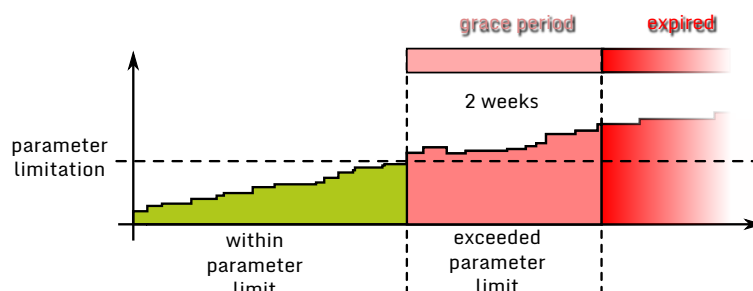
**Figure 3.8. License states**



### 3.6.6. License parameter limitations

The second factor which decides if a license is valid or not is the parameter limitation. Dependent of purchased parameter amount and of used parameter the license can be valid, in grace time or expired.

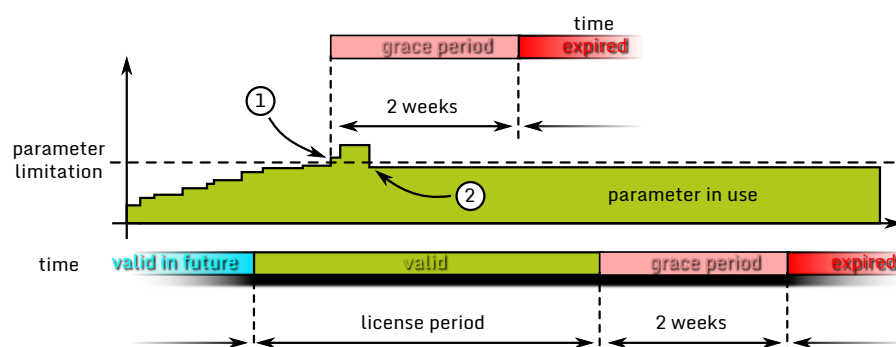
**Figure 3.9. License states**



### 3.6.7. License violation

Licenses can be violated by exceeding the license time period or by exceeding one of the license parameters limitation. In case of exceeding the license parameter limitation also a grace time period starts which is totally independent of the license time period grace time.

**Figure 3.10. License states**



## Lock command to fall below the parameter limitation

For a small violation of parameter limitation there is a **lock** command to get back within parameter limit.

In Figure 3.10, “ License states ” marked with (1) you can see a license violation by exceeding the parameter limitation. Because of a violated license the grace time period starts immediately. In your license overview you will notice a warning message for this violated license.

Now, you have two options.

1. Purchase an extended license for that feature to increase the parameter limitation
2. Lock the license parameter to get back below parameter limitation

There is a special command for the second option:

```
icsw license lock { -d DEVICE } { -l LICENSE }
```

After locking licenses [illustrated on (2) of Figure 3.10, “ License states ”], the **used** license amount falls below the parameter limit and is valid again.

If you want to know if any licenses are locked, use following command:

```
icsw license show_locks
```

It displays all locked licenses and corresponding devices.

---

# Chapter 4. Core concepts

## 4.1. Services and components

This sections explains the core concepts behind **NOCTUA®**. It gives a top level overview of it's components and capabilities.

Following some separate components are listed and to which part of the software it belongs to.





### 4.1.1. We do not need to reinvent the wheel

As relative complex software, **NOCTUA®** use some well known frameworks and technology we absolutely have to mention.

#### Used Frameworks

django	Web application framework written in python.
angularjs	Opensource framework developed by google™.
bootstrap	CSS Framework developed by twitter™

#### Used software solutions

 <i>icinga</i>	Industy standard software for monitoring devices.
 <i>rrd-tool</i>	RRDtool is the OpenSource industry standard, high performance data logging and graphing system for time series data. RRDtool can be easily integrated in shell scripts, perl, python, ruby, lua or tcl applications.
 <i>nginx</i>	Powerfull small and fast webserver.
 <i>SGE</i>	Son of grid engine.

### 4.1.2. Common components

Parts which both software packages needs to works properly.

#### Components

meta-server	This daemon is responsible for restarting services that might have crashed or were otherwise killed. This functionality should be taken over by systemd. <code>/var/lib/meta_server</code> contains the relevant information about which services should be running.
logging-server	Creates the structure needed for receiving logs via rsyslog or syslog-ng.

### 4.1.3. Monitoring components

Monitoring in context of hard- and software will be practiced to get information about specified systems.


#### Components

md-config-server	Responsible for configuration of icinga. Interacts with database.
------------------	---



cluster-server	Responsible for writing config files and general coordination of the cluster. Listens on port <b>TCP/8004</b> . cluster-server.py is a general purpose server that handles various tasks like writing /etc/hosts, generating a valid DHCP configuration, configuration of the BIND nameserver, feeding LDAP and / or YP servers, ...
collclient.py	Client part of the host-monitoring collserver.py.
ccollclientzmq	Frontend program to talk to collrelay.

#### 4.1.4. HPC components

**NOCTUA®** consist of many different parts and services. Each of these services perform a specific set of tasks in the cluster. Most of these services are network enabled and listen on a specific port for commands. The following list tries to give an overview about the most important parts.

The general idea of **NOCTUA®** is simple. Create an image, a kernel and a set of configuration files to be used by your nodes and distribute them to the nodes. The distribution is done via  **PXE**. **NOCTUA®** enables you to describe the node specific configurations in Python.

##### Components

cluster-config-server	Generates the files for the Clusternodes (based on the config stored in the Database) to make the nodes distinguishable.
mother	Creates the tftpboot /ethernet structure, monitors the installation progress of the nodes. Listens on ports <b>TCP/8000 TCP/8001</b> . Mother provides access to  <b>IPMI</b> as well.
package-server	Provides repositories available for installation by the package-client. Listens on port <b>TCP/8007</b> .
package-client	Install required software by using the locally available package management commands. <b>zypper</b> , <b>yum</b> or <b>apt-get</b> .
hoststatus	A small program written in C that transmits node status messages to the cluster-server. The hoststatus is written in C to be easily includable in the initial ramdisk. It listens on port TCP/2002. hoststatus is in the package child
logcheck-server	Do log rotation and deletes logs older than a specified time range.
rms-server.py	Provides integration of <b>NOCTUA®</b> with  <b>SGE</b> . The commands <b>sns</b> and <b>sjs</b> rely on it.
discovery-server	Daemon for automatic configuration of devices.

#### 4.1.5. Clusterdatabase

The database where all the configuration data of **NOCTUA®** installation is stored is generally referred as the "clusterdatabase".

Throughout this document we might refer to:

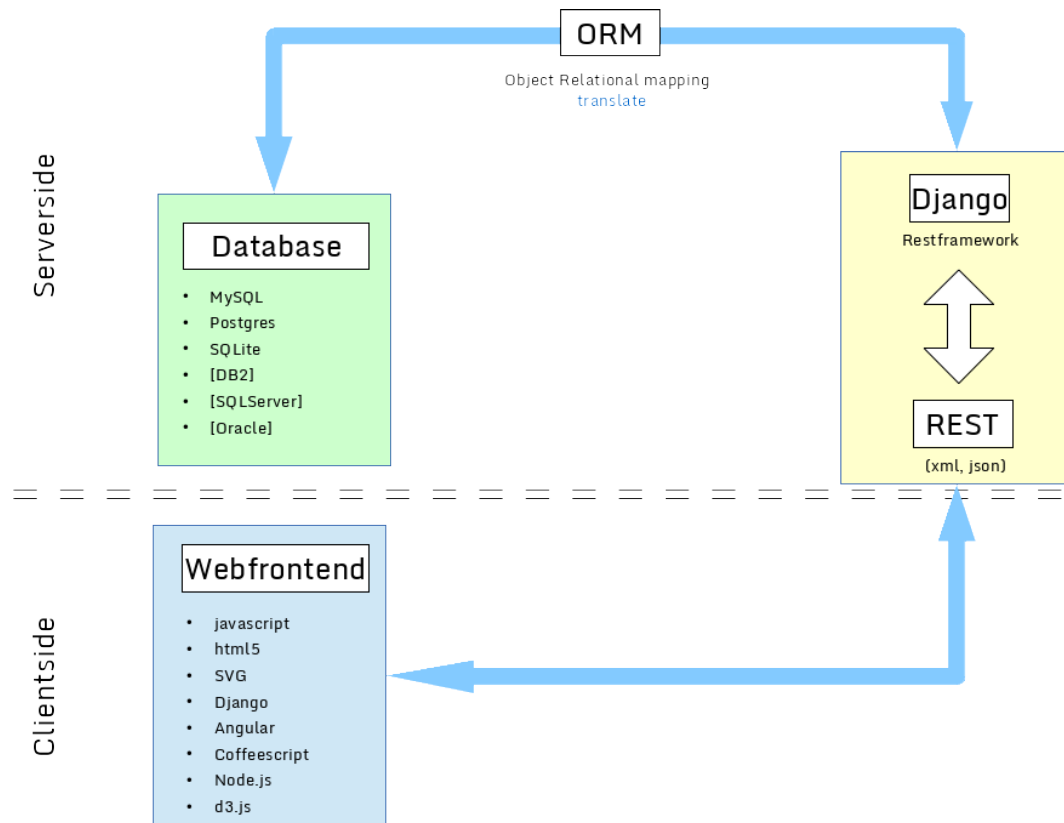
**Server side scripts** . Are generally services or scripts that are running on the clusterserver. Most of them need database connectivity to function properly.

**Node side scripts** . On the other hand, node side scripts are daemons or scripts that are generally run on a node of the cluster. Node side scripts don't require access to the cluster Database.

### 4.1.6. Dataflow between webfrontend and database

The graphic shows just the part between database and webfrontend, only a small but essential part of monitoring.

**Figure 4.1. Webfrontend and database dataflow**



Dataflow with help of django and django REST framework.

## 4.2. Run important services

Because **NOCTUA®** consists of many different parts working together, it is not obligatory to run every service at once. Services like **package-install** or **discovery-server** are not essential to operate monitoring or cluster management.

For that reason, default installation of **NOCTUA®** is rudimentary. Special services and functions **are not** activated by default. Activation of certain services force the user to push some buttons or move some lever.

Two spots where you can activate services are:

- cluster server information
- Device Config



---

# Chapter 5. Webfrontend

## 5.1. First connection

Most configuration in **NOCTUA®** administrators have to do, is accessible over a standard html compatible browser like Mozilla Firefox™ or Google Chrome™. Once **NOCTUA®** is installed and all required services are running, all you have to do is to connect to the server via browser.

Type in

```
http://SERVER-IP-ADDRESS:80/cluster/
```

or

```
http://SERVERNAME/cluster/
```

in your browser addressbar to connect to the server. If you connect the first time to the server you will be redirected to the account info page.

### Important

⚠ You really have to change your password now. If you don't change it, **NOCTUA®** takes his own during installation procedure generated password you never seen before and next time you can not log in.

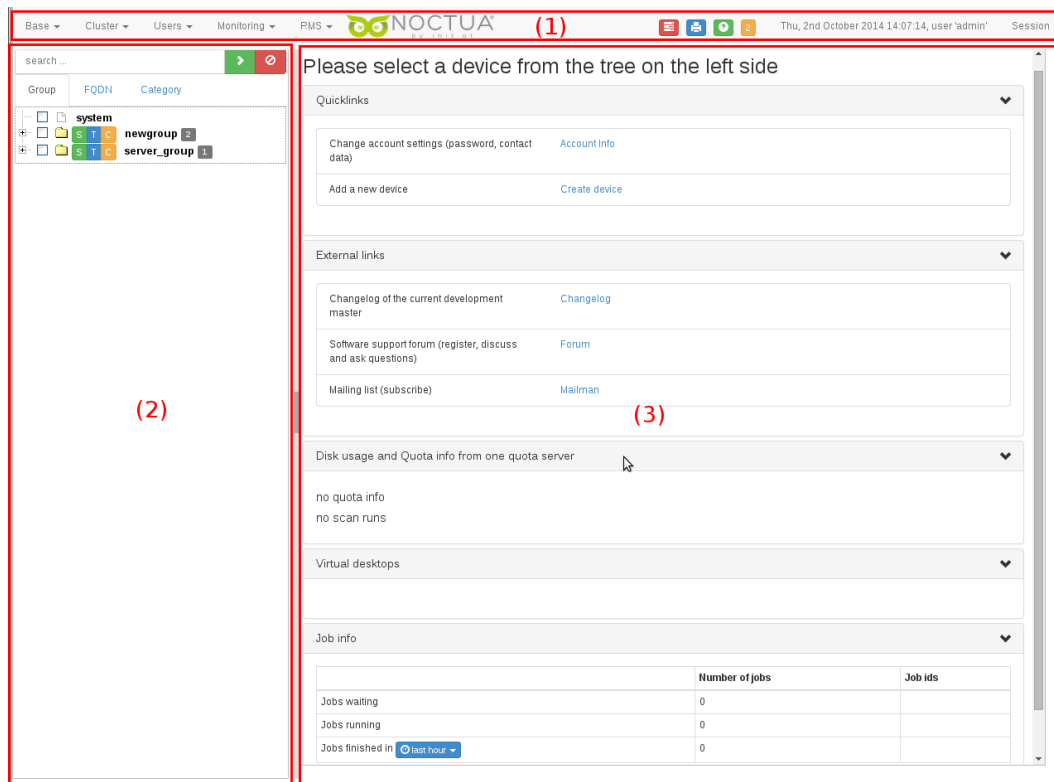
If you run the **setup\_noctua.sh** script manually, it generates you a new admin password. In this case you must look for the password in your shell output.

## 5.2. Areas

**NOCTUA®** webfrontend offers you a very clear view. There are three areas you will work with:

- Menu area (1)
- Sidebar device tree (2)
- Main area (3)

Figure 5.1. Three areas



Areas you'll see after login

### 5.2.1. Menu area (1)

In the menu area you'll find submenus, buttons, date, time and user section.

#### Submenus

1. Base
2. Users
3. Monitoring
4. Session

NOCTUA® offers some additional menus:

1. RMS - Resource management System
2. Cluster

#### Buttons

1. cluster server information
2. show cluster handbook as pdf
3. show index
4. number of background jobs



Figure 5.2. Menu buttons

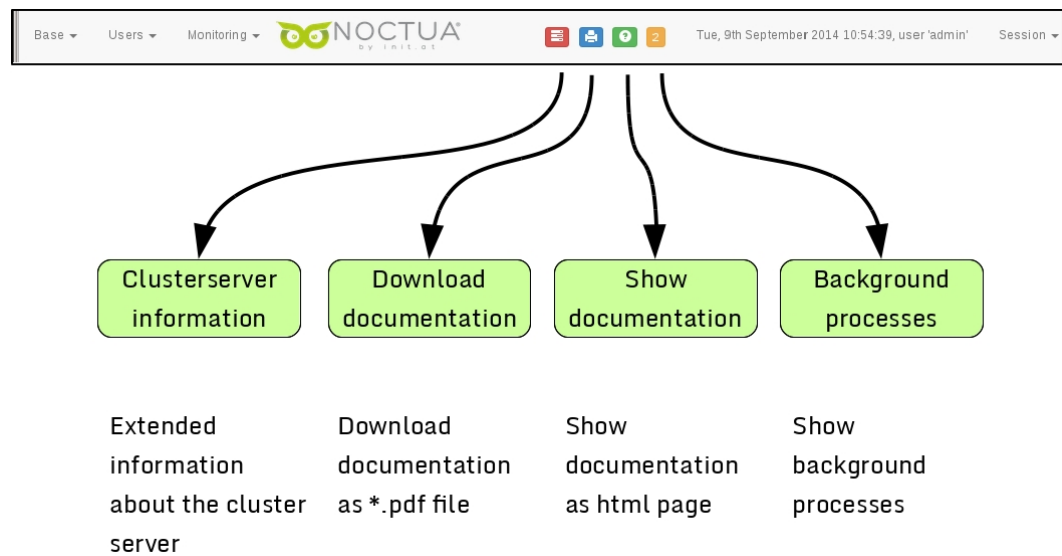
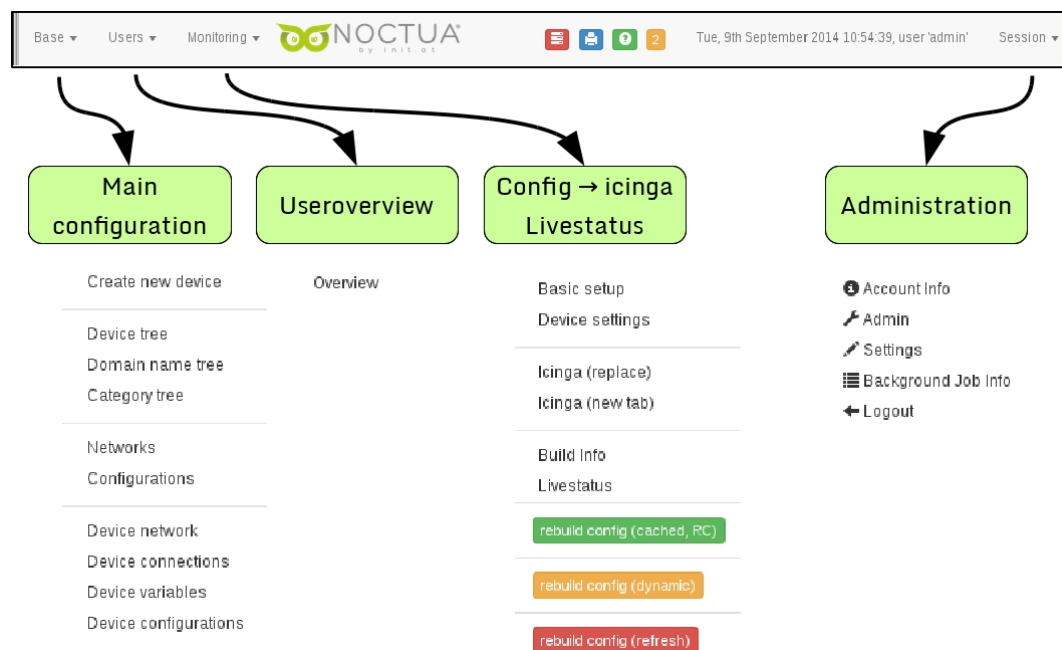
Buttons and submenus for **NOCTUA®**

Figure 5.3. Menus

Menus for **NOCTUA®**

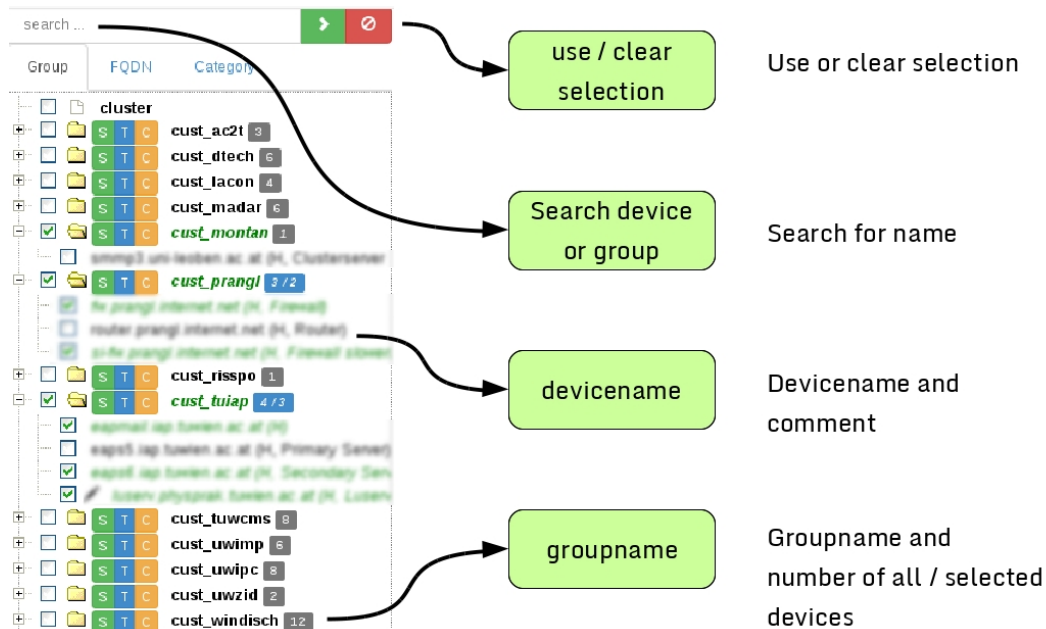
## 5.2.2. Sidebar device tree (2)

In the tree area you can find your device group tree and associated devices. Located on top, there is a searchfield and 2 buttons.

1. Searchfield
2. use selection Button (green with arrow)
3. clear selection Button (red with circle)

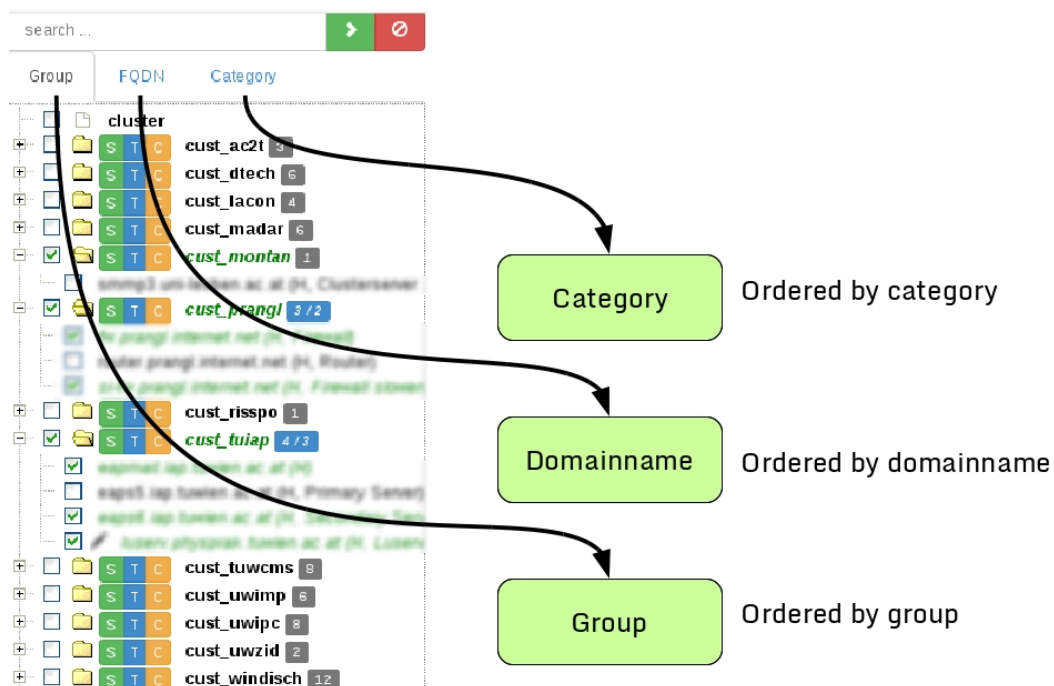
4. Group
5. FQDN (Full Qualified Domain Name)
6. Category

Figure 5.4. Devicetree

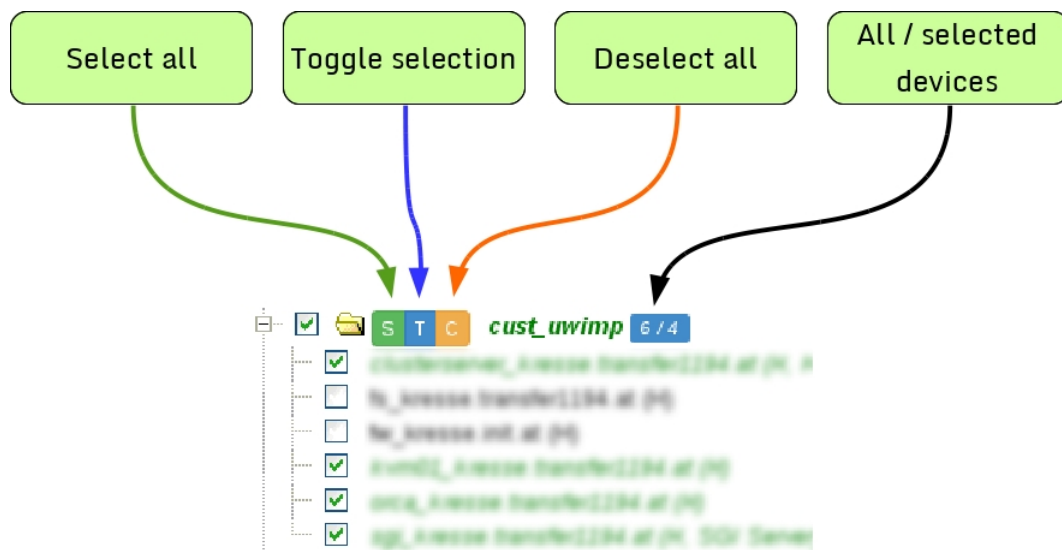


NOCTUA®

Figure 5.5. Devicetree

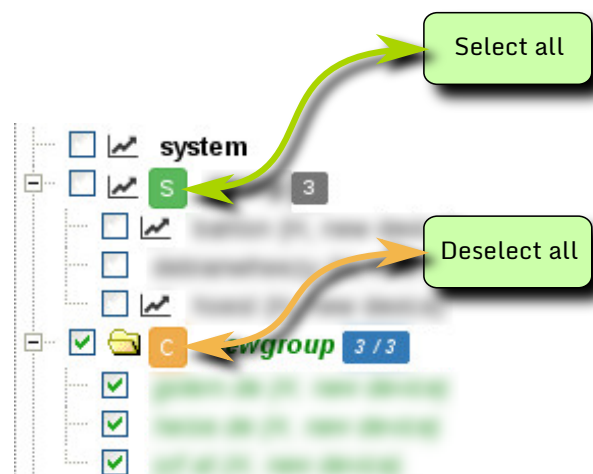


NOCTUA®

**Figure 5.6. STC Selection buttons**

Buttons to select, deselect or toggle selection

Alternativ there is an one button selection method.

**Figure 5.7. S/C Selection button**

One button selection, either select all or deselect all

### 5.2.3. Main area (3)

All the configurations and input takes place in the main area. According to the selected or preselected devices and settings, corresponding page appears.

Figure 5.8. Possible main area

Network config for 2 devices

Devname / IP	Bridge	MAC / Network	Devtype / DTN	routing / alias / enabled	action
2 / 2 / 2	cluster server _services_bridge11294.at	—	cust_uvimp	Kresse Clusterserver	Create new
1 / 1 eth0		00:00:00:00:00:00	eth (ethernet devices [6])	no (1) / yes / yes	info modify delete
1 / 1 lo		00:00:00:00:00:00	lo (loopback devices [6])	no (1) / yes / yes	info modify delete
2 / 2 / 2	server _services_bridge11294.at	—	cust_uvimp		Create new
1 / 1 eth0		00:00:00:00:00:00	eth (ethernet devices [6])	no (1) / yes / yes	info modify delete
1 / 1 lo		00:00:00:00:00:00	lo (loopback devices [6])	no (1) / yes / yes	info modify delete

Copy network from

One possible view of main area after select some devices in "device network"

## 5.2.4. Cluster server information

The cluster server information button shows three overview tabs, one tab with information about **defined cluster roles**, one tab with information about server results and one with information about the **server** itself.

Once called, **server information** restarts the **memcached** daemon.

### Cluster roles defined

Inside this upper tab, there is a table showing the **Name**, **reachableIP** and the defined **cost** of each of them. This tab is only for displaying information.

Each of the defined roles provides special functionality to the server.

### One server result

Also a tab only for displaying general information.

- valid
- name
- Result
- max Memory
- total Memory

### One Server checked

This is the only one tab inside of server information, which allows you to control something. You are able to control services as you do by command line.

Following information will be displayed:

**Server information**

<b>Instance</b>	Name of service
<b>Type</b>	Type of service {node, server,system}
<b>Check</b>	Kind of Check
<b>Installstatus</b>	Status if service is installed or not
<b>Version number</b>	Versionnumber of installed service
<b>Processnumber</b>	Number of processes started
<b>Memory usage</b>	Displays memory usage as number and as statusbar
<b>Action Buttons</b>	Button to apply action to the services

**Figure 5.9. Cluster server information**

Instance	Type	Check	icinga.init.at, 939.343 MB max / 3.389 GB total Memory				
hoststatus	client	simple	3.0–391	1	2 MB	<div><div></div></div>	<div>⛔ disable</div>
logging-server	client	meta	3.0–391	3	18 MB	<div><div></div></div>	
meta-server	client	meta	3.0–391	5	66 MB	<div><div></div></div>	
host-monitoring	client	meta	3.0–391	17	70 MB	<div><div></div></div>	<div>⛔ disable</div>
package-client	client	meta	3.0–391	6	16 MB	<div><div></div></div>	<div>⛔ disable</div>
logcheck-server	server	meta	3.0–391	not running			<div>✅ enable</div>
package-server	server	meta	3.0–391	11	153 MB	<div><div></div></div>	<div>⛔ disable</div>
mother	server	meta	3.0–391	not running			<div>✅ enable</div>
collectd	server	meta	3.0–391	28	499 MB	<div><div></div></div>	<div>⛔ disable</div>
memcached	system	simple		1	25 MB	<div><div></div></div>	
nginx	system	simple		9	18 MB	<div><div></div></div>	
icinga	system	simple		2	57 MB	<div><div></div></div>	<div>⛔ disable</div>
uwsgi-init	system	simple		16	939 MB	<div><div></div></div>	
rrdcached	system	pid_file		1	6 MB	<div><div></div></div>	
rrd-grapher	server	meta	3.0–391	11	249 MB	<div><div></div></div>	<div>⛔ disable</div>
rms-server	server	meta	3.0–391	not running			<div>✅ enable</div>
cluster-server	server	meta	3.0–391	11	214 MB	<div><div></div></div>	<div>⛔ disable</div>
discovery-server	server	meta	3.0–391	23	169 MB	<div><div></div></div>	<div>⛔ disable</div>
cluster-config-server	server	meta	3.0–391	not running			<div>✅ enable</div>
host-relay	server	meta	3.0–391	9	94 MB	<div><div></div></div>	<div>⛔ disable</div>
snmp-relay	server	meta	3.0–391	18	215 MB	<div><div></div></div>	<div>⛔ disable</div>
md-config-server	server	meta	3.0–391	27	653 MB	<div><div></div></div>	<div>⛔ disable</div>

Backgroundinformation about running or stopped services


# Chapter 6. Concept of operation

## 6.1. Working with the web front-end

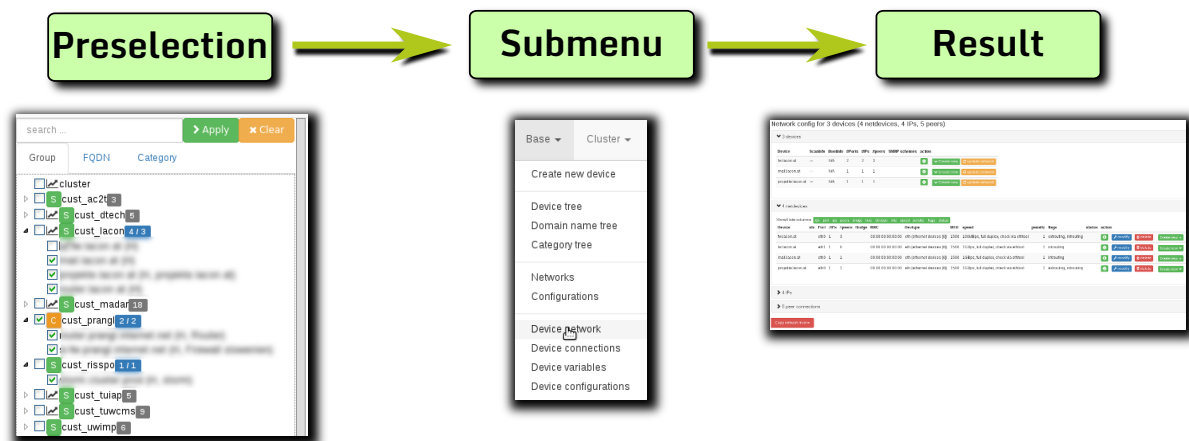
The workflow inside of the web front-end follows a special pattern. This workflow repeats for specific actions and therefore it is worth to mention and learn it. We divide this section into four subsections listed below to show the difference of each one. Of course it is possible to get similar results by different ways but each way have advantages and disadvantages or is more or less efficient.

There are also software regions like **Nodeboot** which are only accessible by one single way.

### 6.1.1. Preselection and submenu

All preselections are done in the sidebar device tree Section 5.2.2, “ Sidebar device tree (2) ”. Select groups or devices and click  in top menu on desired submenu to access .

**Figure 6.1. Submenu method**




Device network for preselected devices as result

Submenu method is recommended for working with multiple devices.

Following areas can be accessed by the submenu method:

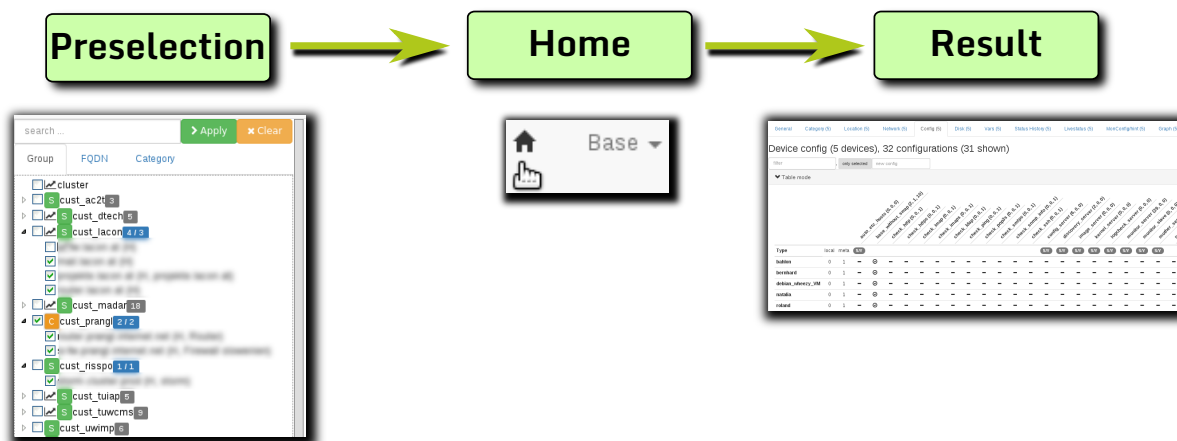
- Device Tree
- Device Variable
- Device Network
- Device Configuration
- Nodeboot
- Package Install
- Device Settings
- Monitoring Overview
- Livestatus

## 6.1.2. Preselection and homebutton

The method by home button is more general. Again preselection of devices or groups takes place in sidebar device tree but afterwards instead of choosing a submenu, this time we click  on the **home** button.

As result we get some useful tabs for each device.


**Figure 6.2. Home button method**




Device configurations for preselected devices as result

Home button method is recommended for working with multiple devices.

Following areas can be accessed by the home button method:


- General
- Category
- Location
- Network
- Config
- Disk
- Vars
- Status History (Detailed information about this tab, see in the chapter  **Status History**)
- Livestatus
- MonConfig/hint
- Graph

By direct click  on the device name in **sidebar device tree** we get similar overview tabs like by home button method but only for one single device.

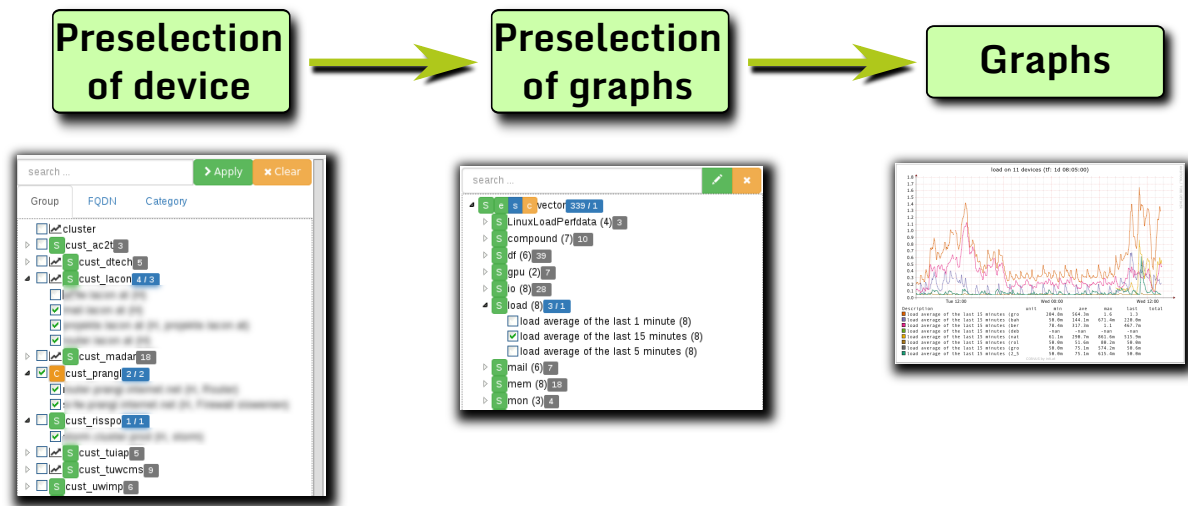
Direct method is recommended for working with single devices.



### 6.1.3. Two-stage selection

For graphing there is the requirement first to select devices, then select the wanted graphs and finally draw this graphs for selected devices. The graph preselection will remain unaffected if you change the device preselection. After changing device preselection you must push  the **Apply** button. Same is true for drawing graphs.

**Figure 6.3. Two-stage selection**



Preselected devices and preselected graphs results in graphs

## 6.2. Input field filter/regex, hide/show column, pagination and sorting

We implemented three little helping functions to ease the handling with large and complex tables inside of the web front-end. This helping functions are located on several places/pages e.g. like **sidebar device tree**, **device tree** or **device configurations** mostly on top of the specific place/page.

### 6.2.1. Input field filter/regex

Most common auxiliary function is the filter input field on top of device trees or configuration tables. Most simple usage of this filter field is to insert some text string or number to filter for. If you do so for example in **sidebar device tree** then only matching devices will be selected.

**Table 6.1. Examples for filter function in sidebar device tree**

Regular expression match character	Matching description	Example	Result
^	Starting position of line	^node	Select all devices whose device name <b>begin</b> with the string <b>node</b> .
[0-9]	Range of numbers	node[0-5]	Select all devices which called node immediately followed by a <b>number between 0 and 5</b>

Regular expression match character	Matching description	Example	Result
\$	End of line	[0-9]\$	Select all devices whose name <b>end</b> with a <b>numeric between 0 and 9</b> .
\d	Digit	\d\$	Select all devices whose name <b>end</b> with a <b>digit</b> .

Regular expressions for input fields.

Further information on regular expressions filter can be found in the world wide web by looking for **javascript regex**.

**Figure 6.4. Input field filter for device configurations**



Displays all device configuration entries beginning with **base**, even they are not selected.

## 6.2.2. Hide/show columns

An other auxiliary function in handling with tables are the **show** and **hide** buttons on top of tables. With this buttons you can easily show or hide specific table columns.

**Figure 6.5. Show and hide buttons for table columns**

Additional columns: TLN RPD store Password MonMaster BootMaster

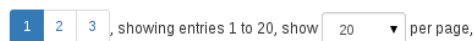
Name	Sel	Description	Enabled	Type	TLN	RPD store	MonMaster	BootMaster	Action
system		system group			[TLN]				<a href="#">modify</a>
georg_testmonitoring	<a href="#">S</a> <a href="#">T</a> <a href="#">C</a>	new devicegroup	yes	5	[TLN]				<a href="#">modify</a> <a href="#">create device</a>
	<a href="#">sel</a>	new device	yes	Host	[TLN]	yes	2_5_branch (master)	N/A	<a href="#">modify</a> <a href="#">delete</a>
	<a href="#">sel</a>	new device	yes	Host	[TLN]	yes	2_5_branch (master)	N/A	<a href="#">modify</a> <a href="#">delete</a>
	<a href="#">sel</a>	new device	yes	Host	[TLN]	yes	N/A	N/A	<a href="#">modify</a> <a href="#">delete</a>

Device tree with hidden **Password** column.

## 6.2.3. pagination

To ease displaying of longer lists and to avoid to much page scrolling there is also a simple pagination function built into the software. With pagination we are able to limit entry output on pages to a specific number. Only the choosen numer of etries will be displayed, the other entries, if there are some, will be divided on separate pages which can be accessed via the page button.


**Figure 6.6. Pages and entries per page**



Shows 20 entries per page, divided into three pages

## 6.2.4. Sort column

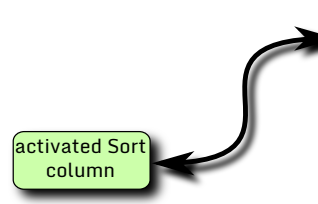
Last but not least we'd like to mention the column sort function. It also could be very useful to display only desired data.

Not all columns provide this sort function, but most of them do. The function will be toggled by clicking  onto the column name. If the function is activated there is a small triangle left beside the column name pointing with its tip either in top direction for **ascending** or in bottom direction for **descending** sorting. If no triangle is visible sorting function is deactivated.

Sorting method is:

1. First numerical
2. Second alphabetical

**Figure 6.7. Sort column**



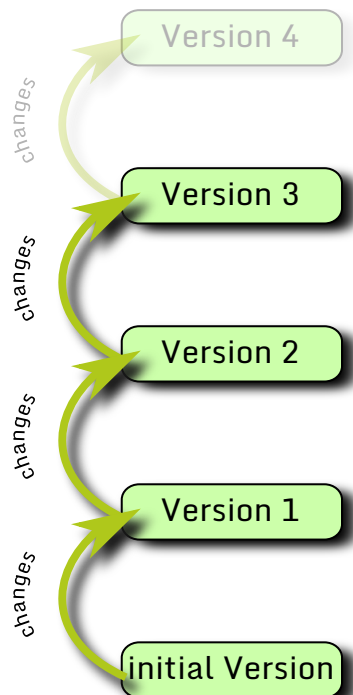
▲ Name	Sel	Description	Enabled	Type	Action
a_node_1	<a href="#">sel</a>	new device	yes	Host	<a href="#">modify</a> <a href="#">delete</a>
a_node_1_special	<a href="#">sel</a>	new device	yes	Host	<a href="#">modify</a> <a href="#">delete</a>
a_node_2	<a href="#">sel</a>	new device	yes	Host	<a href="#">modify</a> <a href="#">delete</a>

Activated ascending sorting marked with a small black triangle pointing in top direction.

## 6.3. Reversion

Sometimes it could be very necessary to undo former applied changes, for example if you have a typo in an script, variable or wherever or if someone else has applied some changes and you want to see the state before and after this changes.


**Figure 6.8. Reversion principle**



Reversion chain

Newest data and changes will be attached on top.

### 6.3.1. History overview

Our developer created a reversion function not only to display what changes were done but also to go back in change history to desired state and drop changes which were done afterwards. History reversion can be found in top menu at **Cluster**  **History**.

The reversion function is based upon the central database (default is PostgreSQL) so in principle every change written into the database can be reversed. Normally there are lot of different data stored in the database to ensure every component works fine but it makes no sense to provide the reversion feature to all of this collected data. For normal user and administrators it is completely enough to revert changes which were done via the web front-end.

For example three new user were added in **User Management**. Like shown below, system history lists all relevant database entries for each of them.

**Figure 6.9. System history**

System history

Data: User

User

showing entries 1 to 5, show 10 per page,

Object											
▼Date	User	Type	Object	ID	Change						
14.04.2015 16:45:03	admin	created	Michael ([4])	4	active	aliases	allowed_device_groups	comment	create_rms_user	date	db_is_auth_for_password email
					true		no entries		false	2015-04-14T14:45:03.218Z	true
14.04.2015 16:44:28	admin	created	Lucy ([3])	3	active	aliases	allowed_device_groups	comment	create_rms_user	date	db_is_auth_for_password email
					true		no entries		false	2015-04-14T14:44:28.239Z	true
14.04.2015 16:44:09	admin	created	Adam ([2])	2	active	aliases	allowed_device_groups	comment	create_rms_user	date	db_is_auth_for_password email
					true		no entries		false	2015-04-14T14:44:08.910Z	true

Database excerpt for **User** data

Now let's suppose there was a typo in one of these names e.g. **Lucy** was changed to **Luci**. If we take a look into the system history under **User** we get exactly this change displayed in a diff-like style.


**Figure 6.10. System history**

▼Date	User	Type	Object	Object ID	Change
15.04.2015 08:53:55	admin	modified	Luci ([3])	3	Changed login: Lucy <i>i</i>

Displayed diff in system history after changed user name

## 6.3.2. Revert to former version

If you are not only satisfied to display changes but also really want to go back to an earlier version there is the **revert to this version** button.

For example, someone changed directory paths in a script located at **Base**  **Configurations** and you would like to display that changes, simply navigate to the **History** tab of **Modification** window to get a list containing all applied changes up to now.


Next step to do is to mark your desired version out of the list. Now, you can either apply reversion by clicking  on the **Modify** button or just switch to the script editor to check how the script looks like after reverting.

Figure 6.11. Reversion overview

▼Date	User	Type	Object	Object ID	Change
14.04.2015 16:36:31	admin	modified	config_script object	4	<div>↻ revert to this version</div> <div>Changed value: # config script (2015-04-14T16:33:58+02:00) # do_nets(config) <del>do_fstab(config)</del> <del>do_routes(config)</del> <del>do_uuid(config)</del></div>
14.04.2015 16:34:59	admin	modified	config_script object	4	<div>↻ revert to this version</div> <div>Changed value: # config script (2015-04-14T16:33:58+02:00) # do_nets(config) do_fstab(config) do_routes(config) <u>do_uuid(config)</u></div>
14.04.2015 16:34:50	admin	modified	config_script object	4	<div>↻ revert to this version</div> <div>Changed value: # config script (2015-04-14T16:33:58+02:00) # do_nets(config) do_fstab(config) <u>do_routes(config)</u></div>

Typical colored diff output

Table 6.2. Colorcode for reversion

green	inserted character
red	deleted character
black	unchanged character

We can see all changes at a glance in above shown figure.

---

# Chapter 7. User and group management

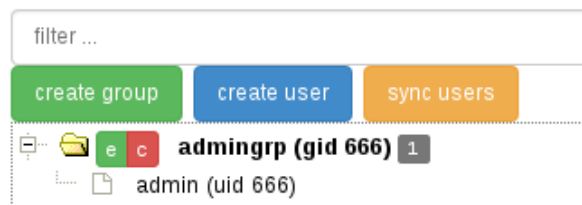
## 7.1. Create user or group

After installation of NOCTUA® the user **admin** and the group **admingrp** already exists. This is the user you have to change password for after first login into your fresh installed system.

User admin has all possible rights and permissions to add, to modify and to delete devices/groups etc. User admin is also able to do reconfiguration of database and of course able to add or delete new user.

If you want to set restrictions for some user or groups, for example for external staff, you have to create this new restricted user/group with following buttons:

**Figure 7.1. Userbuttons**



Create user, group or sync users Button

## 7.2. Create group form

To add a new group in user management, klick the "*create group*" button, fill out the form and confirm your input by klicking the "*Create*" button.

**Figure 7.2. Group create form**

Details for group 'my\_group'

**Basic data**

Groupname

Gid\*

Homestart\*

☒ Active

**Additional data**

Email

Mobile

Tel

Comment

**Permissions**

Allowed device groups

Create

Basic settings form to create group

The form is self-explanatory, but some input should be mentioned anyway:

Gid\* Internal group ID

Device group permissions Set basic permissions to get access to selected devicegroup

Another extended form can be shown by clicking the new created group in the user/group tree:

A more complex permission system appears.

**Figure 7.3. Extended permissions**

Permissions

Parent group

Allowed device groups

Permission

Permission level

Type	Name	Code	Level	Object	Application	Model	Action
------	------	------	-------	--------	-------------	-------	--------

Modify close delete

More complex permission settings

## 7.3. Create user form

Similar structure and procedure is true for creating new user.



Also here we must mention some contents:

Uid*	Internal user ID
Parent group	Is the superior group
Secondary group	Operating system group
Is superuser	Owns all rights and permissions like the admin own


## 7.4. Permission System

### 7.4.1. Permission


The permission system is divided into several parts which covers certain functions. Some permissions depend on other permissions, or in other words, chainpermissions. The more permissions user get the more powerfull they can act. The user "admin" or "superuser" is the most powerfull user. Admin have all possible rights and permissions.

Below is a list with permissions and what their functions are.


#### background\_job





Show background jobs (G)	Shows additional menu button:
	<b>Session</b>  <b>Background Job Info</b>

#### config





modify global configurations (G)	Shows additional menu button:
	<b>Base</b>  <b>Configurations</b>

#### device

Access to device graphs (G/O)	Shows <b>graphs</b> tab for selected devices. Depends on possibility to choose devices ( <b>access all devices</b> )
change disk setup (G/O)	Shows disk tab for selected devices. Depends on possibility to choose devices ( <b>access all devices</b> )
Change basic settings (G/O)	Change basic settings (General) for selected devices. Depends on possibility to choose devices ( <b>access all devices</b> )
Change boot settings (G/O)	Shows new top-menu named <b>Cluster</b>
Change configuration (G/O)	Show <b>Config</b> tab for selected devices. Depends on possibility to choose devices ( <b>access all devices</b> )
Change device category (G/O)	Show <b>Category</b> tab for selected devices. Depends on possibility to choose devices ( <b>access all devices</b> )
Change device connection (G/O)	Shows new top-menu:
	<b>Base</b>  <b>Device connections</b>
Change device location (G/O)	Show <b>Location</b> tab for selected devices. Depends on possibility to choose devices ( <b>access all devices</b> )

Change device monitoring config (G/O)	Shows 3 new tabs for selected devices: <ul style="list-style-type: none"> <li>• Livestatus</li> <li>• Monconfig</li> <li>• MonHint</li> </ul>
Change network (G/O)	Shows new top menu content: <p><b>Base</b>  <b>device network</b> Depends on possibility to choose devices (access all devices)</p>
Change variables (G/O)	Show <b>vars</b> tab for selected devices and new top menu: <p><b>Base</b>  <b>Device variables</b>. Depends on possibility to choose devices (access all devices)</p>
access all devices (G)	The main permission to show devices. Most of above permissions depends on it. Shows existing devices in device tree on the left.
<b>group</b>	
Group administrator (G/O)	...
<b>image</b>	
Modify images (G)	...
<b>kernel</b>	
Modify kernels (G)	...
<b>mon_check_command</b>	
Change monitor settings (G)	Shows new top menu content under: <p><b>Monitoring</b>  <b>Basic Setup / Build Info</b></p>
<b>network</b>	
modify global network settings (G)	...
show network clustering (G)	...
<b>package</b>	
access package install site (G)	Shows new top menu under: <p><b>Cluster</b>  <b>Package install</b>. Additional software packages can be choosen and installed by this menu button.</p>
<b>partition_fs</b>	
modify partitions (G)	...

**user**

Administrator (G/O)	Shows new top menu content unter:  <b>Session</b>  <b>Admin</b>
Change RMS settings (G/O)	...
Modify category tree (G)	Shows new top menu content unter:  <b>Base</b>  <b>Category tree</b>
modify device tree (G)	Shows 2 new top menu content unter:  <b>Base</b>  <b>Crerate new device / Device tree /</b>
modify domain name tree (G)	Shows new top menu content unter  <b>Base</b>  <b>Domain name tree</b>
start and stop server processes (G/O)	...

**7.4.2. Permission level**

The permission level defines what can be done by users. In combination with the permission itself, administrators are more flexible in assigning rights and permissions to user or to groups.

Below are 4 main permission levels which can be assigned.

<b>Read-only</b>	Permits the user to read data. User can't change, create or delete data.
<b>Modify</b>	Permits the user to change existing data. Includes read-only level.
<b>Modify, Create</b>	Permits user to change and create new data. Deletion is not possible.
<b>Modify, Create, Delete</b>	All Permissions are granted.



---

# Chapter 8. Device Management

In this chapter we would like to explain how to add devices, also called hosts, into the system. This works in the same way in the context of monitoring as well as in the context of HPC management.

## 8.1. Device tree

We already know the first [🔗5.2.2: “Sidebar device tree (2)”] placed in the sidebar on the left side of the web front-end. This device tree is mainly used for preselection and direct access to single device info area.

The second and much more powerful device tree can be found at **Base** 🗑️🗑️🗑️🗑️ **Device tree**. It provides an extended filter function, possibility to modify and delete multiple devices at once and of course a function to add new devices and devicegroups. Among this there is also a function to enable or disable single devices or whole device groups.

This is the main tool to managing devices.

### 8.1.1. Filter function

Our filter function allows you to find and select devices even if your choices are very picky.

**Figure 8.1. Device tree filter**

The screenshot shows a web interface for filtering devices. It includes several dropdown menus: 'Group/Device:' with 'all' selected, 'Selected:' with 'selected' selected, 'Enabled:' with 'all' selected, 'MonServer:' with 'ignore' selected, and 'BootServer:' with 'ignore' selected. Below these is a 'Filter:' input field containing the text 'filter'. To the right of the input field are two buttons: 'select shown' (highlighted in blue) and 'deselect all'.

#### Possible filters

<b>Groups/Device</b>	<b>all</b> , displays both devices and groups
	<b>Devices</b> , display only devices
	<b>Group</b> , display only groups
<b>Selected</b>	<b>all</b> , displays both selected and unselected groups or devices
	<b>selected</b> , displays only selected groups or devices
	<b>unselected</b> , displays only unselected groups or devices
<b>Enabled</b>	<b>all</b> , displays both enabled and disabled groups or devices
	<b>enabled</b> , displays only enabled groups or devices
	<b>disabled</b> , displays only disabled groups or devices
<b>MonServer</b>	<b>ignore</b> , ignore filter option
	SERVERNAME, displays only groups or devices linked to this Monitor Server
<b>BootServer</b>	<b>ignore</b> , ignore filter option
	SERVERNAME, displays only groups or devices linked to this Monitor Server

**Filter**

This input field allows you to filter devices and groups by name and special regular expressions described in [Section 6.2.1](#), “Input field filter/regex”

⚠ There are two selection buttons at the end of the filter input field: **select shown** and **deselect all**. This selection buttons are only related to this device tree and have no influence on the sidebar preselection.

## 8.1.2. Device tree overview

On top of device tree overview, right below the filter options, there are four buttons. The green button function on the left side are creation of new devices or new groups. The blue and red button on the right side handle deletion and modification of multiple groups or devices at once.

Below our creation, modification and deletion buttons there is a pagination row and some column filter buttons. Next there is a table displaying overall information for groups (highlighted with a light green color) and devices:

Name	Device or groupname
Sel	Selection button
Description	Group or device escription text
Enabled	Displays if device or group is enabled (yes) or disabled (---)
Type	Device or grouptype
TLN	Top level node
RRD store	Shows if sensor data is stored on disk
Password	Shows if password is set or not
MonMaster	Shows monitoring master server for this device
Bootmaster	Shows the boot master server if it exist
Action	<b>modify</b> , modification of single device or group <b>delete</b> , deletion of single device or group <b>create device</b> , creation of single device

**Figure 8.2. Device tree overview**

<div> <span>+ create device</span> <span>+ create devicegroup</span> <span>↗ modify selected</span> <span>🗑 delete selected</span> </div>										
showing entries 1 to 10, show <input type="text" value="20"/> per page,										
Additional columns: <span>TLN</span> <span>RRD store</span> <span>Password</span> <span>MonMaster</span> <span>BootMaster</span> <span>cURL</span>										
Name	Sel	Description	Enabled	Type	TLN	RRD store	Password	MonMaster	Action	
TLN	S T C	TLN	yes	9	[TLN]				↗ modify	+ create device
TLN_group	sel	TLN_group	yes	Host	TLN_group	yes	not set	N/A	↗ modify	🗑 delete
TLN_group	sel	TLN_group	yes	Host	TLN_group	yes	not set	N/A	↗ modify	🗑 delete
TLN_group	sel	TLN_group	yes	Host	TLN_group	yes	not set	N/A	↗ modify	🗑 delete
TLN_group	sel	TLN_group	yes	Host	TLN_group	yes	not set	N/A	↗ modify	🗑 delete
TLN_group	sel	TLN_group	yes	Host	TLN_group	yes	not set	N/A	↗ modify	🗑 delete
TLN_group	sel	TLN_group	yes	Host	TLN_group	yes	not set	N/A	↗ modify	🗑 delete
TLN_group	sel	TLN_group	yes	Host	TLN_group	yes	not set	N/A	↗ modify	🗑 delete
TLN_group	sel	TLN_group	yes	Host	TLN_group	yes	not set	N/A	↗ modify	🗑 delete
TLN_group	sel	TLN_group	yes	Host	TLN_group	yes	not set	N/A	↗ modify	🗑 delete


### 8.1.3. Create device, create devicegroup, modify selected and delete selected

With help of these four top buttons and their equivalent on the right side of the device table overview you are able to comfortable managing devices and groups.

#### Create device, create devicegroup

Use this buttons to create new devices or new groups.



**Table 8.1. Settings for device**

<b>Basic settings</b>	<b>Name</b>	Name of device
	<b>Comment</b>	Device comment field to set additional information
	<b>Device type</b>	Dropdown with existing device types
	<b>Device group</b>	Dropdown with existing device groups
<b>Domain tree node</b>	Dropdown for domain tree node selection	
	<b>Bootserver</b>	Dropdown for boot server selection
	<b>Monitor sServer</b>	Dropdown for monitor server selection
<b>Security</b>	<b>root passwd</b>	Input Field for root password (for node setup  )
<b>Flags</b>	<b>Enabled</b>	Enable or disable device. (If device is disabled it will not be displayed in sidebar device tree)
	<b>Enable perfdata, check IPMI and SNMP</b>	Enable or disable performance data, IPMI and SNMP checks
	<b>Store rrd data</b>	Enable or disable storage of rrd data of this device to disk
	<b>Flap detection enabled</b>	Enable or disable flap detection for this device

**Table 8.2. Settings for devicegroup**

<b>Basic settings</b>	<b>Name</b>	Name of devicegroup
	<b>Description</b>	Devicegroup description field
<b>Additional settings</b>	<b>Domain tree node</b>	Dropdown for domain tree node selection
<b>Flags</b>	<b>Enabled</b>	Enable or disable whole group

#### Modify

There are two different ways to modify devices in **device tree**. Either you can modify a single device by pushing the **modify** button beside it [see  Figure 8.2, “Device tree overview”] or you can modify several devices at once by selecting all desired devices and leftclick  on **modify selected** on top.

Multiple modification of devices provides the possibility to change almost all settings for devices like single modification provides. Only the name and the comment field of devices can not be changed in multiple modify mode because it does not make any sense.

Typical usage of multiple modification could be changing device groups , changing the monitoring server or just enable or disable a couple of devices at once.

## Delete

Due to the fact that all configuration data is stored in tables of our central database and that data objects of one single device or group can be referenced to one or more other objects we have to be very careful deleting objects.

A safe delete function is implemented to prevent database damage by deleting database objects which reference to other objects. This safe delete function works for two scenarios and provides additional options.

First scenario deals with objects without any **hard reference**. Lean back and enjoy deleting such devices or groups because it's completely secure. This is the usual case.

Second scenario deals with for objects with **hard references**. In this case you are also able to delete, but you have to specify actions to handle with references.

**Figure 8.3. Safe delete**

Delete

Delete settings

☒ Delete asynchronously

Deletable without deleting hard references

☒ ☒ ☒

delete

Deletable with deleting hard references

The following devices can only be deleted if references of them are also deleted, which can cause delete cascades. Please select actions for each of these devices.

Monitoring hint

Device

show (39)

0

delete referenced object

force delete

## Delete dialog

### Delete settings

If the checkbox is disabled deletion will be done in foreground blocking the web front-end until delete process is finished.

**Delete asynchronously**, if the checkbox is enabled deletion will be done in background and web front-end is unblocked. This option is recommended for deletion of many devices.



**Deletable without deleting hard references**

Deletion of devices listed here is safe, i.e. deleting these objects does not affect any other objects in the database. Using the checkboxes, you can restrict the devices you actually would like to delete.

**Deletable with deleting hard references**

Each device with hard references will be displayed in a separate tab.

**Table**, shows the table name in which the reference occurs

**Field**, shows the table field name which references the object we want to delete

**First level references and Second level references**, displays the number of references. An additional **show/hide** button opens a detailed content list of the table.

⚠ In order to prevent unwanted delete cascades, user interaction is needed to select one of the following actions in the **Action** dropdown menu:

**set reference to null**, replace reference by null, causing no deletions of referenced objects.


**delete referenced objects**, delete referenced objects (if only first level reference)

**delete cascade on referenced objects**, delete referenced objects recursively, i.e. deleted all objects referenced by this objects and all objects referenced by any reference.


⚠ Be aware, deleting cascades may cause a faulty installation. Only delete cascades if you know what you are doing.

## 8.1.4. Fast device creation



Sometimes administrators would like to add devices without the need of complex configuration. For this case there is a simple and fast device adding dialog, focused only on essential settings listed below.

The dialog for fast device device creation can be found under **Base**  **Create new device**.

**Figure 8.4. Create new device**


Fully qualified device name	<input style="width: 90%;" type="text" value="device name"/> <div style="font-size: 0.8em; color: #dc3545; margin-top: 2px;">Need name without any special characters or whitespace (for instance 'server.my.domain')</div>
Devicegroup	<input style="width: 90%;" type="text" value="newgroup"/>
IP Address	<input style="width: 90%;" type="text" value="IP Address"/> <div style="font-size: 0.8em; color: #dc3545; margin-top: 2px;">Please enter a valid IP address</div>
Icon	 <input style="width: 80%;" type="text" value="linux40"/>
Monitor host via	<div style="display: inline-block; border: 1px solid #ccc; padding: 2px 5px; margin-right: 5px; background-color: #28a745; color: white;">IP</div> <div style="border: 1px solid #ccc; padding: 2px 5px; background-color: #d3d3d3;">Name</div>
is routing capable (switch, firewall)	<div style="display: inline-block; border: 1px solid #ccc; padding: 2px 5px; margin-right: 5px; background-color: #d3d3d3;">yes</div> <div style="border: 1px solid #ccc; padding: 2px 5px; background-color: #28a745; color: white;">no</div>
Connect to	<div style="border: 1px solid #ccc; padding: 2px; display: flex; justify-content: space-between; align-items: center;"> <span>no peering</span> <span>▼</span> </div>
Comment	<input style="width: 90%;" type="text" value="new device"/>


## Input fields for easy and fast device addition

Fully qualified device name	Fully qualified domain name. For example <b>host.domain.com</b>
Devicegroup	Pick an existing group or create a new one
IP Address	Field for IP address, if an  <i>FQDN</i> exists and the nameserver works, IP address will be inserted automatically. <b>Resolve</b> button appears beside the input field.
icon	List of existing icons. Selected icon will be displayed inside of icinga.
Monitor host via	
is routing capable (switch, fire-wall)	Device is really routing capable, that means it is able to forward IPs
Connected to	Connection to network topology central node, also called <b>peer connection</b>
Comment	Field for additional information

---

# Chapter 9. Network Management

There are two basic network settings inside of NOCTUA®. The first one is more general and relates to the monitoring system itself. It is placed in **Base**  **Networks** and is subdivided into three areas.

The second one is more specific and is direct related to devices. It resides in **Base**  **Device networks** and is also subdivided into three areas.

## 9.1. Networks



### 9.1.1. Network device types



### 9.1.2. Network types



### 9.1.3. Networks



## 9.2. Device Network

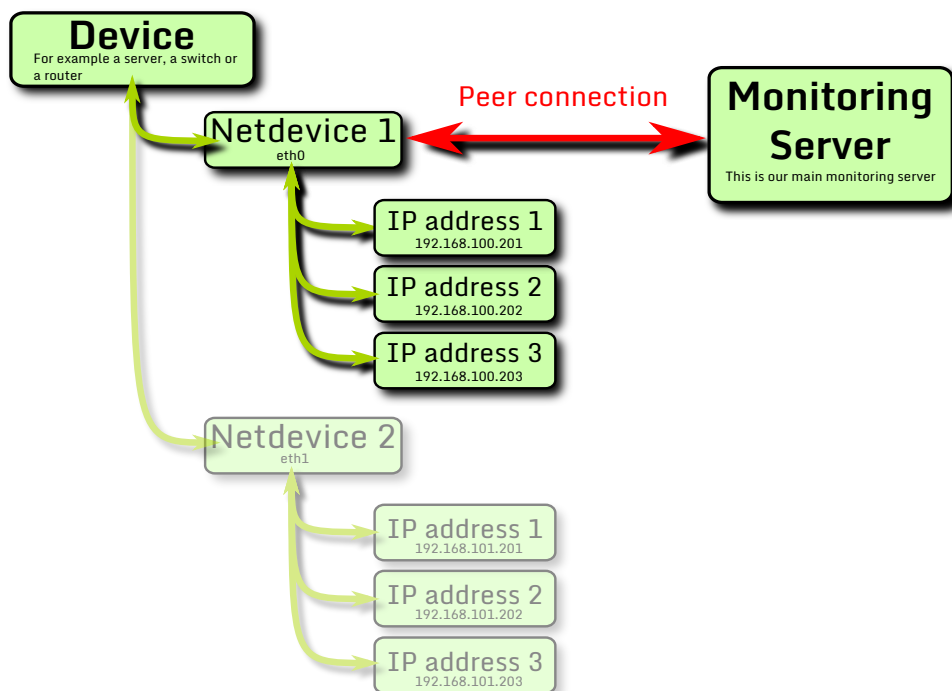
Like mentioned above the second network settings are related to devices.

### 9.2.1. Settings

This is the main place for device network settings and the place where **devices**, **netdevices**, **IP address** and **peers (network topology central node)** will be assigned.

Basically inside of **Device networks** our structure is also a tree. For each device one or more netdevice can be assigned. Same is true for IP address. For each netdevice one or more IP address can be assigned. Only exception here is the **peer connection**. Following picture illustrates the tree structure inside of **device network**.

Figure 9.1. Typical tree structure



Single device with one or more netdevice again assigned with one or more IPs

## Devices

Devices, also often called hosts, are real hardware components or VM devices integrated into the software either through [ 8: “Device Management”] or through [ 8.4: “Create new device”].

One device is absolutely unique bounded into the system with an unique name.

Figure 9.2. List of devices

Device	ScanInfo	BootInfo	#Ports	#IPs	#peers	SNMP schemes	action
...	...	N/A	5	4	2	<a href="#">generic.netip</a> <a href="#">generic.net</a> <a href="#">generic.base</a>	<a href="#">Create new</a> <a href="#">update network</a>
...	...	N/A	1	1	1		<a href="#">Create new</a> <a href="#">update network</a>
...	...	N/A	2	2	1		<a href="#">Create new</a> <a href="#">update network</a>

<b>Device</b>	Name of the device like listed in [Chapter 8, <i>Device Management</i> ]
<b>ScanInfo</b>	
<b>BootInfo</b>	
<b>#Ports</b>	Number of assigned netdevices
<b>#IPs</b>	Number of assigned IPs
<b>#Peers</b>	Number of assigned peers
<b>SNMP schemes</b>	Name of used snmp schemes (detected and used by [Chapter 13, <i>SNMP discovery</i> ])
<b>action</b>	<b>Infobutton</b> for extended information





**Create new** button to create new netdevices, IPs or network topology connections (peers).

**update network** button to use the discovery server with SNMP or host-monitor for automatically scan and integration of found netdevices. Take a look into [Chapter 13, *SNMP discovery*] for more information.

### Create new netdevice

After usage of **Create new** netdevice button, an modal window appears.








**Table 9.1. Basic settings**

<b>Basic settings</b>	<b>Devname*</b>	Name of netdevice
	<b>Description</b>	Netdevice description text
	<b>Mtu*</b>	Maximum Transmission Unit
	<b>Netdevice speed*</b>	Speed settings for netdevice, with or without check
	<b>Enabled</b>	Enable or disable netdevice.
	<b>Is bridge</b>	Defines netdevice as bridge. Disabled flag results in additional vlan settings.
<b>Routing settings</b>	<b>Cost</b>	Prioritize network connections. The higher the value the less priority a network connection has.
	(important for proper monitoring) <b>Network topology central node (peer)</b>	Define netdevice as  <b>network topology central node (peer)</b>
	<b>Inter device routing</b>	
<b>VLAN settings (only available if "Is bridge" flag is disabled)</b>	<b>Master device</b>	
	<b>Vlan id</b>	

Fields marked with \* are required

⚠Speed settings are required only for proper monitoring

**Table 9.2. Hardware**

<b>Basic settings</b>	<b>Driver</b>	
	<b>Driver options</b>	
<b>ethtool settings (for cluster boot)</b>	<b>Ethtool autoneg*</b>	Dropdown for on, off and default. 
	<b>Ethtool duplex*</b>	Dropdown for on, off and default. 
	<b>Ethtool speed*</b>	Dropdown for speed definition. 
<b>MAC address settings</b>	<b>Macaddr</b>	Inputfield for MAC address
	<b>Fake macaddr</b>	Inputfield for fake MAC address 
	<b>Force write DHCP address</b>	

Fields marked with \* are required

## Netdevices

One or more netdevice can be linked to each device. As usual for normal computer there are one loopback device and one ethernet device. But in real networks there can be much more netdevices. Server, switches or router for example could have assigned a large number of netdevices.

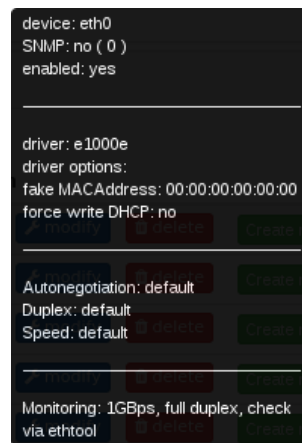
On top of netdevice overview there are show and hide buttons for some columns described below. Use this buttons to show or hide columns which are important or not for you.

**Figure 9.3. List of netdevices of one device**


Show/Hide columns: [idx](#) [port](#) [ips](#) [peers](#) [bridge](#) [mac](#) [devtype](#) [mtu](#) [speed](#) [penalty](#) [flags](#) [status](#)

Device	idx	Port	#IPs	#peers	Bridge	MAC	Devtype	MTU	speed	penalty	flags	status	action
	enp3s0	1	0		slave (transbridge)	54:04:a6:4c:b2:ae	en (Ethernet new scheme [6])	1500	1GBps, full duplex, check via ethtool	1	introuting		<a href="#">modify</a> <a href="#">delete</a> <a href="#">Create new</a>
	lo	1	0			00:00:00:00:00:00	lo (loopback devices [6])	1500	1GBps, full duplex, check via ethtool	1	introuting		<a href="#">modify</a> <a href="#">delete</a> <a href="#">Create new</a>
	nodebridge	0	0		bridge	fe:34:03:ff:ff:12	bridge (generic bridge [6])	1500	1GBps, full duplex, check via ethtool	1	introuting		<a href="#">modify</a> <a href="#">delete</a> <a href="#">Create new</a>
	transbridge	1	0		bridge (enp3s0)	54:04:a6:4c:b2:ae	bridge (generic bridge [6])	1500	1GBps, full duplex, check via ethtool	1	introuting		<a href="#">modify</a> <a href="#">delete</a> <a href="#">Create new</a>


<b>Device</b>	Devicename
<b>idx</b>	Like primary key in database context, idx is an unique id
<b>Port</b>	Netdevice name
<b>#IPs</b>	Number of assigned IP address
<b>#peers</b>	Number of assigned peers
<b>Bridge</b>	Name of bridge device if one exists
<b>MAC</b>	The hardware MAC address.
<b>Devtype</b>	Detected netdevice type
<b>MTU</b>	Maximum Transmission Unit
<b>speed</b>	Netdevice link speed
<b>penalty</b>	Value to prioritise network connections, the higher the value, the less privileged network connection is.
<b>flags</b>	Displays enabled flags for netdevices
<b>status</b>	Shows netdevice status if scanned by snmp. Up means the device is administrative up and ready to work, up/down means the device is administrative up but not operational e.g. because of missing link
<b>action</b>	<p><b>Info</b> button for extended information</p> <p><b>modify</b> button to modify netdevices</p> <p><b>delete</b> button to delete netdevices</p>

**Figure 9.4. Netdevice info**


### Modify netdevice

Modification of existing netdevice is as easy as create new one. Simply push  the **modify** button and the modification window appears.




### Delete netdevice

To delete netdevices from your list click  on the **delete** button.

### Create new IP address

IP address can be assigned to netdevices by pushing  the **Create new** button and selecting the dropdown **IP Address**. Below you can see a description table for risen window.

**Table 9.3. IP Adress**

<b>Basic settings</b>	<b>Netdevice*</b>	IP address will be assigned to this netdevice
	<b>Ip*</b>	Valid IP address
	<b>Network*</b>	Dropdown with available networks
	<b>Domain tree node*</b>	Dropdown with available domain tree nodes 
<b>Alias settings (will be written without node postfixes)</b>	<b>Alias</b>	
	<b>Alias excl</b>	

Fields marked with \* are required

### Create new network topology connection

Like shown in [Figure 9.1, “Typical tree structure ”] **network topology connection** (peer connection) is a very important network setting to get your system and especially your monitoring work.

Please look into [Section 9.2.2, “Network topology ”] for detailed information.

**Table 9.4. Create new network topology connection**

<b>Settings</b>	<b>Cost*</b>	Prioritize network connections. The higher the value the less priority a network connection has.
-----------------	--------------	--

	<b>Source</b>	The source netdevice
	<b>Network topology central node</b>	This is the central node other netdevices can be connected to. Usually this will be a monitoring server or a cluster server.
	<b>Source spec</b>	Additional field for source information or comment
	<b>Dest spec</b>	Additional field for destination information or comment
	<b>Info</b>	General info or comment field

Fields marked with \* are required

## IPs

Third list in device network settings is reserved for IP overview.

**Figure 9.5. IPs overview**

Device	Port	IP	Network	DTN	alias	action
autogen1	enp3s0	169.254.0.0/8	autogen1 (169.254.0.0/B, o)	[TLN]		<a href="#">modify</a> <a href="#">delete</a>
loopback	lo	127.0.0.1	loopback (127.0.0.0/A, o)	[TLN]	localhost	<a href="#">modify</a> <a href="#">delete</a>
all	nodebridge	0.0.0.0	all (0.0.0.0/0.0.0.0, o)	[TLN]		<a href="#">modify</a> <a href="#">delete</a>
lan	transbridge	192.168.1.0/24	lan (192.168.1.0/C, o)	[TLN]		<a href="#">modify</a> <a href="#">delete</a>

Overview of existing IPs

## Peer connections

Last area inside of settings tab lists existing peers also called network topology connections.

**Figure 9.6. Peer overview**

Device	Port	IPs	cost	Dest	type	Autocreated	Info	action
transbridge (1)		192.168.1.0/24	with cost 1 <a href="#">3</a>	to eth0 (1) on 2_5_branch	---	no		<a href="#">modify</a> <a href="#">delete</a>
enp3s0 (1)			with cost 1 <a href="#">3</a>	to eth0 (1) on 2_5_branch	---	no		<a href="#">modify</a> <a href="#">delete</a>
eth0 (1)		192.168.1.0/24	with cost 1 <a href="#">3</a>	to eth0 (1) on 2_5_branch	---	no		<a href="#">modify</a> <a href="#">delete</a>
enp2s0 (1)		192.168.1.0/24	with cost 1 <a href="#">3</a>	to eth0 (1) on 2_5_branch	---	no		<a href="#">modify</a> <a href="#">delete</a>
eth0 (1)		192.168.1.0/24	with cost 1 <a href="#">3</a>	to eth0 (1) on 2_5_branch	---	no		<a href="#">modify</a> <a href="#">delete</a>

Overview of existing peers


## 9.2.2. Network topology

What we called peering is the possibility to link single hosts and devices in the network to central nodes. It is not only the possibility but also a condition to connect and link devices with a monitoring server. As a result you get a structural network map called **network topology**.



## Network topology

Select some devices in the sidebar, navigate to **Base**  **Device network**, and then into the **network topology** tab.

**Control:** Selected devices should be displayed by default. Use  left mouse button to move topology view, use your mouse scroll wheel to zoom in or out. Pinning of devices is also possible with drag and drop. Pinned devices will be marked as *red*.

Following settings are available:

### Show network topology for

With this dropdown, you are able to display more or less of your network facility. Select + 1 (next ring) displays your selected devices, plus all devices which are one level above the selected ones. Analogue it's true for +2, +3.


### Redraw

Draws topology view again.

### show/hide livestatus

Toggle between livestatus view and pure view

### Filter options

Displays only selected segments of livestatus. For details please look into section  livestatus filter options

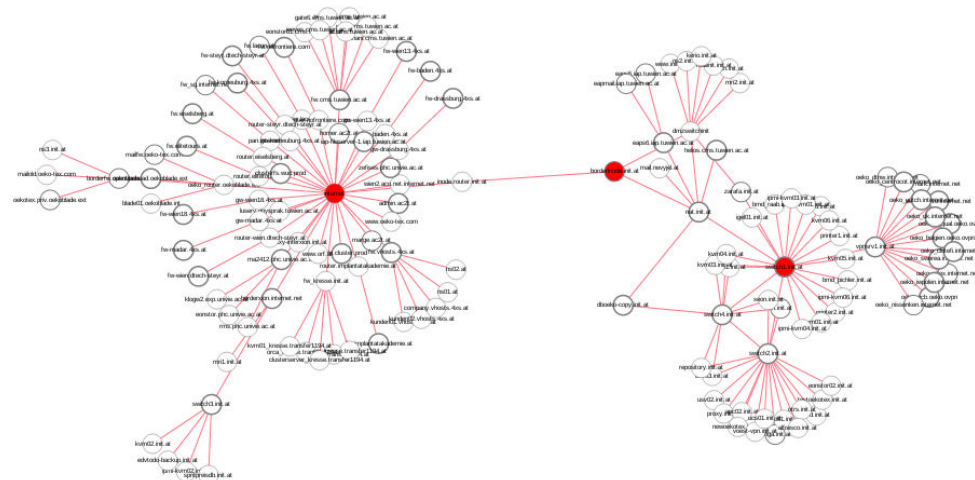
**Figure 9.7. Topology view settings**

Show network topology for  mode:   livestatus, Filter options: 

## Two Network topology views

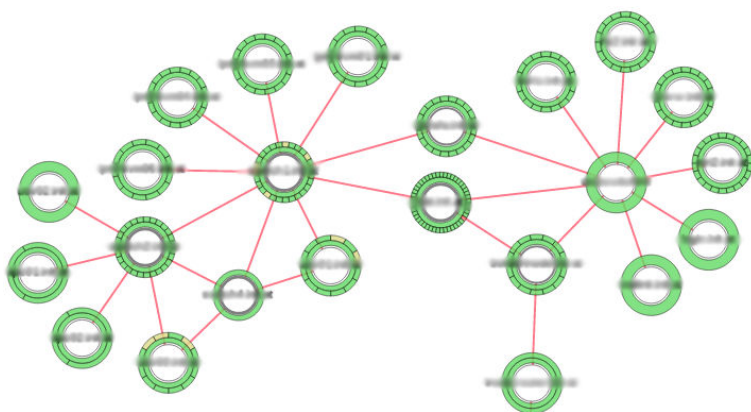
Pure view displays only hosts without any additional information. Because of less data transfer it is, especially for wide networks, faster than a view with livestatus.

**Figure 9.8. Pure network topology**



In contrast to the pure view, livestatus view provides much more useful information. It overlays the real livestatus of every device into network topology view.

Figure 9.9. Network topology with integrated livestatus



### 9.2.3. clusters



Figure 9.10. Network cluster overview

Network clusters (1), [reload](#)

Index	has netdevices	devices	netdevices	selected	action
1	yes	9	16	yes (8)	<a href="#">show</a>

# Chapter 10. Monitoring

The primary purpose of **NOCTUA®** is to monitor network devices and devicegroups. Nearly each measurable value like space, speed, temperature, rpm, availability and much more can be monitored, observed, recorded and evaluated.

Even applications and their stdout or stderr data can be integrated into the monitoring workflow by self designed commands

There are almost no limits about which device can be monitored. Typical devices are: **Fileserver, Cluster, Web-server, Switches, Printer, Router, Thin clients and even Telephone systems**

## 10.1. Three different ways of monitoring

### 10.1.1. Basic Monitoring

No client software needed

As "Basic Monitoring" we call monitoring which do not require any additional software on the client side. This kind of monitoring can be done out of the box. Only thing administrators have to know is the network IP-address of the client machine which should be monitored.

**Typical checks for basic monitoring are:**

- http (check port 80)
- https (check port 443)
- ldap (check port )
- ldaps (check port 636)
- ping
- ssh (check port 22)

All these checks can also be performed via command line of a linux box.

**Figure 10.1. Basic monitoring**



### 10.1.2. Advanced Monitoring SNMP/IPMI

Need a running SNMP daemon on the client side or a IPMI interface

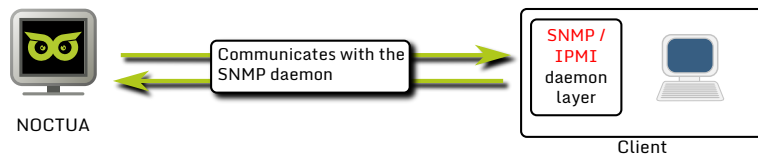
A bunch of checks are available over the SNMP protocol. In contrast to above mentioned passive monitoring checks, this kind of checks need a running SNMP daemon to work with. Many hardware out there like server, switches or router already provides SNMP information by default. For other machines don't providing SNMP by default, it is mostly possible to install a SNMP daemon afterwards.

Use following commands to install the snmp-client package on the machine:

**apt-get install snmpd**

**zypper install snmpd**

The other way to get active monitoring data and therefore check results is the IPMI interface integrated in devices. **NOCTUA**® is also able to evaluate this kind of data.

**Figure 10.2. SNMP monitoring****10.1.3. Advanced Monitoring with host-monitoring**

Need a running host-monitoring process on the client side (part of the **icsw-client** package)


Last but not least way to monitor devices is our self developed host-monitoring service, which is one part of the **icsw-client** package. It provides not only common checks like ping or http-check but also special system monitoring checks like mailq, quota, free and much more.

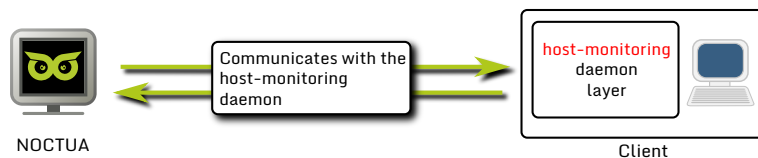
If you like to monitor system values like that you have to install the **icsw-client** package on each client. For that use the following commands:

**apt-get install icsw-client**

**zypper install icsw-client**

Then start the host-monitoring service on the client side

To be able to install the package you must already set the right repositories on your operating system. Please look into the  **Installation** Chapter for details.

**Figure 10.3. host-monitoring monitoring****10.1.4. Setup host-monitoring and SNMP daemon**

After successful installation of host-monitoring package, you have to edit its config file: `/etc/sysconfig/host-monitoring.d/machvector.xml`


File content of `machvector.xml` should looks like this:

```
<mv_target
target="192.168.1.232"
send_every="30"
enabled="1"
port="8002"
send_name=" "
full_info_every="10"
immediate="1"
send_id="1"
sent="0" />
```

Of course there can be more than one target line, each with unique `send_id`.

Relevant parts are the **target** and the **enabled** parameter. The other parameters are just for fine tuning. Finally you have to restart the host-monitoring service with **rhost-monitoring restart**.

### Parameter description

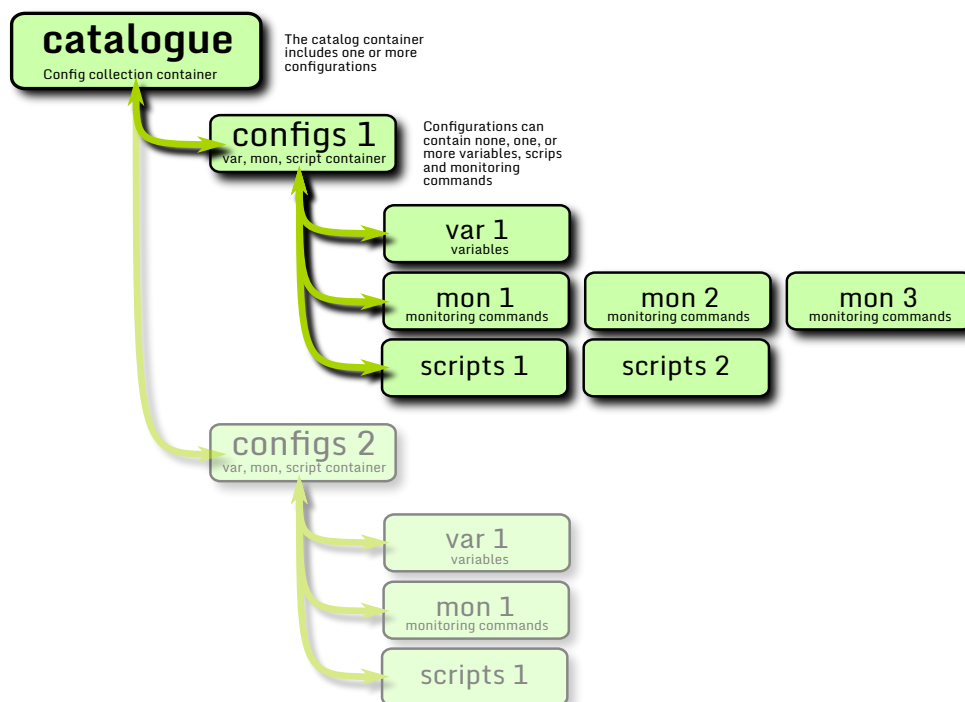
<b>target</b>	IP address of the monitoring server  Possible value is any valid IP address
<b>send_every</b>	Period of sent data
<b>enabled</b>	Enable or disable sending function
<b>Port</b>	Monitoring server port on which the server waits for data stream
<b>send_name</b>	<div style="border: 1px solid #92d050; padding: 5px;"><b>Δ</b>Hostname of the monitored device. This must be the same name as the devicename given in <b>NOCTUA®</b>, otherwise collected data can not be assigned to the right device.</div>
<b>full_info_every</b>	Timeperiod of sent full information data
<b>immediate</b>	Option to send data immediately or cache data and send later
<b>send_id</b>	ID number of sent data
<b>sent</b>	

## 10.2. Configuration hierarchy

Monitoring configuration is build as a hirarchial tree structure. Root of every configurations is the **catalog** . The catalog contains **configurations** , which contains again either **variables** , **monitoring checks** or **scripts** , all together or only one of them.

Catalogs can contain various numbers of configurations.

Figure 10.4. Tree structure configuration



Variables let you override **NOCTUA®** specific settings or pass information into **NOCTUA®**. Monitoring Configs are used to describe which check should be performed against the devices that are associated with the config. The most powerful part of the Configuration system are the Scripts. These allow you to execute arbitrary Python code to generate files and directories on the fly. There are several utility functions already accessible.

```

do_fstab()
do_etc_hosts()
do_nets()
do_routes()
do_uuid()

```

The Python dictionary `conf_dict` is available as well. It contains configuration information like node ip and other.


## Tip

To include an already existing file in the node config use `show_config_script.py` to render the content as Python code ready for inclusion.

```
show_config_script.py [FILENAME]
```

## 10.2.1. Catalog

The highest layer in configuration hierarchy is the catalog.

By default there is only one catalog containing some basic configurations. Take a look into **Base**  **Configurations** and choose the **catalog** tab. You will see a table with at least one entry and a **modify** button. The column **#configs** shows you how many configurations are stored in this catalog.

**Figure 10.5. Catalog table**

configurations

catalogs

1 catalogs,

create new catalog

name	URL	Author	Version	#configs		action
local	http://www.initat.org/	Andreas Lang-Newjel	1	24	<div>modify</div>	

If you ever plan to collect your own configurations in one central place this is your first class choice. There will be a download function for catalogs in future versions. The **create new catalog** button shows a window with a few basic settings:

<b>Name</b>	The catalog name
<b>Author</b>	The name of the author
<b>URL</b>	\$\$Place where catalog will be saved

### Note

⚠ You can only delete a catalog if it doesn't contain configs any more.

## 10.2.2. Configurations

The second layer below the catalog layer is the configurations layer. All check commands, variables and scripts are stored into it. Typically a configuration consists of check commands, scripts and variables.

The configurations tab resides right beside the catalog tab, it displays an overview table with some configuration entries, filter tools and action buttons.

Figure 10.6. Configurations overview

23 configurations, shown: 23, [create new config](#)

filter  name script var mon 1 2, showing entries 1 to 15, show 15 per page

Name	Pri	Flags	Catalog	enabled	Description	parent	var	script	mon	cats	action	refs
auto_etc_hosts	0	S Y	local	yes	/etc/hosts file can be created from local cluster-server	-	0	0	0	-	modify Create	refs: 1
base_without_swap	1000	S Y	local	yes	description for base	-	2	1	10	-	modify Create delete	
check_http	0	S Y	local	yes		-	0	0	1	-	modify Create	refs: 1
check_https	0	S Y	local	yes		-	0	0	1	-	modify Create	refs: 1
check_imap	0	S Y	local	yes		-	0	0	1	-	modify Create	refs: 1
check_imaps	0	S Y	local	yes		-	0	0	1	-	modify Create	refs: 1
check_ldap	0	S Y	local	yes		-	0	0	1	-	modify Create	refs: 1
check_ping	0	S Y	local	yes		-	0	0	1	-	modify Create	refs: 1
check_pop3s	0	S Y	local	yes		-	0	0	1	-	modify Create	refs: 1
check_smtps	0	S Y	local	yes		-	0	0	1	-	modify Create	refs: 1
check_snmp_info	0	S Y	local	yes		-	0	0	1	-	modify Create	refs: 1
check_ssh	0	S Y	local	yes		-	0	0	1	-	modify Create	refs: 1
discovery_server	0	S Y	local	yes	enables network discovery and inventory	-	2	0	0	-	modify Create delete	
monitor_server	0	S Y	local	yes	sets device as the monitor master server	-	39	0	0	-	modify Create delete	
monitor_slave	0	S Y	local	yes	sets device as a monitor slave (satellite)	-	0	0	0	-	modify Create delete	

[download selected configs](#)  No file chosen

[Down and uploadbutton](#)

## Configuration table

<b>Filter</b>	Allows filtering for configuration name, variables, scripts or check commands.
<b>Create new config</b>	Creates a new empty configuration entry
<b>Pagination</b>	Turns on other pages or define how many entries will be displayed
<b>Action buttons</b>	Buttons to create, modify or delete variables, scripts or check commands
<b>Down and uploadbutton</b>	Either download selected configurations or upload own configuration files (in *.xml format)

Buttons like this one: 2 shows you the amount of entries and allows you to open the specific section for more detailed view.

### 10.2.3. Configurations

To bind checks or configurations to devices and groups we use **Base** **Device configurations**. We can select single configurations for devices or for whole device groups. Displayed icons for enabled configurations differ for devices and groups.

Configurations which can not be assigned to groups due to **server** and **system** flags are marked as cross on red background. Assigned configurations are displayed as check mark on green background for devices and as circle with check mark inside of it for groups.



Figure 10.7. Device configurations

Type	local_mot	S/Y	auto_etc_hosts (0, 0, 0)	check_idap (0, 0, 1)	config_server (6, 0, 0)	discovery_serv	imac
[Group]	1	*	✓	*	*	*	*
node	1 1	-	⊙	-	-	-	-
node	2 1	✓	⊙	-	-	-	-
node	2 1	✓	⊙	-	-	-	-
server_group [Group]	0	*	-	*	*	*	*
cluster_dev	9 0	-	✓	✓	✓	✓	✓

Annotations:

- Server and system flag: points to the **local\_mot** column header.
- Group configs enabled: points to the **check\_idap (0, 0, 1)** column header.
- Device configs enabled: points to the **cluster\_dev** row.
- Not available configs: points to the **server\_group [Group]** row.

## 10.2.4. Server configurations

Following list describes only server and system configurations:

### Server and system configurations


<b>auto_etc_hosts</b>	If enabled on server, the command <b>cluster-server.py -c write_etc_hosts</b> is able to write into <b>/etc/hosts</b> .
<b>config_server</b>	...
<b>discovery_server</b>	Enables the server to act as discovery_server (required for SNMP scan or disk scan)
<b>image_server</b>	Enables the server to act as image server, providing images to nodes.
<b>kernel_server</b>	Enables the server to act as kernel server, providing kernels to nodes
<b>logcheck_server</b>	...
<b>monitor_server</b>	Enables the server to act as monitoring master server
<b>monitor_slave</b>	Enables the server to act as monitoring worker server (For distributed monitoring)
<b>mother_server</b>	...
<b>package_server</b>	Enables the server to act as package server (required for package install)
<b>quota_scan</b>	...

<b>rms_server</b>	Enables the server to act as rms server (required for RMS)
<b>rrd_collector</b>	...
<b>rrd_server</b>	...
<b>server</b>	...
<b>syslog_server</b>	...
<b>user_scan</b>	...
<b>usv_server</b>	...
<b>virtual_desktop</b>	Enables the virtual desktop feature on server side
<b>virtual_desktop_client</b>	Enables the virtual desktop feature on client side

## 10.3. Basic monitoring for devices

To begin slowly, first lets do a basic example configuration. In this basic example we want to check a simple ping response for a host in a local network. With this monitoring information we can make assumption about the networkdevice itself or its sourrounding network area.


1. Create a new device (connected to your monitoring server) and configure at least one **network device** for it, one **IP address** and one **peer/network topology connection** .
2. Select the new device from device tree and navigate to the **config** tab.

Alternative you can preselect one device in the sidebar device tree and go to **Base**  **Device configurations**.

3. Enable the **check\_ping** config to activate the check.

**Figure 10.8. Selected ping check**



	auto_etc_hosts	check_http	check_https	check_imap	check_imaps	check_ldap	check_ping	check_pop:
	{0, 0, 0}	{0, 0, 1}	{0, 0, 1}	{0, 0, 1}	{0, 0, 1}	{0, 0, 1}	{0, 0, 1}	{0, 0, 1}
Type	local	meta	S/N					
	1	0	-	-	-	-	✓	-

To make sure your configuration will be applied you must rebuild your config database. Go to top menu, click **Monitoring**  **rebuild config (cached, RC)**

## 10.4. Basic monitoring for devicegroups

Now, that we know how to create simple checks for single devices, lets do a more complex configuration with more than one device and more than one check.

For this plan we have to use devicegroups with defined check configs:

1. In top menu navigate to **Base**  **Device tree**
2. Create a new devicegroup by pushing the **create devicegroup** button.
3. Create some new devices by navigating to **Base**  **Create new device** and entering some domains into the *Fully qualified device name* field. The IP address should be automatic resolved, if not, try to push the **Resolve** button.

Choose your monitoring server as "**Connect to**" device.

4. Select the checkbox of the new devicegroup from devicetree menu on the left side and push the **home** button or the green arrow **use selection** button.
5. In Config tab click the minus sign below check\_ping topic to activate ping checks for the whole group. Now every device assigned to the **http** group has the check\_ping check activated.

**Figure 10.9. Selected ping check**

			auto_etc_hosts (0, 0, 0)	check_http (0, 0, 1)	check_https (0, 0, 1)	check_imap (0, 0, 1)	check_imaps (0, 0, 1)	check_ldap (0, 0, 1)	check_ping (0, 0, 1)	che (0, 0, 1)
Type	local	meta	S/Y							
http [Group]	1			-	-	-	-	-		

Configurations with red background and a visible cross inside a circle are locked and can not be selectet. Mostly this affects configs which don't make any sense to be selected.

## 10.5. Livestatus

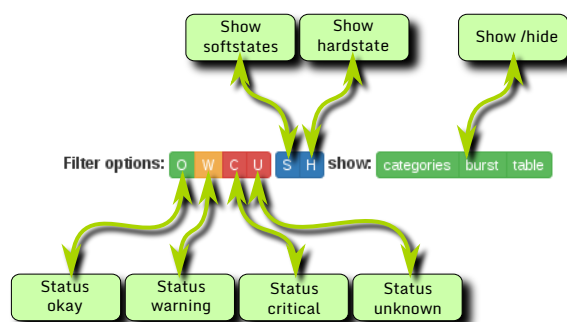
One of the most helpful feature in **NOCTUA®** is the livestatus. The livestatus view shows you current status of **devicegroup**, **devices**, **services** as different output formats.

The classic view of livestatus is the graphical burst. The burst consists of multi layer circles and its circle segments. Each segmet stands for a specific group, device or service and displays the current state of it almost in realtime. The segments color can be either green (okay), yellow (warning) or red (critical).

### 10.5.1. Livestatus filter options

Like on other places within **NOCTUA®**, also in livestatus there are handy filter options. You can show or hide the different check states, and livestatus views.

**Figure 10.10. Livestatus filter options**



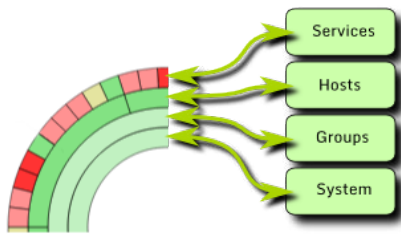
Accessible filter options

The filteroption **map** only appears if location or image maps are defined otherwise it is hidden.

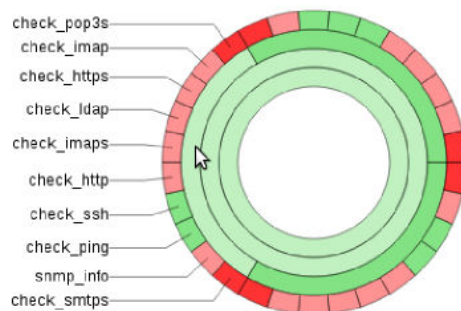
### 10.5.2. Livestatus burst

The layer of livestatus view follows this confentions:

The outer the layer the more exact is the indicator. The innermost circle layer respresents the whole system with all connected and configured checks.

**Figure 10.11. Livestatus burst layer description**

Four visible layer, from system to service

**Figure 10.12. Displayed services**

Mouseover function shows details of top layer

Mouseover function provides in addition an information table beside of the burst view. For each segment of the burst there will be faded according information table.

**Table 10.1. Description for mouseover information table**

<b>Device</b>	Devicename
<b>Description</b>	Description text for device
<b>Output</b>	Output of command or check
<b>State</b>	State of check
<b>Style type</b>	Either hard or soft state type
<b>Check type</b>	Type of check, either active or passive
<b>attempts</b>	Number of checking attempts, i.e. 1 of 5

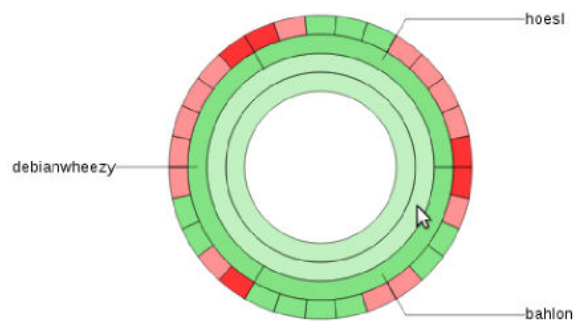
Dependent of the chosen segment there will be more or less details in the table.

**Figure 10.13. Mouseover information table**

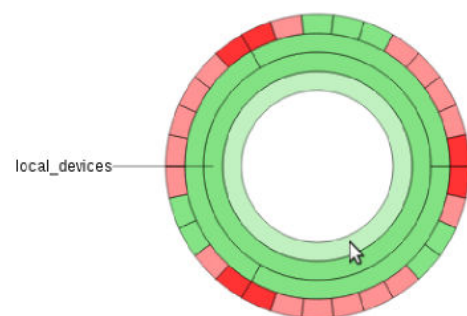
Level: service

Device	hoesl
Description	check_http
Output	HTTP OK: HTTP/1.1 301 Moved Permanently - 458 bytes in 0.163 second response time
State	OK
State type	hard
Check type	active
attempts	1


Information table displayed while mouseover on burst segment

**Figure 10.14. Displayed devices/hosts**

Mouseover function shows details of second layer

**Figure 10.15. Displayed devicegroup**

Mouseover function shows details of third layer

With left mouse click  into a device/host segment, only selected device and its services will be displayed on the whole livestatus burst, now other devices should be hidden. To go back to livestatus overview of all selected devices click with the left mousebutton into one of the inner burst layer.

### 10.5.3. Livestatus tables

If you don't like the graphical burst view there is also a table or list view integrated into NOCTUA®. To hide the burst view use the show/hide buttons within the filter options. With an active **table** filter button, you now should see only the table view.

**Figure 10.16. Livestatus table view**

Show table columns: node name state description categories state type last check last change result

showing entries 1 to 9, show 20 per page, filter...

node name	state	description	categories	state type	last check	last change	result
bahlon	OK	check_ping	services	hard	4 minutes	3 hours	192.168.1.209: 5 of 5 (0.14 ms mean time)
bahlon	OK	check_ssh	services	hard	3 minutes	3 hours	SSH OK - OpenSSH_6.2 (protocol 2.0)
bahlon	Critical	check_http	services/web	hard	2 minutes	3 hours	connect to address 192.168.1.209 and port 80: Connection refused
debianwheezy	OK	check_ping	services	hard	a minute	4 hours	192.168.1.235: 5 of 5 (0.36 ms mean time)
debianwheezy	OK	check_ssh	services	hard	5 minutes	4 hours	SSH OK - OpenSSH_6.0p1 Debian-4+deb7u2 (protocol 2.0)
debianwheezy	Critical	check_http	services/web	hard	4 minutes	3 hours	connect to address 192.168.1.235 and port 80: Connection refused
hoesl	OK	check_ping	services	hard	3 minutes	4 hours	192.168.1.193: 5 of 5 (0.34 ms mean time)
hoesl	OK	check_ssh	services	hard	2 minutes	4 hours	SSH OK - OpenSSH_6.6.1 (protocol 2.0)
hoesl	OK	check_http	services/web	hard	a few seconds	3 hours	HTTP OK: HTTP/1.1 301 Moved Permanently · 458 bytes in 0.162 second response time

On top of the table view there are some hide/unhide buttons for every column of the table. Use this buttons to show or hide columns you would like to be shown or not.

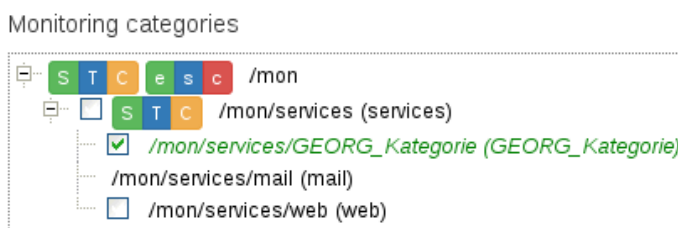
There is also a simple pagination function to limit the table length and a filter input field to sort according to strings in the node name, description and result column. And of course the main filter function options mentioned in [Livestatus filter](#) can be applied too.

### 10.5.4. Livestatus monitoring categories

Other than livestatus burst and table view, livestatus categories serve an other powerful and handy arrange function. Administrators can group configurations or check commands into self created categories in the **category tree**.

This way it is very simple to group thematic related configurations or check commands. Grouped configurations/check commands can be easily displayed or hidden with the livestatus monitoring category tree.


**Figure 10.17. Livestatus monitoring categories**



Only one selected category and its related checks will be displayed in livestatus burst and tables view.

Please look into [Configurations categories](#) section to find out how to set categories.

## 10.6. Monitoring overview

The livestatus burst is also part of the monitoring overview placed in menu **Monitoring**  **Monitoring Overview**. Monitoring overview is the best choice if you want to get a summary of your host status and service status.

There is a filter input field and an **only selected** button on top of the monitoring overview. Below that filter there is a simple pagination function to limit displayed device number on the page. The main area contains availability pie charts for hosts and services and additionally the livestatus view. It shows only the status for the last week, for yesterday and for now.

Pay attention of the different color meanings for hosts and services, especially the orange and red color.

**Table 10.2. Colorcode for service status**

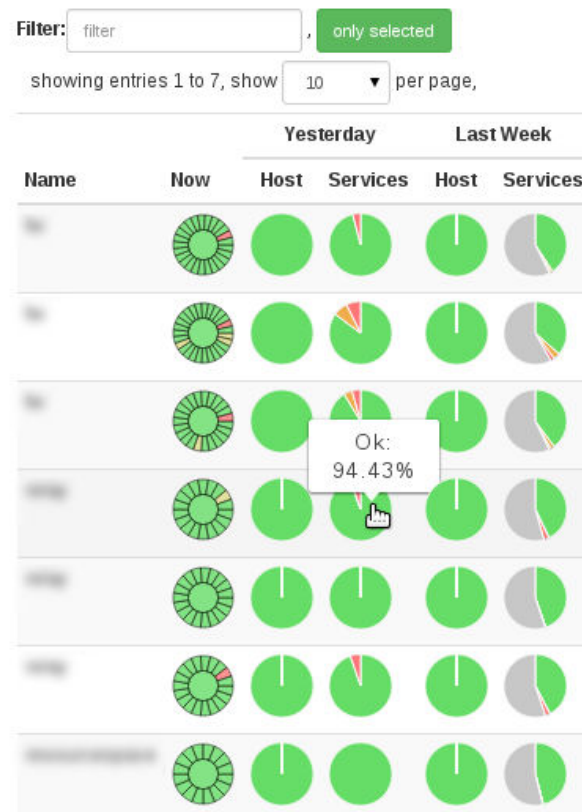
green	Ok
orange	Warning
red	Critical
gray	Undetermined

**Table 10.3. Colorcode for hosts status**

green	Ok
orange	Unreachable
red	Down
gray	Undetermined

With mouseover function you are able to see related values of the pie chart.

**Figure 10.18. Monitoring overview**



Monitoring overview for seven devices with displayed mouseover values for one of it

Monitoring overview do not work if **NOCTUA®** is running with the SQLite database.

## 10.7. Status History

In contrast to the `livestatus` function, status history provides output data for hosts and checks for past timeperiods. This can be very useful to generate availability reports and check the status of hosts and services to a specific timeperiod.

"Status History" tab shows the device status information, which can be divided into three parts:

- The summary about reachability of device for the specific period
- The summary about checks status for the predetermined timerange
- Extensive information about events and delivered messages of each check for the selected time cycle

For determination of the timerange at first select the time unit. There are 4 different values to define the time unit: day, week, month and year.

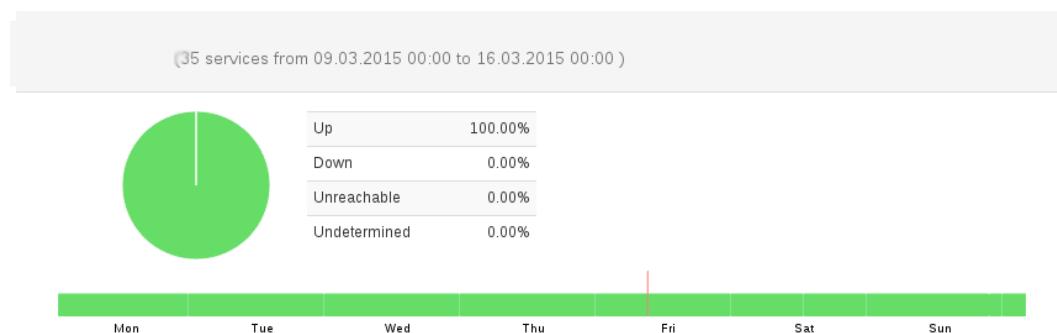
If the time unit set, in the field on right site you can select the specific period: day - if the unit set to day or week, month - for monthly unit, year - for unit "year".

If the timerange defined correctly, above the timeline appears the messages "Showing data from ... to ...".

And below the timeline shows up the refreshed status burst of the device reachability for the specific period.

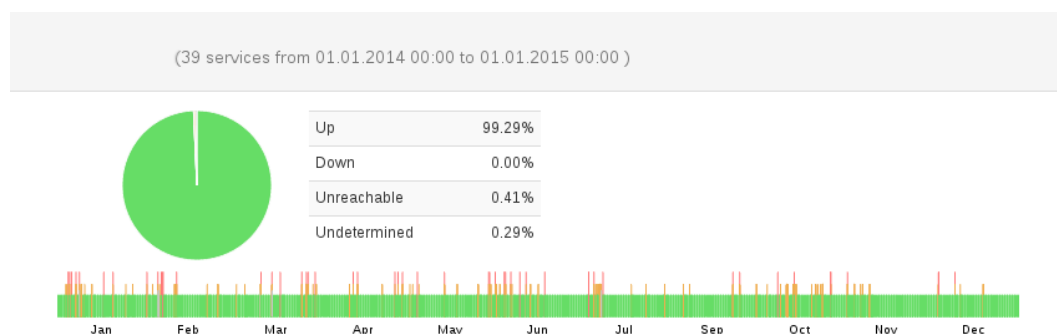
For example, the weekly overview can look so:

**Figure 10.19. Weekly host status history**



or yearly overview:

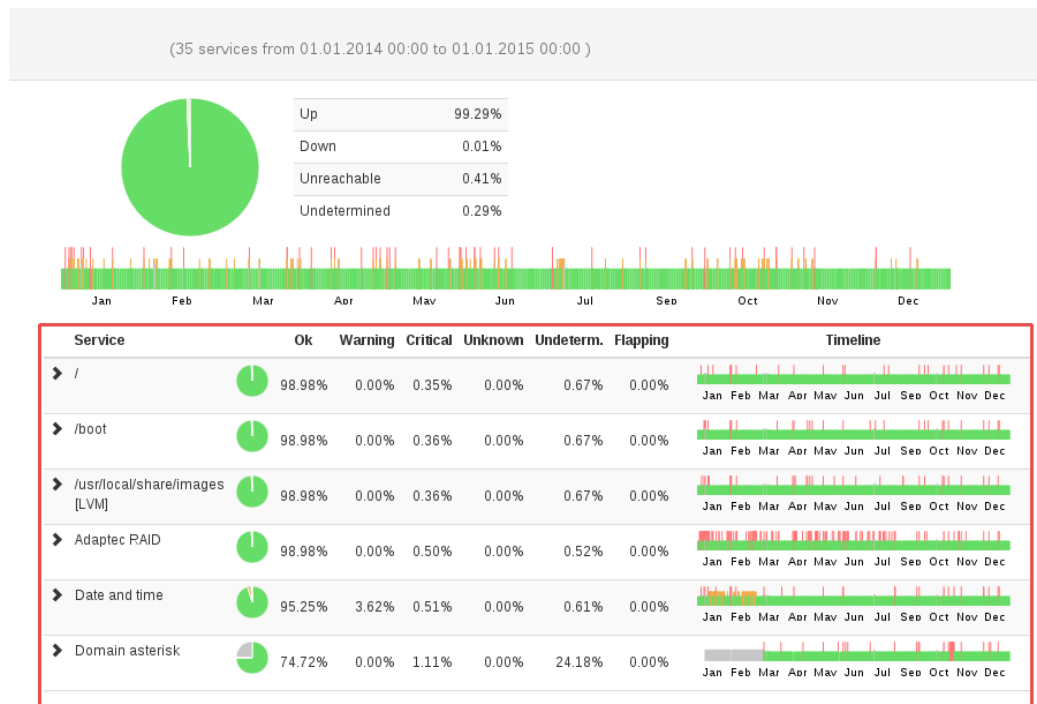
**Figure 10.20. Yearly host status history**



Below the status burst there is the check status panel. The panel shows percentage of the events with status "ok", "Warning", "Critical", "Unknown", "Undetermined", "Flapping".

Graphical representation of the checks status for specific timeline is presented with the last column in this table.

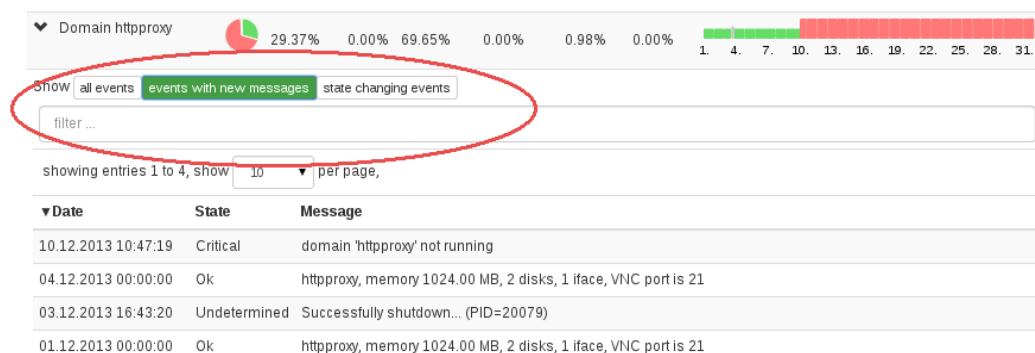


**Figure 10.21. Status history of checks**

To get the detailed information about the events, click the "arrow" symbol on the left site for desired check. For selection criteria there are 3 options:

- "all events" - shows all events for selected check during predefined time period
- "events with new messages" - shows the events with different messages (works similar to "group by" option)
- "state changing events" - shows only the events, which inform about the state change for the check during selected timeline

Select the appropriate option to get the requested information. Additionally, there is the "Filter" - field for precise selection.

**Figure 10.22. Status history of checks**

## 10.8. Notifications

The **NOCTUA®** notification system is made up of seven parts. Each of them is responsible for a specific task and is located in a separate tab.

- Periods
- Notifications
- Contacts
- Service templates
- Device templates
- Host check commands
- Contactgroups

### 10.8.1. Periods

First tab in notification view is the period tab. By default there is only one period defined, its name is **always**, it has no alias and can't be deleted. This period is true for every day and every hour, that means it is always true.

Periods can be helpful to set exact notification times for example only on weekends, hollidays, day and night stages or working hours. Or it is also possible to notify different people to different times for example during day or night shift operation at which different administrators are responsible for monitoring.

⚠ Periods are used not only for notifications but also for monitoring purpose itself.

**Figure 10.23. Periods overview**

Periods (3 entries), [create new](#)

showing entries 1 to 3, show 10 per page,

▲ Name	Alias	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Use count	Action
always		00:00-24:00	00:00-24:00	00:00-24:00	00:00-24:00	00:00-24:00	00:00-24:00	00:00-24:00	1	<a href="#">modify</a>
Wednesday	wednesday	00:00-00:00	00:00-00:00	00:00-24:00	00:00-00:00	00:00-00:00	00:00-00:00	00:00-00:00	0	<a href="#">modify</a> <a href="#">delete</a>
weekend	Sa und So	00:00-00:00	00:00-00:00	00:00-00:00	00:00-00:00	00:00-00:00	00:00-24:00	00:00-24:00	0	<a href="#">modify</a> <a href="#">delete</a>

Period overview with one default period and two user defined periods

### 10.8.2. Notification templates

Notifications are basically message templates.


By default there are four monitoring notifications templates, SMS and MAIL notifications for each of services and hosts.

New templates can be created by the green **create new** button on top. In risen window most of the form fields are self-explanatory like Name, Channel (Either E-Mail or SMS), Notification type. To get exact information about the hostname, hostaddress, hostoutput, hoststate, date and time in recived notification, you are able to insert a couple of variables into the forms. This variables will be replaced by related values of related hosts. The table below explains all possible variables.

## Configuration table

<b>Name</b>	Notification name
<b>Channel</b>	E-Mail or SMS
<b>Notification type</b>	Host or service
<b>Subject</b>	Mail subject
<b>Content</b>	Message Content. Following variables can be used to generate message content: \$HOSTSTATE\$, \$NOTIFICATIONTYPE\$, \$INIT_CLUSTER_NAME\$, \$HOSTNAME\$, \$HOSTSTATE\$, \$HOSTADDRESS\$, \$HOSTOUTPUT\$, \$LONGDATETIME\$

**Table 10.4. Description of available variables**

Variable	Description
\$HOSTSTATE\$	State of Host, can be UP, DOWN or UNREACHABLE
\$HOSTNAME\$	Name of host that notifies
\$INIT_MONITOR_INFO\$	Icinga version that notifies
\$NOTIFICATIONTYPE\$	Either PROBLEM, 
\$INIT_CLUSTER_NAME\$	Name of cluster that notifies
\$HOSTADDRESS\$	Host IP address that notifies
\$HOSTOUTPUT\$	Information about what goes wrong
\$LONGDATETIME\$	Date and time of notification

**Figure 10.24. Default notification templates**

Notifications (4 entries), [create new](#)

showing entries 1 to 4, show  per page,

▲ Name	Channel	Type	Subject	Content	Enabled	Action
host-notify-by-mail	mail	host	Host \$HOSTSTATE\$ alert for \$HOS...	***** \$INIT_MONITOR_INFO\$ ***** Notification Type...	yes	<a href="#">modify</a> <a href="#">delete</a>
host-notify-by-sms	sms	host		\$HOSTSTATE\$ alert for \$HOSTNAME\$ (\$HOSTADDRESS\$)	yes	<a href="#">modify</a> <a href="#">delete</a>
service-notify-by-mail	mail	service	\$NOTIFICATIONTYPE\$ alert - \$HOS...	***** \$INIT_MONITOR_INFO\$ ***** Notification Type...	yes	<a href="#">modify</a> <a href="#">delete</a>
service-notify-by-sms	sms	service		\$NOTIFICATIONTYPE\$ alert - \$SERVICEDESC\$ is \$SERVIC...	yes	<a href="#">modify</a> <a href="#">delete</a>

Notificatio template overview

**Figure 10.25. Host-notify-by-mail template**

Monitoring Notification

Base data

Name\*

Channel\*

Notification type\*

Flags and text

☒ Enabled

Subject

Content\*

```
***** $INIT_MONITOR_INFO$ *****

Notification Type: $NOTIFICATIONTYPE$

Cluster: $INIT_CLUSTER_NAME$
Host : $HOSTNAME$
State : $HOSTSTATE$
Address: $HOSTADDRESS$
Info : $HOSTOUTPUT$

Date/Time: $LONGDATETIME$
```

[Modify](#)


Default mail notification template with some variables

### 10.8.3. Contacts

The next part of notification setup is the Contact area. Usage of contacts makes it easy to forward specific notifications to specific user. With contacts you are able to distribute notifications to different user.

Sometimes it's not a good idea to send all notifications to one single person. Instead of this you can create a second contact with a second user which should get only notifications of special services or only at specific time. This is why notification in **NOCTUA®** are so powerful. The administrator or user has a very powerful, easy to control and high configurable tool.

Combination of **Periods**, **Notification Templates** and **contacts** gives you a couple of options to setup your notification system.

To add other user to your notification contacts, and in case admin is the only one user, first of all you have to create new users in **Users**  **Overview**.

**Figure 10.26. Contacts overview**

Contacts (1 entries), [+ create new](#)

showing entries 1 to 1, show  per page,

▲User	Service:	rec	crit	warn	unkn	flap	planned	Host:	rec	down	unrch	flap	planned	Notifications	Alias	Action
admin	always	yes	yes	yes	yes	yes	yes	always	...	yes	...	...	...	host-notify-by-mail	...	<a href="#">modify</a> <a href="#">delete</a>

Overview of existing contacts

Use the **create new** button in order to add new notification contacts.

Monitoring contacts window consists of three areas:

**Three monitoring contact sections**

- Base data
- Service settings
- Host settings

**Base data section**


<b>User</b>	User who will get notifications
<b>Notifications</b>	Notification template which will be used
<b>Alias</b>	Alias for notification

**Service settings section**

<b>Service period</b>	Period which will be used
<b>Notify on service recovery</b>	Notifies user if monitored service is in recovery state.
<b>Notify on service critical</b>	Notifies user if monitored service is in critical state.
<b>Notify on service warning</b>	Notifies user if monitored service is in warning state.
<b>Notify on service unknown</b>	Notifies user if monitored service is in unknown state.
<b>Notify on service flapping</b>	Notifies user if monitored service is in flapping state.
<b>Notify on service planed down-time</b>	Notifies user if monitored service is in planed downtime state.

**Host settings section**

<b>Host period</b>	Period which will be used for host
<b>Notify on host recovery</b>	Notifies user if monitored host is in recovery state.
<b>Notify on host down</b>	Notifies user if monitored host is down.
<b>Notify on host unreachable</b>	Notifies user if monitored host is unreachable.
<b>Notify on host flapping</b>	Notifies user if monitored host is in flapping state.
<b>Notify on host planed downtime</b>	Notifies user if monitored host is planed downtime state.

All of the above mentioned states are icinga states. Please consult the icinga manual for further information of icinga states -->  [\[http://docs.icinga.org/latest/en/statetypes.html\]](http://docs.icinga.org/latest/en/statetypes.html)

⚠ There must be at least one contact to get permission to the icinga layer.

**10.8.4. Service templates**

Service templates ease the use of notifications in conjunction with contact groups enormously. Once a service template is created, it can be used as a template inside of contactgroups tab or on other places (e.g. in **configurations**).

Possible settings for service templates are:

**Base data**

**Name** Name of the template

**Volatile** 

**check**

**Nsc period** Notification period

**max attempts** 

**Check interval** Intervall of checks in minutes

**Retry interval** Number of checks until it become from **SOFT** state to **HARD state**

**Notification**

**Nsn period** Notification period 


**Ninterval** Notification interval

**Notify on recovery** Send notification in case of recovery state

**Notify when critical** Send notification in case of critical state

**Notify when warning** Send notification in case of warning state

**Notify when unknown** Send notification in case of unknown state

**Notify when flapping** Send notification in case of flapping state. For details please look into the  **flapping** section.

**Notify when planed downtime** Send notification in case of planed downtime.

**Freshnes settings**

**Check freshness** 

**Freshness threshold** Limit for freshness (only if "Check freshness" is enabled) 

**Flap settings**

**Flap detection enabled** Enables the flap detection mode

**Low flap threshold** If state in percent of the last 20 checks falls below it, flapping stops (only if "Flap detection" is enabled)

**High flap threshold** If state in percent of the last 20 checks exceeds it, flapping starts (only if "Flap detection" is enabled)


**Flap detect ok** Enables flap detection for state **ok** (only if "Flap detection" is enabled)

**Flap detect warn** Enables flap detection for state **warn** (only if "Flap detection" is enabled)

**Flap detect critical** xEnables flap detection for state **critical** (only if "Flap detection" is enabled)

**Flap detect unknown** Enables flap detection for state **unknown** (only if "Flap detection" is enabled)

## 10.8.5. Device templates


Like service templates, device templates ease the use of notifications. 

Possible settings are:


### Base data

<b>Name</b>	Device template name
<b>Mon service template</b>	Use specific mon service template



### check

<b>Host check command</b>	Choose a given check command
<b>Mon period</b>	Time period which will be used
<b>Check interval</b>	Intervall of checks in minutes
<b>Retry interval</b>	Number of checks until it become from <b>SOFT</b> state to <b>HARD state</b>
<b>Max attempts</b>	maximum number of attempts til 

### Notification

<b>Not period</b>	Notification period 
<b>Ninterval</b>	Notification interval
<b>Notify on recovery</b>	Send notification in case of recovery state
<b>Notify when host down</b>	Send notification in case if host is down
<b>Notify when unreachable</b>	Send notification in case if host is unreachable
<b>Notify when host is flapping</b>	Send notification in case if host is flapping
<b>Notify when for planed downtime</b>	Send notification if host is in planed downtime state

### Freshnes settings

<b>Check freshness</b>	
<b>Freshness threshold</b>	Limit for freshness (only if "Check freshness" is enabled) 

### Flap settings

<b>Flap detection enabled</b>	Enables the flap detection mode
<b>Low flap threshold</b>	If state in percent of the last 20 checks falls below it, flapping stops (only if "Flap detection" is enabled)
<b>High flap threshold</b>	If state in percent of the last 20 checks exceeds it, flapping starts (only if "Flap detection" is enabled)
<b>Flap detect up</b>	Enables flap detection for host state <b>up</b> (only if "Flap detection" is enabled)

<b>Flap detect down</b>	Enables flap detection for host state <b>down</b> (only if "Flap detection" is enabled)
<b>Flap detect unreachable</b>	Enables flap detection for host state <b>unreachable</b> (only if "Flap detection" is enabled)

## 10.8.6. Host check commands

This are the commands which check if hosts are in up or in down state. By default four check commands are included.

### check-host-alive

```
$USER2$ -m localhost ping $HOSTADDRESS$ 5 5.0
```

Check the host with **ping** command

### check-host-alive2

```
$USER2$ -m $HOSTADDRESS$ version
```

Check the host with **version** command

### check-host-down

```
$USER1$/check_dummy 2 down
```

Check if the host is down with a **dummy\_check** command

### check-host-ok

```
$USER1$/check_dummy 0 up
```

Check if the host is ok with a **dummy\_check** command

## 10.8.7. Contact groups

Notifications enables you also to send messages to a group of contacts. Every existing contact (⚠ not existing user!) can be marked as a member of a contact group. This way it is possible to notify not only one single person, but several persons at once.

For each member of the group, there are individual settings stored in the contacts tab itself. That means you can create a contact group, consisting of many different contacts which again could consist of many different no

<b>Name</b>	Contact group name
<b>Alias</b>	Alias of contact group
<b>Member</b>	Member of contact group
<b>Device groups</b>	Specify device group to use in contact group
<b>Service template</b>	Specify service templates to use in contact group

## 10.8.8. Flapping

Flapping is the repeatedly change of host or service state during the last 21 checks. It will be calculated with special algorithms. The unit of flapping is percent.



Not every state change has to be notified to an administrator. Often there are regular processes which includes temporary service downtimes, timeouts and similar states. All this can happens in a smaller or bigger period of time. To make it possible to differ between normal flapping and between flapping due to an host or service issue, there are built in threshold values for flapping.


There are two threshold values which allows you to set up when flapping detection should starts and when it should stops. As a rule of thumb you can use following description:

### Rule of thumb to set flapping thresholds

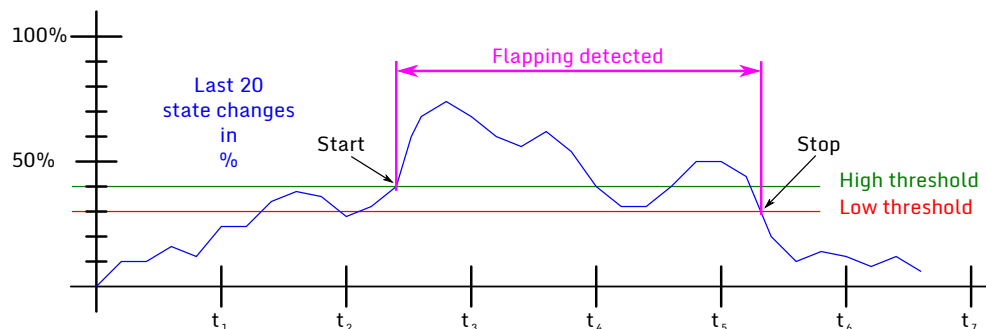
**Low flap threshold** This is the lower value in percent, related to the last 21 checks (20 possible states). If the value goes below it, flapping stops.

**High flap threshold** This is the higher value in percent, related to the last 21 checks (20 possible states). If the measured value exceeds it, flapping starts.

For example: If the last 20 states, changes for 15 times, you get about 75% status change ( $15/20 \cdot 100\%$ ). Of course this is not the absolute truth because of a more realistic calculation algorithm which weights current state changes more than older one. But to get a feeling for how flapping in **NOCTUA®** works it is enough.

To find out how exact flapp detection works, please take a look into the well written icinga manual at  [http://docs.icinga.org/latest/en/flapping.html]

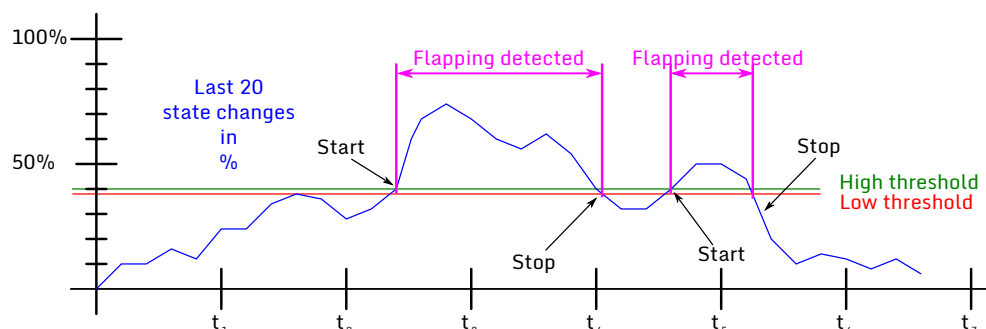
**Figure 10.27. Smooth flapping notification settings**



Bigger difference between high and low threshold to get smoother flapping notifications

In above figure Description of flapping you can clearly see the function of high and low threshold. So to control your flapping notifications you have to choose a proper percent value **and** a proper difference value between the high and low threshold. The smaller the difference of high and low threshold is, the more accurat detected flapping notification is. Or in other words: The bigger the difference between high and low threshold is, the smoother detected flapping notification is.

**Figure 10.28. More accurate flapping notifications**



Smaller difference between high and low threshold to get more accurate flapping notifications

## 10.9. Parameterizing checks

To explain parameterized checks, first of all we have to understand checks itself. Usually a check is a command, created in the monitoring web interface and executed by icinga. Some possible icinga commands are:

```
check_apt
check_breeze
check_by_ssh
check_clamd
check_cluster
check_dhcp
check_dig
check_disk
check_disk_smb
check_dns
check_dummy
check_file_age
check_flexlm
check_ping
...
```

For every single command there are some special options. Below are some options for the `check_ping` command:

Options:

```
-h, --help
    Print detailed help screen
-V, --version
    Print version information
--extra-opts=[section][@file]
    Read options from an ini file. See
    https://www.monitoring-plugins.org/doc/extra-opts.html
    for usage and examples.
-4, --use-ipv4
    Use IPv4 connection
-6, --use-ipv6
    Use IPv6 connection
-H, --hostname=HOST
    host to ping
-w, --warning=THRESHOLD
    warning threshold pair
-c, --critical=THRESHOLD
    critical threshold pair
-p, --packets=INTEGER
    number of ICMP ECHO packets to send (Default: 5)
-L, --link
    show HTML in the plugin output (obsoleted by urlize)
-t, --timeout=INTEGER
    Seconds before connection times out (default: 10)
```

Now that we know what checks really are we can go ahead and explain parameterized checks.


In **NOCTUA®** there are two different methods to create checks (icinga commands).

## 10.9.1. Types of checks

### Fixed

Checks will be defined individual with fixed options and bound on specific devices. These checks are always specific, that means to change one option of the check is the same as to change the whole check.

### Parameterized

Checks are defined globally as  *Parameterized check* and bound on devices. These checks are not specific, that means to change one option of the check it is enough to change the parameter of it.

## 10.9.2. Examples

### Different port

There are 10 devices, 7 of them should be checked on port 80 and 3 of them on port 8080:

#### Solution fixed method:

You have to set up two different checks, one check with option set to port 80 (-p 80) and one check with option set to port 8080 (-p 8080).

#### Solution parameterized method:

You have to set up only one check with parameterized options (-p \$PORT\_NUMBER). Now you are able to modify the port option parameter to every desired value without changing the check itself.

### Different warning value

For some reason we will create checks with 5 different warning values.

#### Solution fixed method:

You have to set up five different checks with five different warning option values. If there are even 10 different values you have a lot to do because you need to create 10 different checks.


#### Solution parameterized method:

You have to set up only one check with parameterized warning option value and change the parameter for each of the five different warning values. If there are also 10 different warning option values you only have to change the warning option parameter for each device instead of renew the check.

## 10.9.3. Advantages of parameterizing

The main advantage of parameterized checks in contrast to fixed defined checks is a more flexible way to handle checks. A direct influence on check options is also a benefit.

With parameterizing it is possible to change some check option values after creating it. Additional, it is faster to set option values than to set whole checks, so your administration effort decrease.

The bigger and more complex a  *Network* is, the more efficient it is to use parameterized checks.

An other advantage of parameterizing is the possibility to react faster in case of alternation established setup.

## 10.10. Distributed Monitoring

 COMING SOON

## 10.11. Evaluation of monitored data

 COMING SOON


## 10.12. Overview Configuring Nodes

There are three types of configuration data that can be associated with a configuration.

1. Variables
2. Monitoring Config
3. Scripts

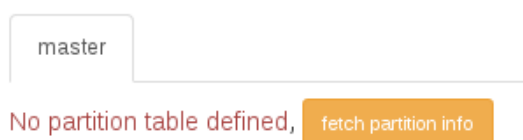
## 10.13. Discovery server

An other special feature of **NOCTUA®** is the ability to get partition data without any need to configure it. To get this feature run, only thing you have to do is to install the **discovery-server** on the machine you want and activate it.

Once it is installed you must activate it in deviceconfig like you do for RMS or Package-Install. (Klick on **device**  **Config** and on the blue arrow, select "discovery\_server")

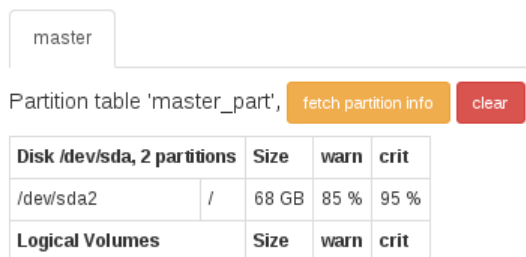
Now you can easily get partition data by pushing the **fetch partition info** button.

**Figure 10.29. Before fetching partition information**



Before fetching partition information

**Figure 10.30. After fetching partition information**



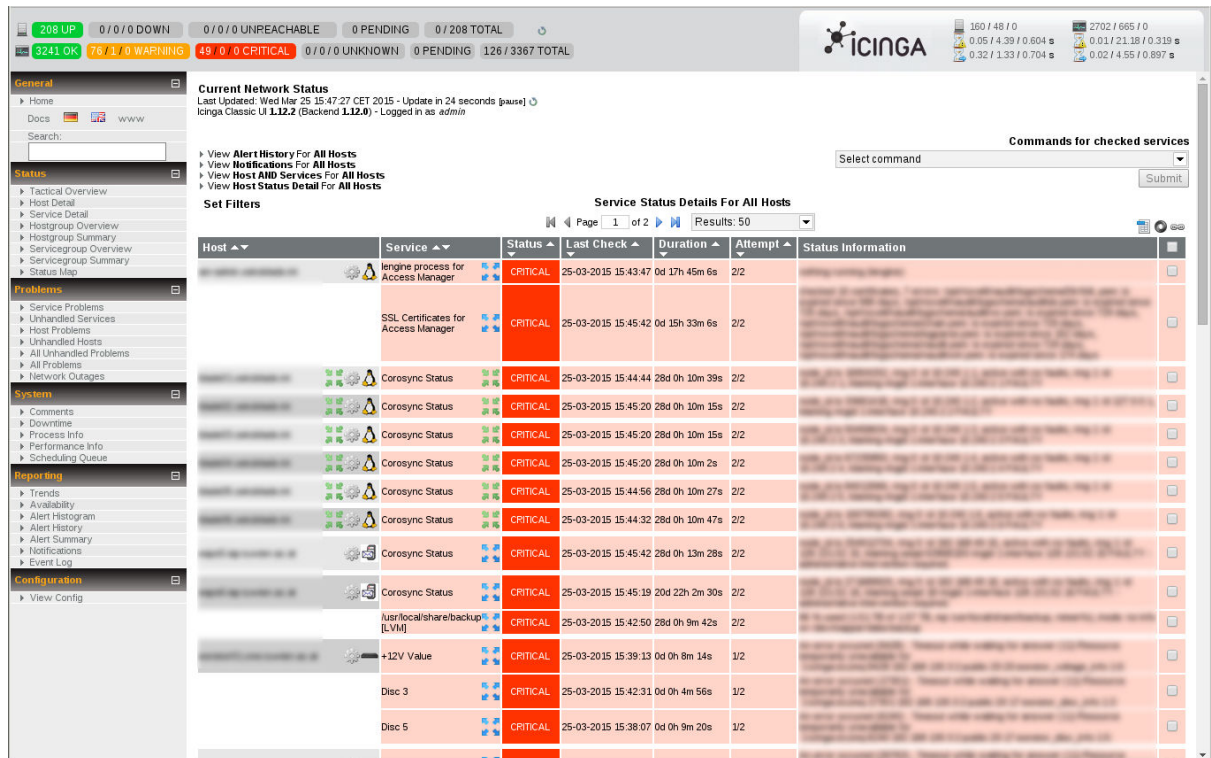
Disk /dev/sda, 2 partitions		Size	warn	crit
/dev/sda2	/	68 GB	85 %	95 %
Logical Volumes		Size	warn	crit

After fetching partition information

## 10.14. Icinga

If you are familiar with icinga it may be pleasant to read that **NOCTUA®** completely supports icinga. All devices/hosts and services inside of **NOCTUA®** are available by icinga. As monitoring backend and very important part of our monitoring solution we distribute icinga with every installation as part of it.

**Figure 10.31. Icinga, industry standard for monitoring**



Normal icinga front-end



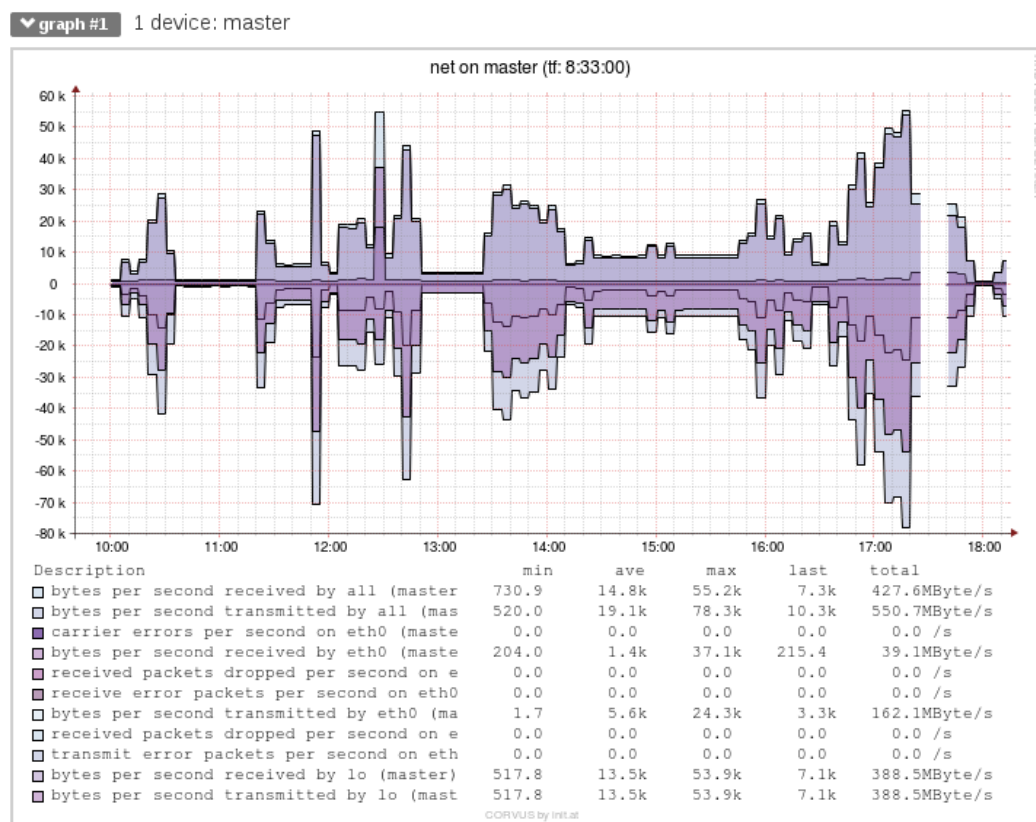


# Chapter 11. Graphing

## 11.1. Introduction to Graphing

Graphs are one of the most important tools for monitoring devices. They allow you to create graphs of collected data for different time ranges easily. You do not have to write couple of config files or modify existing one. All the configuration will be done by the web front-end, lean back and keep an eye of your automatic generated graphs.

**Figure 11.1. Typical rrd-graph**



Network traffic graph

Below the graph itself there is a legend and a table with numeric values. It contains following parts:

### RRD-graph legend

Description	Describes the color of lines or areas and corresponding data.
unit	Physical unit of displayed values.
min	Minimum value of displayed graph
ave	Average value of displayed graph
max	Maximum value of displayed graph
last	Last value in timeline of displayed graph

total

Total amount of displayed graph

### 11.1.1. Principles of RRD

RRD stands for **R**ound **R**obin **D**atabase and is a special designed database structure to collect data circular. That means that the database must be setup for the right amount of data which should be collect.

For that reason there are following advantages and disadvantages:

- No danger to overfill database
- After some time data will be overwritten and can not more be displayed with higher resolution.

But the monitoring software takes care for this details so you have not to agonize about it.

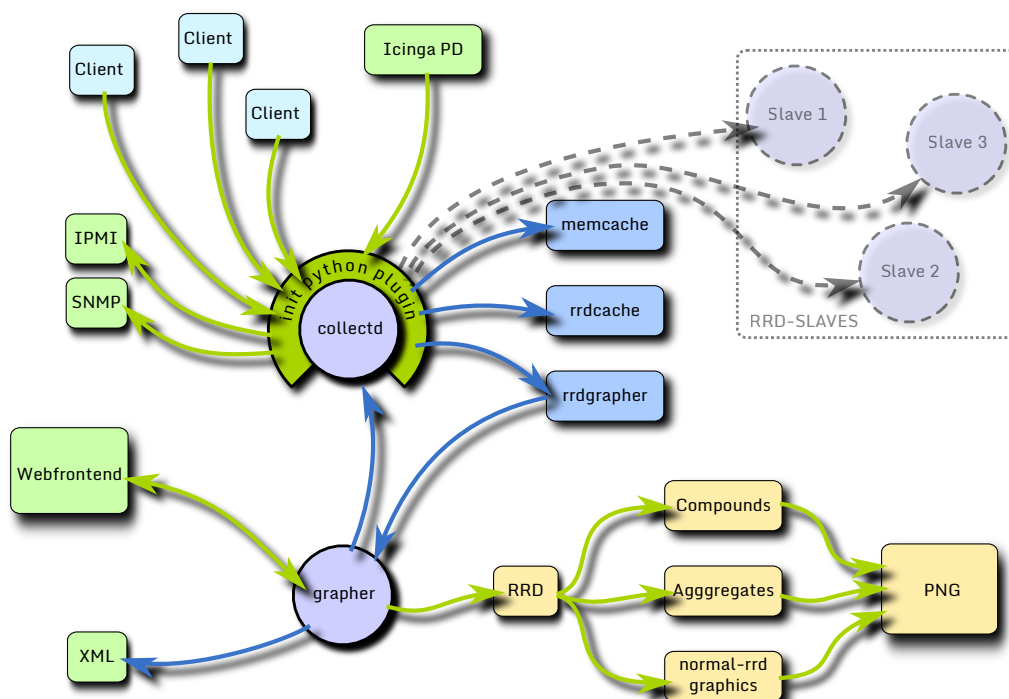
### 11.1.2. Data collection and graphing

To collect data and draw graphs in **NOCTUA®** there are more services appropriate for. Lower figure illustrates how they work together and how the dataflow between each other is.

Dotted parts are still in progress but will be very soon implemented into **NOCTUA®** because of better data flow distribution, less read/write access and therefor less load on the server.

Datatransfer should only takes place if rrd-graphs will be requested.

**Figure 11.2. RRD graph cycle**



Cycle of rrd graphic dataflow

To makes rrd-graphing work, the **rrd-grapher** service and **collectd-init** service must already run.



### 11.1.3. How to display RRD graphs?


Select one or more devices you want RRD graphs for and click either on the "house" button in top menu or on the green "use selection" button below the top menu.

If you only need RRD graphs for one device, just click on the device name in the device tree view.

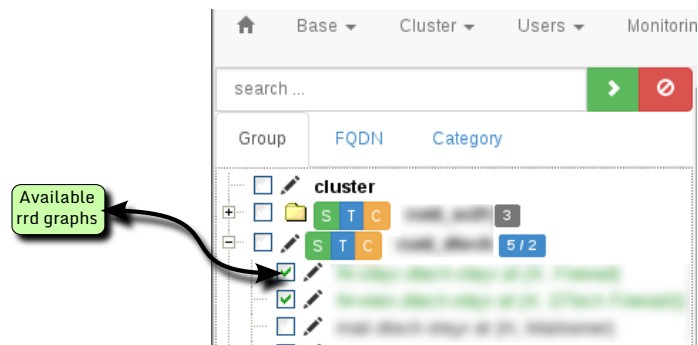
In both cases there will be some new tabs displayed, one of them named **Graphs**

### 11.1.4. RRD frontend

#### Availability of rrd graphics

RRD data is not collected mandatory for every device. To find out if there are some rrd-graphs for devices, look for a pencil logo beside the name of the device in the devicetree.  **rrd\_pencil**

**Figure 11.3. Available rrd graphs**



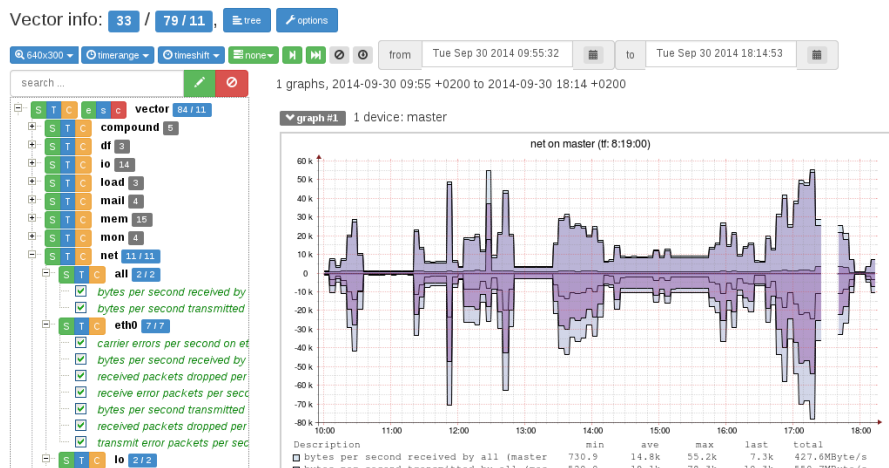
Existing rrd graphs marked with pencil logo.

### Overview

The rrd frontend follows the same structure like other parts of **NOCTUA®**. There are buttons, lists selections, inputfields and if drawn of course the graphs itself.

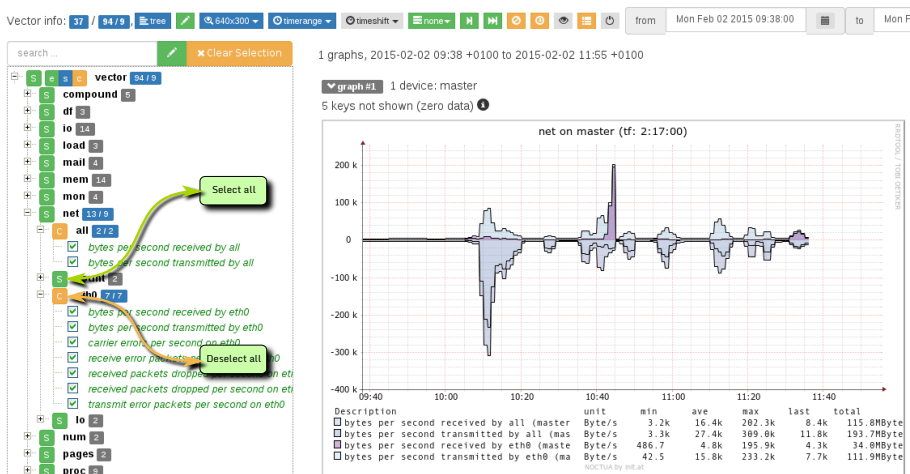
Also there is a tree on the left side, but this time not for devices but for monitored or collected data.

Figure 11.4. RRD classic front-end with three button selection method



Front-end inside of NOCTUA®

Figure 11.5. RRD front-end with one button selection method



Front-end inside of NOCTUA®

## Graphic size

640x300

The size in pixel the output graph will be. This size relates only for the graphs, not for legend. Keep this in mind if you want to insert graphs somewhere else.

## Output graph size

- 420x200, 640x300, 800x350, 1024x400, 1280x450

## Timerange

timerange

Selection which timerange should be displayed. There are "last" and "current" selections.

## Timerange

last 24 hours

draw graphs of the last 24 hours from now ((now-24h) - now)

<b>last day</b>	draw the whole last day (00:00 -23:59)
<b>current week</b>	draw the whole current week (sunday -saturday)
<b>last week</b>	draw the whole last week (sunday -saturday)
<b>current month</b>	draw the whole current month
<b>last month</b>	draw the whole last month
<b>current year</b>	draw the whole current year (Jan - Dec)
<b>last year</b>	draw the whole last year (Jan - Dec)

## Timeshift

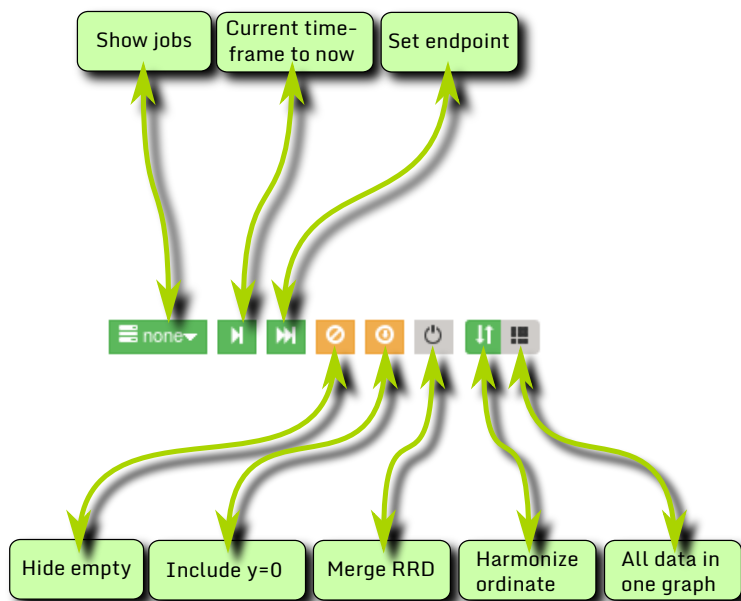
With the timeshift option you get a tool in your hands to map current graphs on future timeline. For example this is handy to compare current graphs with graphs drawn 1 week ago.

### Timeshift

<b>none</b>	do not draw extra comparing graphs
<b>1 hour</b>	draw one normal graph () plus the same graph 1 hour later (dotted)
<b>1 day</b>	draw one normal graph plus the same graph 1 day later (dotted)
<b>1 week</b>	draw one normal graph plus the same graph 1 week later (dotted)
<b>1 month</b>	draw one normal graph plus the same graph 1 month (31 days) later (dotted)
<b>1 year</b>	draw one normal graph plus the same graph 1 year (365 days) later (dotted)

## Controlbuttons

Figure 11.6. Options



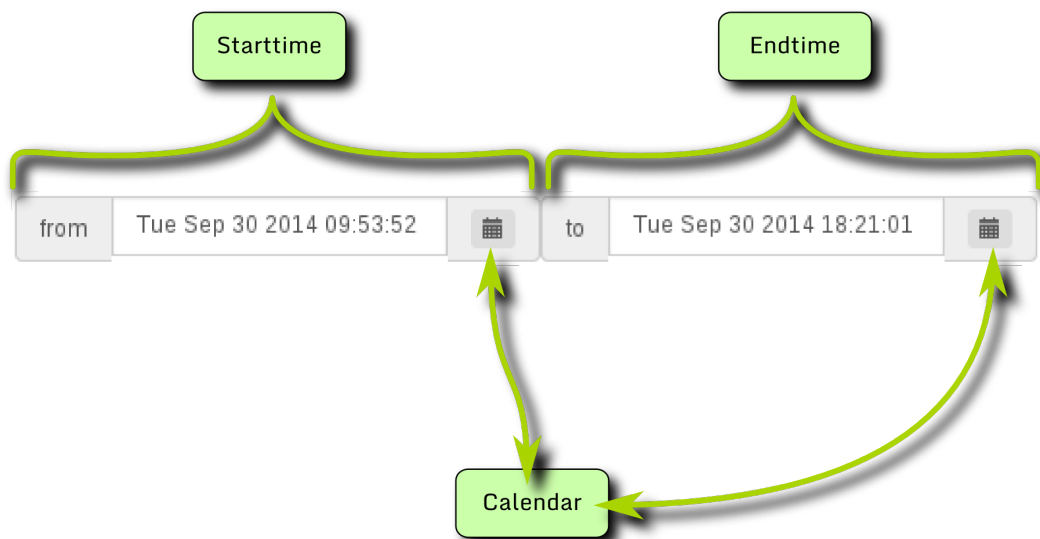
Additional options

<b>Show jobs</b>	Show specific jobs
<b>Current timeframe to now</b>	Set the current timeframe to now
<b>Set endtime</b>	Set the endtime of the graph
<b>Hide empty</b>	Hide empty graphs
<b>Include y=0</b>	Always include y-axis = 0 into graph
<b>Merge RRD</b>	Merge RRD from controlling devices
<b>Harmonize ordinate</b>	harmonize ordinate, for direct comparison with other graphs
<b>One graph for all devices</b>	Draw one graph for all devices

By default, no IPMI input data of controlling device will be shown in RRD tree. To display IPMI sources in RRD tree select the **Merge RRD** button and reload the page or push the green arrow button.

**Ping** data belongs to IPMI and will also be shown in RRD tree after selecting **Merge RRD** button.

Figure 11.7. Date section



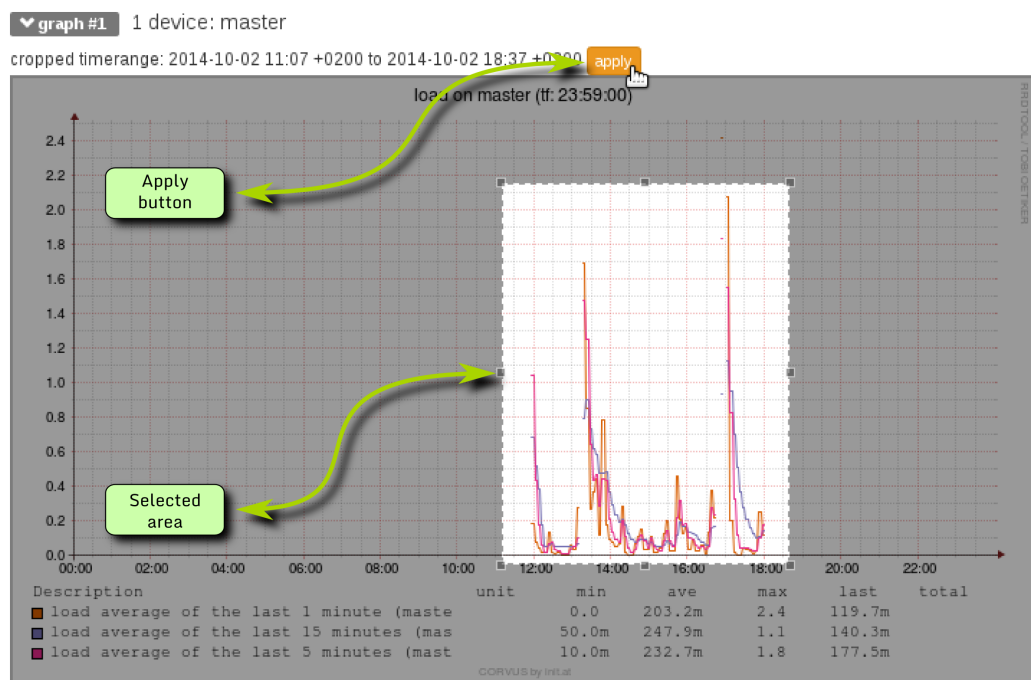
Start and endpoint for drawing date

## Zoom into graph

Apart from typing starttime and endtime of graph into the inputfield or picking the start and endtime from calendar, you can also select timearea direct from the graph itself. To zoom into desired timearea, simply move your mouse over the graph, the mousearrow changes to a cross hair and now you are able to draw a rectangle field over the graph. At the same time the area outside the selection gets darker. After releasing the mousebutton, area can be moved around or resized.

Push the **apply** button to zoom into selected area or use **Esc** key to abort selection.

Figure 11.8. Zoom area



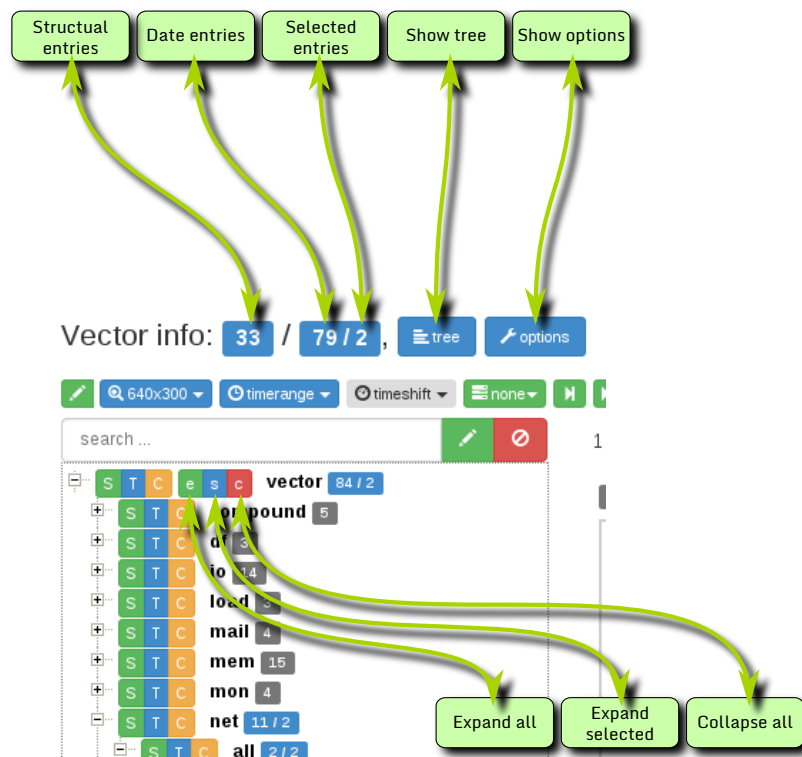
Zooming area ready for apply

**IMPORTANT**

⚠ If more than one graph is displayed, zooming into timearea of one graph affects all other graphs.

**Treeview**

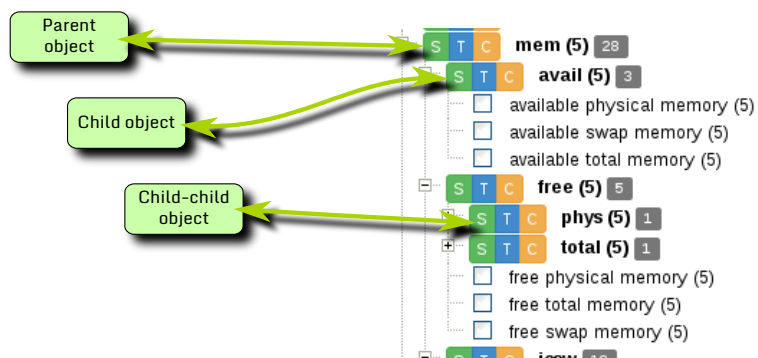
Monitored rrd-data is organized as tree. Corresponding data is stored in the same branch.

**Figure 11.9. Tree**

Tree organisation of rrd-data

**11.1.5. RRD tree components**

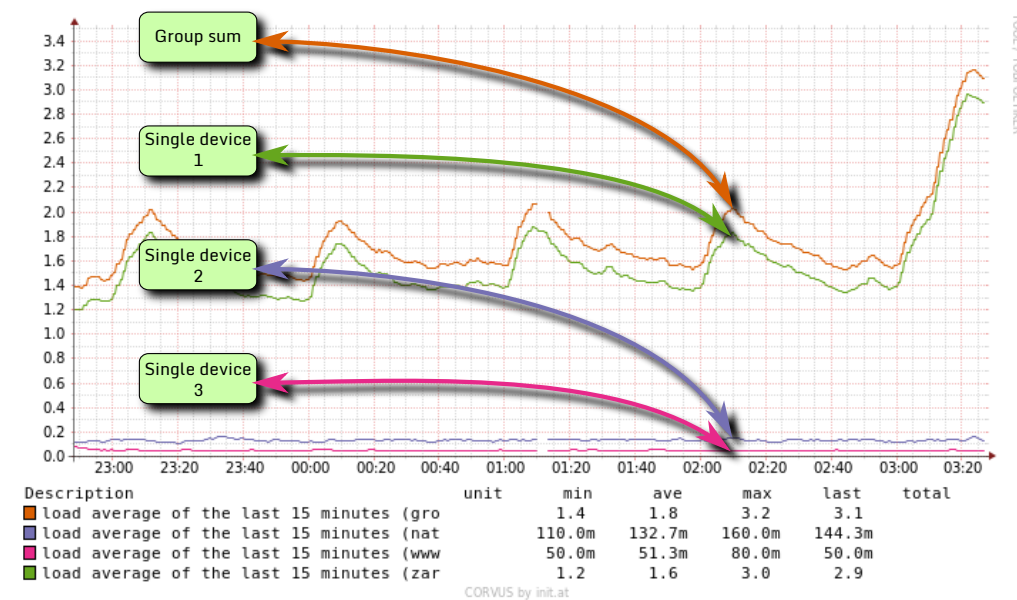
The rrd tree is, similar to the device tree, a overview. The difference is that a parent object can consists of one or more child objects. For instance the parent object **mem** contains 4 child objects, **avail**, **free**, **icsw** and **used**.

**Figure 11.10. Tree parents and childs**

Showing tree parents und children

## 11.1.6. Summarized graphs

Figure 11.11. Aggregation of 3 devices



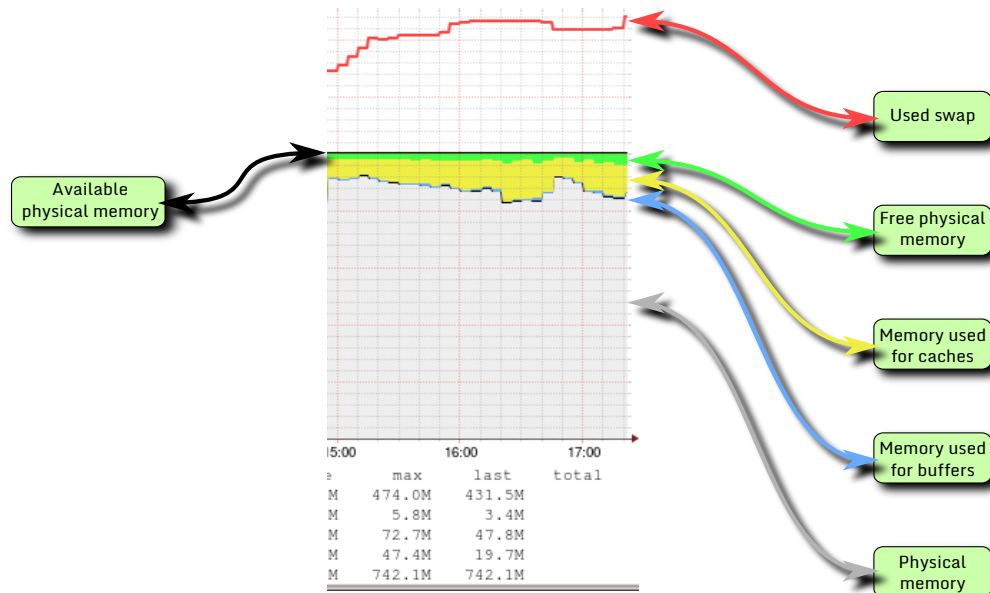
### Data aggregation of 3 devices

For parent groups it makes sense to summarize some graphs. This kind of summarization is called *aggregation* in **NOCTUA®**. Best way to get group information or to get overview of a cluster is to use aggregation. Aggregation is more complex than simple addition of data. It manages interferences and calculates values the most effective way to display sums as realistic as possible.

Above graph shows the 15 minutes single load value for 3 devices (pink, purpur and green) and a combined sum graph (brown) of all device graphs.

## 11.1.7. Compound graphs

Figure 11.12. Compound memory graph



Compound memory graph with stacked graphs.

Compound view can be found on top of the monitoring data tree. It combines several data (for example load, cpu, processes, memory and io) on one multigraph, no need to select  $n$  graphs.

An other advantage of compund graphs is stacking. Some graphs are more significant if values are displayed stacked.

Above figure explains stacking in context of memory graphing.

In this example you can see straightaway the parts of memory usage in relation to available memory.



# Chapter 12. Device localisations


One of the most interesting question admins wondering about is where monitored devices are located. Location means on the one hand the real **physical** position of devices.

On the other hand location could be **structural** location representing network infrastructure in context of functionality not in context of realistic physical locations or network connections.

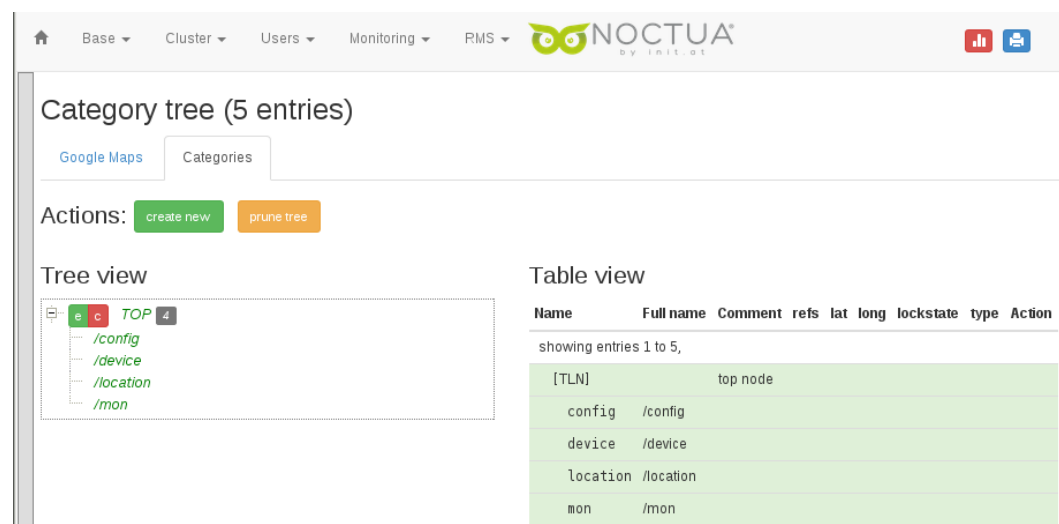
No matter if structural or physical locations, both of them have to be configured the same way.

## 12.1. Setup localisation

To add new device locations first of all we must create a new entry into the category tree. For this step you can, but do not have to select any device before.


Navigate to **Base**  **Category tree** and choose the **Categories** tab.

**Figure 12.1. Location setting**



Location settings inside category tree

Left click  on **create new** button, a new window appears below. Enter a new category name and choose *location* as parent category.

For advanced settings of new created category entry click left  onto the category in category tree or push the **modify** button beside.

### Advanced location settings

<b>Basic settings</b>	Name of category tree entry and its parent category
<b>Latitude / Longitude</b>	Coordinates for defined google map points
<b>Locked</b>	Checkbox to lock google map points in place
<b>physical</b>	Checkbox to define location as physical one

**Figure 12.2. Advanced location setting**

Category details for  
'LOCATION\_VIENNA\_1'

Basic settings

Name\* LOCATION\_VIENNA\_1

Parent\* /location

Additional fields

Comment

Positional data

Latitude\* 48.1

Longitude\* 16.3

☐ Locked

☒ Physical

Submit close delete

Advanced location settings

### 12.1.1. Upload and edit user images

If we go back and choose the **Google maps** tab, we notice a red Flag onto the google map and also two new buttons, an icon and category name appeared beside the map.

The blue **locate** button zooms the map in. With the green **add location gfx** button you are able to upload user image maps in two steps:

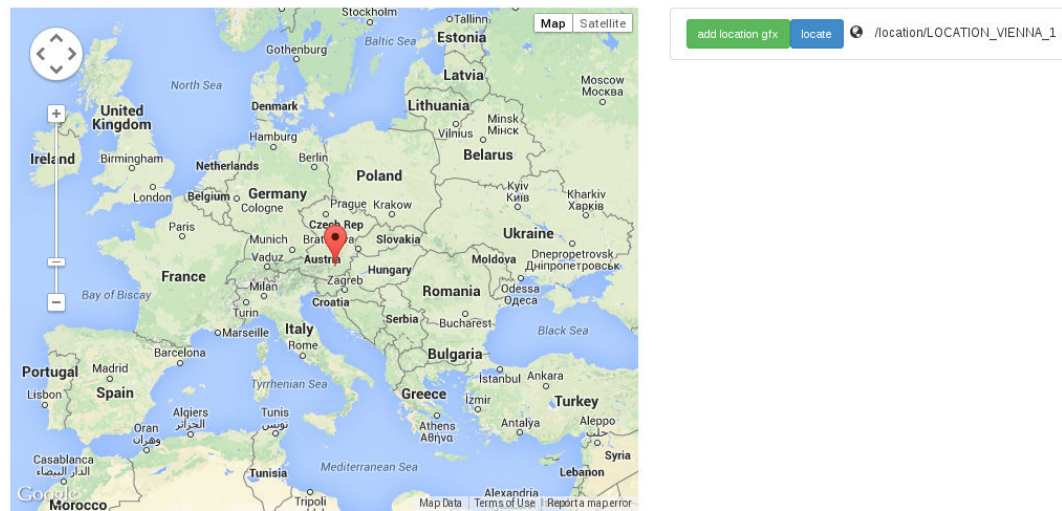
- Define Location graphic name

Once you named your new location graphics, a new **modify** button appears. Use the button to upload user images.

- Modify added graphic entry to upload user image

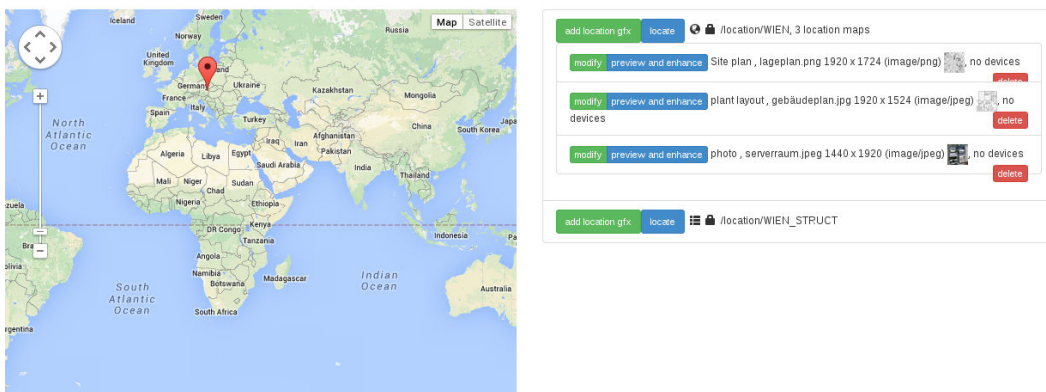
**Figure 12.3. Advanced location setting**

Map view, 1 locations

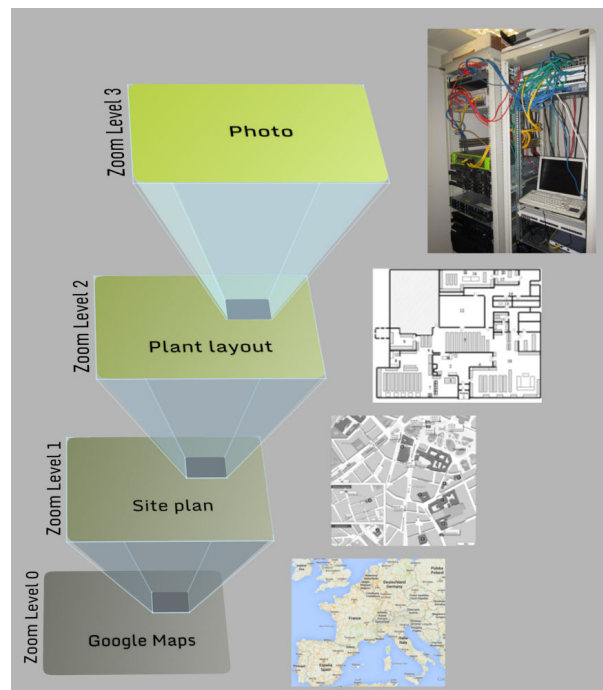


Advanced location settings

Of course you can add even more than just one user image, so you can create a stepwise zooming from google map to detailed server room photographs.

**Figure 12.4. Three user images added to location**

Zoom levels with according user image maps

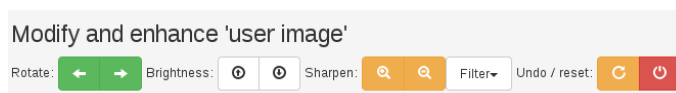
**Figure 12.5. Concepts of zoom levels with multiple image maps**

Zoom levels with according user image maps

## 12.1.2. Edit uploaded photos

The web front-end allows you also to edit uploaded images with the **preview and enhance** button.

Following self-explanatory buttons are accessible if you want to edit your uploaded image for quality reasons.

**Figure 12.6. Accessible buttons to modify user images.**

Zoom levels with according user image maps

Following editing buttons are integrated:

- left/right rotation (rotates image 90° clockwise or counter clockwise)
- increase/decrease image brightness
- sharpen/unsharpen image
- Filter (includes a bunch of predefined filter for)
- undo (undo last editing action)
- restore original image

## 12.2. Livestatus integration in maps

With localisation it is not only possible to display and locate the exact position of devices in different zoom levels, but also the status of monitored devices. That way you can get the best possible overview of your server-room for example.

### 12.2.1. Binding livestatus burst to maps

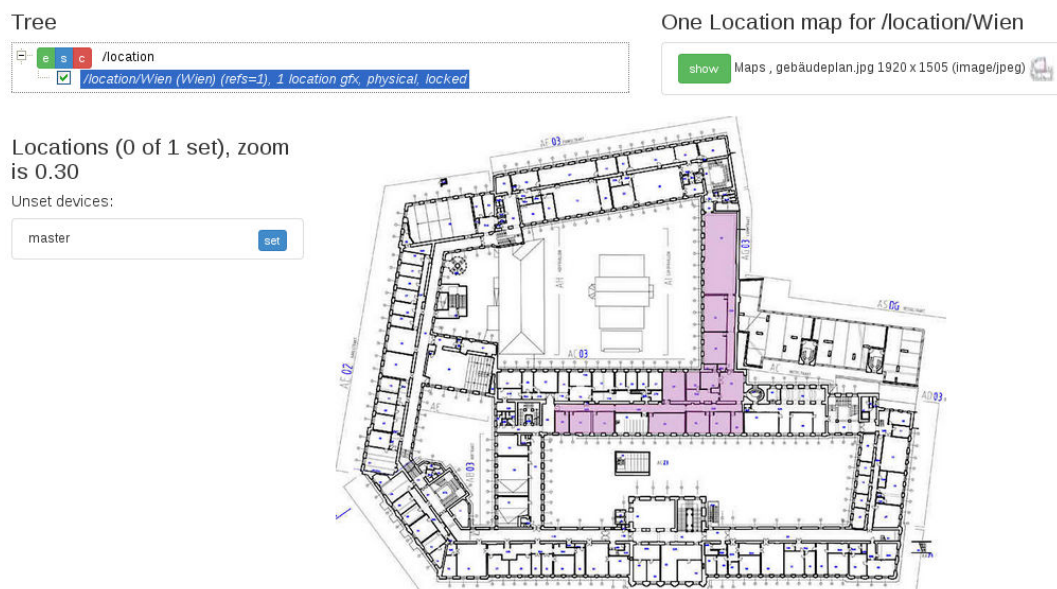
Once you have created new location categories and added some photos or images, you can easily add device livestatus to it.


Select all devices you wish to add and click either the **home** button or **use selection** button.

Navigate to the **Location** tab select the checkbox and left click on the location category. It appears a **show** location map button on the right side with some information about the image and a small preview of it. Push the button to show the image map.

Now you can place your livestatus burst on the right place at the image by clicking on the **set** button.

**Figure 12.7. Adding livestatus to image maps**



After placing livestatus burst on the right place left click  on the **lock** button to prevent the livestatus burst from moving.

Use the **remove** button to remove livestatus burst from image.

### 12.2.2. Display livestatus burst

Select your desired device and choose the livestatus view to display livestatus burst on imagemap. If there are more than one assigned location map, there will be tabs for each image map.

**Figure 12.8.** Adding livestatus to image maps



# Chapter 13. SNMP discovery

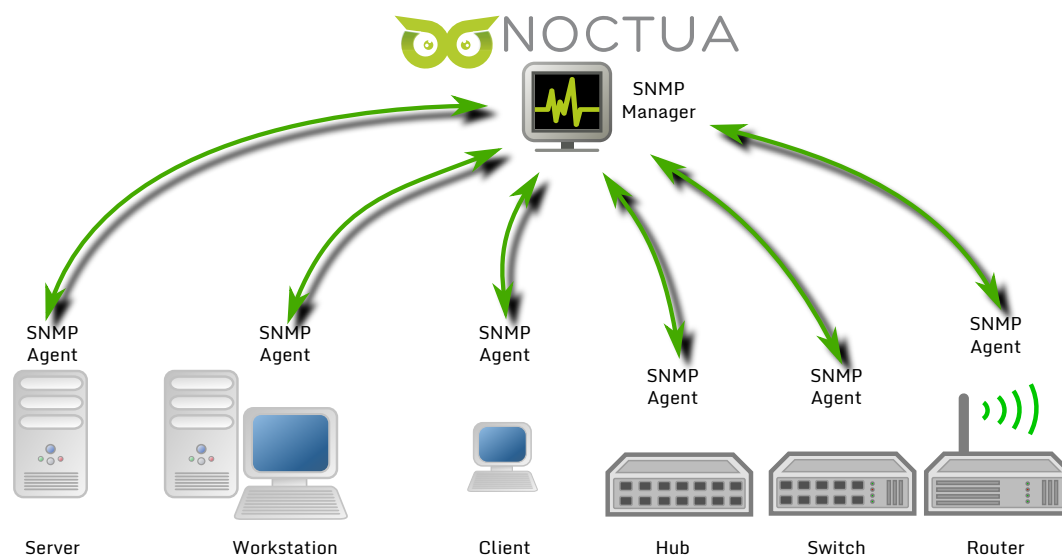
The Simple Network Management Protocol is a official RFC internet-standard-protocol which is designed to handle variables of devices like switches, router, server, workstation, printer, bridges, hubs, and more.

Variables contain hardware information and configuration of devices and can be picked up manually by special SNMP commands like **snmpwalk**. NOCTUA® implements SNMP as "autodiscovery" service, capable to scan network devices and get as much information about it as possible.

In the context of monitoring, snmp can deliver a huge amount of information about devices. Unfortunately there are some differences of implementation from several hardware vendors, as a result it is very difficult extracting useful and realistic data out of the snmp stack.

For this reason, NOCTUA® uses some intelligent algorithm and filter to avoid insertion of faulty data into the database.

**Figure 13.1. SNMP Agents and Manager**



Agents and Manager


## 13.1. Automatically network discovering with SNMP

To get SNMP data from devices, first of all target devices are required to provide such SNMP data. Most hardware in the network segment like swithes, router, server, printer, etc... provide SNMP by default.

For operating systems like windows or SUSE/RedHat machines, there are SNMP daemons which fist have to be started before they provide SNMP data.

Please read your operating system documentation or contact your administrator to find out how to activate SNMP daemon on your machines.


### 13.1.1. Setup of SNMP discovery

To activate SNMP discovery for one device, simply select the checkbox **Enable perfddata, check IPMI and SNMP**. To get this checkbox, either select your device and left click  the home icon on top, or double click the device.

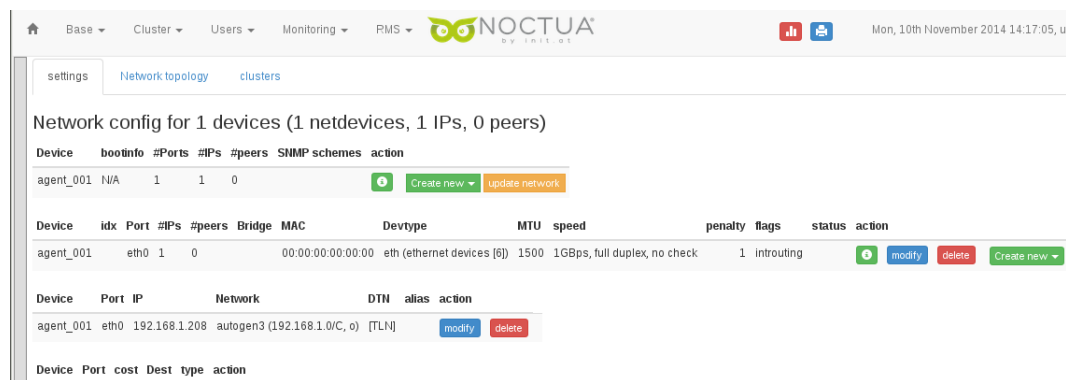


## 13.1.2. Auto discover with SNMP

To reach SNMP scan, go to **Base**  **Device network**.

There are no SNMP schemes yet in the settings window. Now perform a SNMP scan with left click  on the orange **update network** button.

**Figure 13.2. Auto discover with SNMP**



Button to auto discover network

It appears a SNMP setting window, where you are able to adjust some basic settings.

**Figure 13.3. SNMP scan settings**

SNMP scan settings

### Settings

<b>IP</b>	The IP address of the device, a valid domainname or a valid host name.
<b>Snmp community</b>	SNMP security settings, either public or private
<b>Snmp version</b>	Number of snmp version, either 1 or 2
<b>Remove not found checkbox</b>	If this flag is marked, previously done config, SNMP auto discovery scan can not reading out will be deleted.



Depending on your network size and structure, it takes some time to get complete SNMP data tree, apply filter and algorithms to it and write the extracted data into the database.

After performing SNMP scan, you will get some new network config entries for the scanned device.

**Figure 13.4. Before SNMP scan**

Device	bootinfo	#Ports	#IPs	#peers	SNMP schemes	action
agent_001	N/A	1	1	0		<a href="#">Create new</a> <a href="#">update network</a>

Device	idx	Port	#IPs	#peers	Bridge	MAC	Devtype	MTU	speed	penalty	flags	status	action
agent_001		eth0	1	0		00:00:00:00:00:00	eth (ethernet devices [6])	1500	1GBps, full duplex, no check	1	introuting	up	<a href="#">modify</a> <a href="#">delete</a> <a href="#">Create new</a>

Device	Port	IP	Network	DTH	alias	action
agent_001	eth0	192.168.1.208	autogen3 (192.168.1.0/C, o)		[TLN]	<a href="#">modify</a> <a href="#">delete</a>

Network config for one device before scan

**Figure 13.5. After SNMP scan**

Device	bootinfo	#Ports	#IPs	#peers	SNMP schemes	action
agent_001	N/A	5	3	0		<a href="#">Create new</a> <a href="#">update network</a>

Device	idx	Port	#IPs	#peers	Bridge	MAC	Devtype	MTU	speed	penalty	flags	status	action
agent_001	1	lo	1	0			softwareLoopback	65536	10MBps, full duplex, check via ethtool	1	introuting	up	<a href="#">modify</a> <a href="#">delete</a> <a href="#">Create new</a>
agent_001	2	enp3s0	1	0		54:04:a6:4c:b2:ae	etherneCsmacd	1500	1GBps, full duplex, check via ethtool	1	introuting	up	<a href="#">modify</a> <a href="#">delete</a> <a href="#">Create new</a>
agent_001	3	transbridge	1	0		54:04:a6:4c:b2:ae	etherneCsmacd	1500	unspec., full duplex, no check	1	introuting	up	<a href="#">modify</a> <a href="#">delete</a> <a href="#">Create new</a>
agent_001	4	vnet0	0	0		fe:54:09:32:00:dd	etherneCsmacd	1500	10MBps, full duplex, check via ethtool	1	introuting	up	<a href="#">modify</a> <a href="#">delete</a> <a href="#">Create new</a>
agent_001	5	vnet1	0	0		fe:54:09:32:02:dd	etherneCsmacd	1500	10MBps, full duplex, check via ethtool	1	introuting	up	<a href="#">modify</a> <a href="#">delete</a> <a href="#">Create new</a>

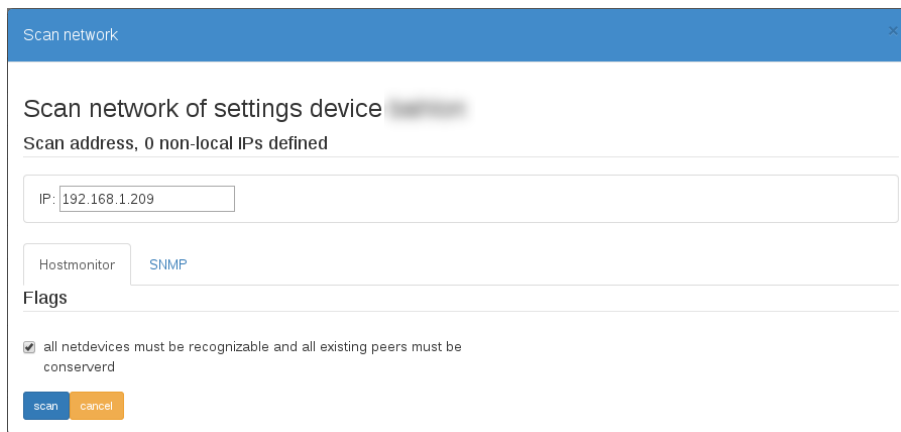
Device	Port	IP	Network	DTH	alias	action
agent_001	lo	127.0.0.1	autogen1 (127.0.0.0/A, o)		[TLN] localhost (exclusive)	<a href="#">modify</a> <a href="#">delete</a>
agent_001	enp3s0	169.254.9.99	autogen2 (169.254.0.0/B, o)		[TLN]	<a href="#">modify</a> <a href="#">delete</a>
agent_001	transbridge	192.168.1.208	autogen3 (192.168.1.0/C, o)		[TLN]	<a href="#">modify</a> <a href="#">delete</a>

Device network config after successful SNMP scan

That way you automatically get a couple of netdevices with according names, values, MAC addresses, MTU values, speed, etc... without to invest much time or manpower. A very handy and timesaving tool for administrators.

### 13.1.3. Auto discover with host-monitoring

Alternative to SNMP scan there is a tab for scan with host-monitoring. This scan works only if the **host-monitoring** service is installed and running on the host machine. You can find more information about host-monitoring in [Section 10.1.3, “Advanced Monitoring with host-monitoring”].

**Figure 13.6. host-monitoring scan settings**

Scan network

Scan network of settings device XXXXXXXXXX

Scan address, 0 non-local IPs defined

IP: 192.168.1.209

Hostmonitor **SNMP**

Flags

☒ all netdevices must be recognizable and all existing peers must be conserved

scan cancel

host-monitoring scan settings

The flag **all netdevices must be recognizable and all existing peers must be conserved** ensures that existing peers (network topology connections) will not be deleted after scan.

# Chapter 14. Debugging and Error hunting

## 14.1. General information

To obtain information about the general status of your server use **icsw service status**.

**Example 14.1. Using icsw status to view services status**

**Figure 14.1. icsw status command**

```
Wed, 20. May 2015 20:15:21:28
Name      type      source License status
-----
hoststatus client simple - running
logging-server client meta - running
meta-server client meta - running
host-monitoring client meta - running
package-client client meta - running
logcheck-server server meta - running
package-server server meta valid not configured
mother server meta - running
collectd server meta valid running
rrd-grapher server meta valid running
rms-server server N/A valid not configured
cluster-server server meta - running
discovery-server server meta valid running
cluster-config-server server meta - running
host-relay server meta - running
snmp-relay server N/A - error
md-config-server server meta valid running
memcached system simple - running
nginx system simple - running
icinga system simple - running
uwsgi-init system simple - running
rrdcached system pid - running
```

## 14.2. Show errors

To show the last errors from the logfile you can use **lse** .

**lse** [-l *Error number* ]

For more information type **lse --help** .

**Example 14.2. Using lse to display the last error**

```
clusterserver:~ # lse -l 1
```

Found 40 error records

Error 40 occurred yesterday, 17:12:47, pid 11507, uid/gid is (30/8 [wwwrun/www]), sou

```
0 (err) : IOS_type : error
1 (err) : args : None
2 (err) : created : 1409152367.94
3 (err) : exc_info : None
4 (err) : exc_text : None
5 (err) : filename : routing.py
6 (err) : funcName : _build_resolv_dict
7 (err) : gid : 8
8 (err) : levelname : err
9 (err) : levelno : 40
10 (err) : lineno : 179
```

```
11 (err) :      message      : device 'METADEV_server_group' (srv_type grapher) has
```

## 14.3. Node information

Retrieving node information in an automated fashion is often useful in hunting down errors and bugs. To retrieve information about the nodes use **collclient.py**.

```
collclient.py [ --host Nodename ] [command]
```

For more information execute **collclient.py --help**

### Example 14.3. Retrieving information from nodes

```
clusterserver:~ # collclient.py --host node01 df
```

## 14.4. Logging

The server provides its own logging service. Like usual in \*NIX environments there are special directories log-files will be written into. Access to these log files is given by the command **lse**. Of course it is also possible to read the logfiles directly by your favorite editor.

### 14.4.1. Directories for log files

In case of something goes wrong the logging-server writes its logs under `/var/log/cluster/logging-server/[HOSTNAME]/`.

Denotation of log files and subdirectories is related to the service which writes the log. For example if the meta-server can not start some service it will write its log into the directory

If you want to see background information for package-installation on some nodes the file you have to check is `package-client`. Analogue this is true for server side, this time the filename is `package-server`

Files called `*.bz2` are compressed logging backup files.

### 14.4.2. Automatic log mail delivery system

Critical errorlogs will also be delivered by mail. So you do not have to check your logs permanent, you will be notified by mail about critical errors.

Setting for recipient of errorlog mails is stored in `/etc/sysconfig/logging-server`.

Another configuration file for mail notification is `/etc/sysconfig/meta-server`.

Replace the given mailaddress in the line containing **TO\_ADDR=** with your desired mail address.

By uncommenting and editing the line beginning with **#FROM\_ADDR=** you are able to set the *sender "From"* name of recived emails.

```
# from name and addr
FROM_NAME=pythonerror
#FROM_ADDR=localhost.localdomain
# to addr
TO_ADDR=mymail@gmail.com
# mailserver
```

MAILSERVER=localhost

After editing the logging-server configuration file, the logging-server daemon must be restarted:

#### icsw service restart logging-server

The new configuration take effect after restart logging-server daemon.

### 14.4.3. icsw logwatch

A very handy command to read out logfiles is **icsw logwatch**. Logwatch makes it possible to display logs for different services and daemons at once. Even if you don't know in which file the logs are written to you are able to watch it. Thats the reason why logwatch is an allround tool for logging.

As usual for log files they have a typical output format.

**Table 14.1. Logwatch columns**

Column number	Column name	Example
1	Date and time	Thu Apr 09 17:58:41 2015
2	Device	2_5_branch
3	System (logging daemon)	/collectd-init
4	Node	/---
5	Loglevel	warn
6	Processname,processid	MainThread.19137
7	Logmessage	sending 733 bytes to vector_socket

Without any parameter logwatch.py displays the last 400 lines of logging messages for all services writing log-files. With **icsw logwatch -n 20** you can limit output lines to the last 20.

A very useful parameter for **icsw logwatch** is **--system-filter**. This flag restricts the output to one single daemon (service) e.g

```
icsw logwatch [ --system-filter rrd ]
displays only log messages related to an rrd (daemon) service.
```

With the **-f** flag it is possible to view logs in realtime. Use

```
icsw logwatch [-f] [ --machine MACHINE ] [ -n N ] [ --system-filter rrd ]
to output appended data as the file grows. Try icsw logwatch --help to list all possible options.
```

### 14.4.4. Service communication ports

In case of malfunction it is very likely that the portnumber will be written into a logfile or appears in the web front-end. To find out which service or process causes the error we have to know which service communicates on which port. Following table shows you a little summary of common services and their communicating ports.


**Table 14.2. Portnumber and services**

Service	Port
md-config-server	8010
rrd-grapher	8003, 8003
logging-server	8011


Service	Port
meta-server	8012
discovery-server	8006
cluster-server	8004

---

## Chapter 15. Extensions

One of the main advantages in contrast with proprietary software is the ability to extend or adapt functionality to user-defined targets. There are some documented APIs which allows you to customise and optimise the workflow and integration into your companys facility. 

### 15.1. Shared script directories

There is a place for user scripts under `/opt/cluster/share` 





# Chapter 16. Frequently Asked Questions

This is a Collection of repeated Questions.

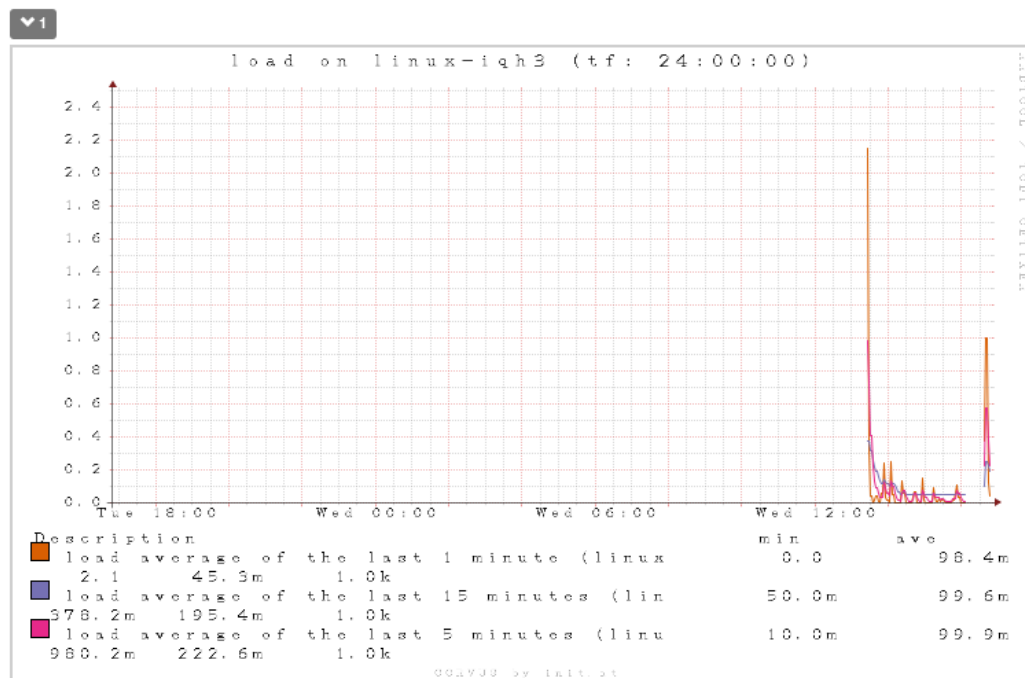
## 16.1. FAQ

### Miscellaneous questions

#### 16.1.1. Bad looking font in RRD Graph

16.1.1. Why are my fonts looks so ugly?

1.



ugly looking fonts due to wrong font setup

If you get something like in the picture above, you have to install **fetchmsttf** (OpenSUSE) and ... (debian) package.

#### 16.1.2. Server Error (500)

16.1.2. Why i get Server Error (500)?

1.

This is a server internal error, likely the server can't find some files. Take a look into `/var/log/nginx/error.log` for detailed error message. Also the `ls` command could be helpful.

#### 16.1.3. Unable to connect to the web front end

16.1.3. Why i can not connect to the web front end?

1.

For some reason the webserver nginx doesn't run. Start it manually, for example with "icsw service nginx start"

### 16.1.4. An error occurred

16.1.4. Why i get a message "An error occurred"?

1.

Please wait a moment till database connection is active and reload the page. If you still get this message after waiting a time you have to start uwsgi-init, for example with "service uwsgi-init start"

With **top** you can display a job list. If there is something like **yuglify** in the top row than wait some time until it disappears. After that and after reloading the page the error message should dissapears.

### 16.1.5. Configurations seems to be ignored

16.1.5. I changed my configurations but it seems to be ignored.


1.

For some changes in your configuration you have to **rebuild config (cached, RC)** first. If your config is stored in cache, you have even to **rebuild config (refresh)**

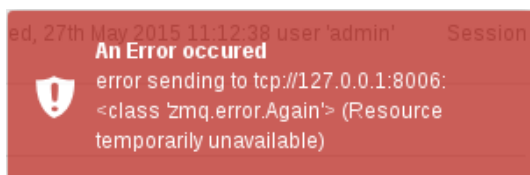
### 16.1.6. An Error occurred

16.1.6. Why does my discovery not working?

1.



Most likely the discovery-server service is not running. Make sure the discovery-server is installed and running. Run the **icsw service status** command and look for "discovery-server". If it is not running, start it either by commandline **icsw service start discovery-server** or via the webfrontend in top menu under **server information**  **One server checked**

**Figure 16.1. discovery-server not defined in routing**



Errormessage

An other possible reason for that malfunction could be disabled **discovery server** config for your monitoring server. To enable it select your monitoring server device, navigate to the config tab and select the **discovery server** config.

After that you have to wait some time or refresh the memcached by  left clicking on the server information button as described in  Figure 5.9, "Cluster server information "

### 16.1.7. Slow network topology graph

16.1.7. Why is my network topology graph so slowly?

1.

Sometimes complex network topology slows down display output in firefox. This issue affects firefox up to version 31.0. Reason is likely bad javascript interpretation on firefox side. If you get bad graphic display performance, try to use another browser e.g. chromium or google chrome™.

## 16.1.8. Lost password

16.1.8. Ilost my password, how can i get a new one?

1.

A short guide how to reset a login password by direct access to the database via **clustershell** follows:

1. Open a terminal (e.g. xterm, Konsole, gnometerminal) on your system and start the clustershell:

**clustershell**

```
Python 2.7.8 (default, Jul 29 2014, 08:10:43)
[GCC 4.8.1 20130909 [gcc-4_8-branch revision 202388]] on linux2
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>>
```

2. Import relevant database content

```
from initat.cluster.backbone.models import user
```

3. Define new variable to work with:

```
my_user = user.objects.get(login="admin")
```

4. Set your new password.

```
my_user.password="MY-new-_passW0Rd17"
```

### Important

⚠ Please set a secure password with more than 8 character

5. Control your new set password:

```
print my_user.password
```

6. Save your new created password to the database:

```
my_user.save()
```

7. Exit the clustershell

```
exit()
```

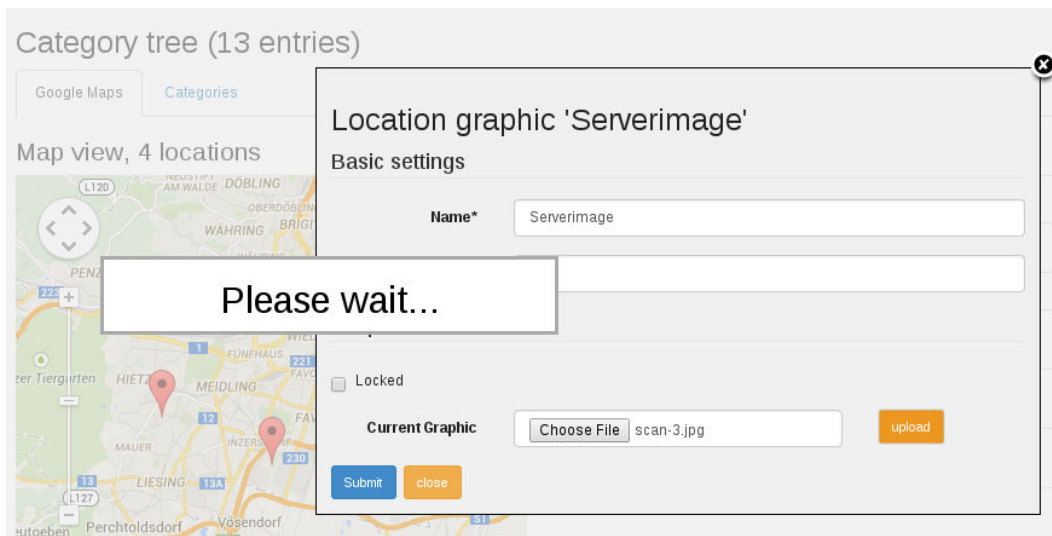
From now you are able to login with your new password.

## 16.1.9. "Please wait..." after add location gfx

16.1.9. What if "Please wait..." message occurs for longer time?

1.

If you must wait long time while pending upload and the infolabel "Please wait..." is shown after upload image with add location gfx button, reload the page to resolve this issue.

**Figure 16.2. Please wait ...**

"Please wait..." message after image upload

## 16.1.10. Weird mouse events on virtual desktop

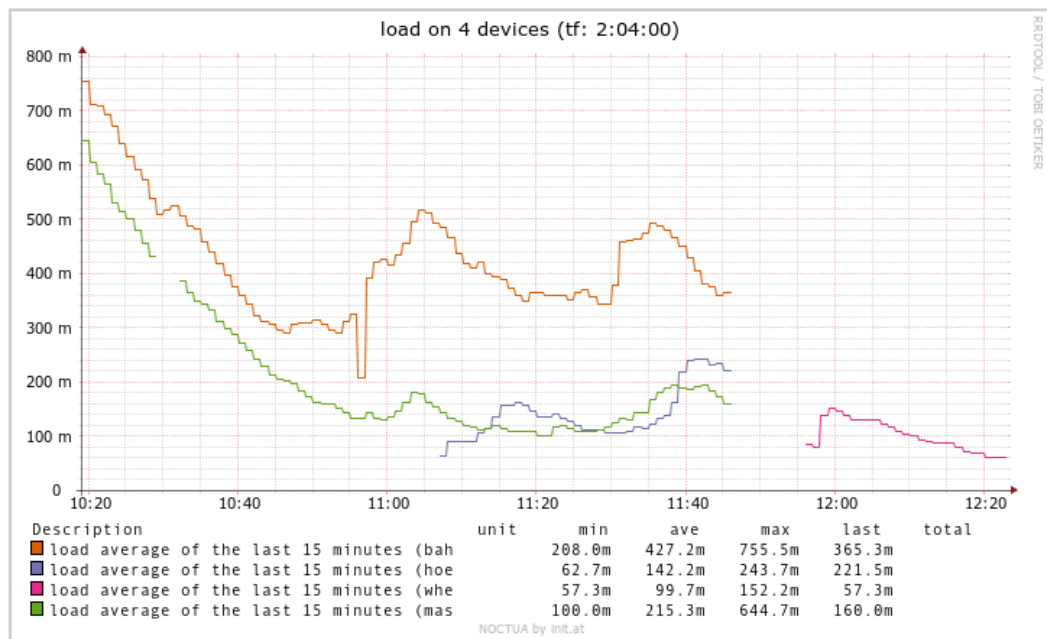
### 16.1.1 The mouse pointer position is wrong, what can i do to resolv this? 0.1.

Some vnc-server tends to break correct mouse pointer handling in virtual desktop. To get back correct mouse pointer, log out of your session and back in again.

## 16.1.11. Asynchron graphs

### 16.1.1 Why are my rrd graphs ascnchron? 1.1.

If you get wrong graphs, for example 1 hour in past or 1 hour in future like the pink graph line below, make sure to set the correct timezone and times on the affected machines.

**Figure 16.3. Wrong graphs due to wrong timezone**

Wrong drawn graphs because of wrong timezone

### Important

△Generally, make always sure to set the correct timezone and times on every machine. This is essential for a proper running monitoring or cluster management.

## 16.1.12. I have no permissions to icinga

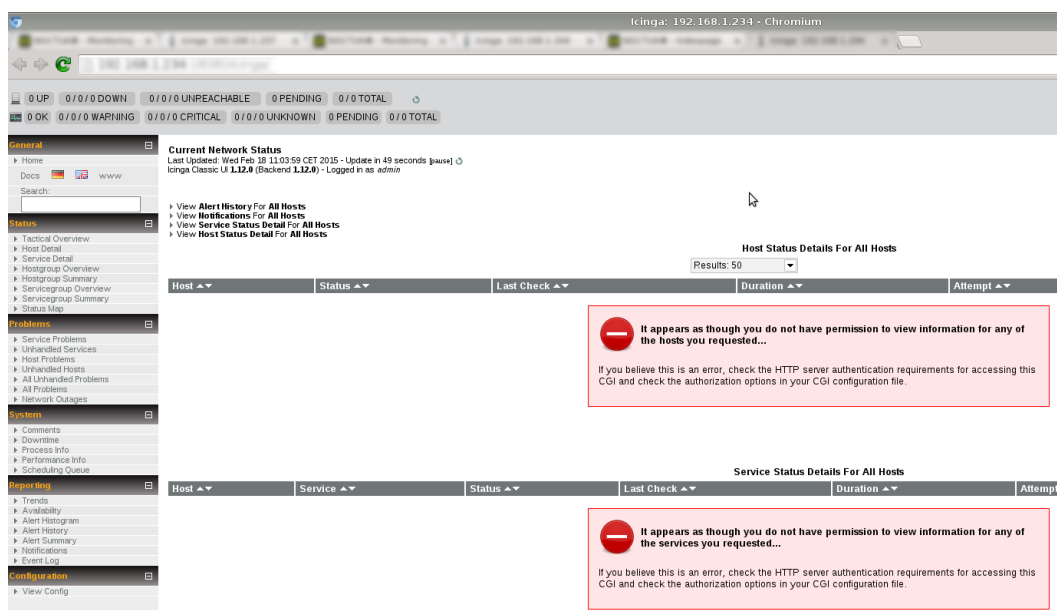
16.1.1 How can i get the right permissions to access the icinga view?

2.1.

To get the right permissions to icinga, you have to have at least one contact in **Monitoring** **Basic setup** **Contacts** and you must be logged in with this contact.

Rebuild your icinga config to apply your new contact entry. **Monitoring** **rebuild config (refresh)**

Figure 16.4. No permission to icinga



Without at least one contact you are not able to use the icinga view

## 16.1.13. Can not reach any network devices

### 16.1.1 Why i can not reach any network device? 3.1.

⚠ Due to the fact that your network will be mapped into the database, you have to make sure your server finds itself in the database.

To make this sure it is essential to name your **server device** equal as your **server hostname**.

## 16.1.14. Unable to delete group from device tree

### 16.1.1 How can i delete groups? 4.1.

There is no delete button visible for preselected device groups in device tree.

Figure 16.5. Deleting groups



<div> <span>+ create device</span> <span>+ create devicegroup</span> <span>↗ modify selected</span> <span>🗑 delete selected</span> </div>						
showing entries 1 to 5, show <input type="text" value="20"/> per page,						
Additional columns: TLN RRD store IPMI capable Password MonMaster BootMaster						
Name	Sel	Description	Enabled	Type	Action	
cluster		ClusterGroup			↗ modify	
192.168.1.234	S T C	Nagios Server ZID UW	yes	1	↗ modify	+ create device
192.168.1.234	S T C	Kunden Windisch	yes	1	↗ modify	+ create device
192.168.1.234	S T C	init server interxeon	yes	1	↗ modify	+ create device
192.168.1.234		KVM Server	yes	0	↗ modify	🗑 delete + create device

Reason for this behavior is that there are disabled devices in this group. First delete this disabled devices and finally you are also able to delete the group.

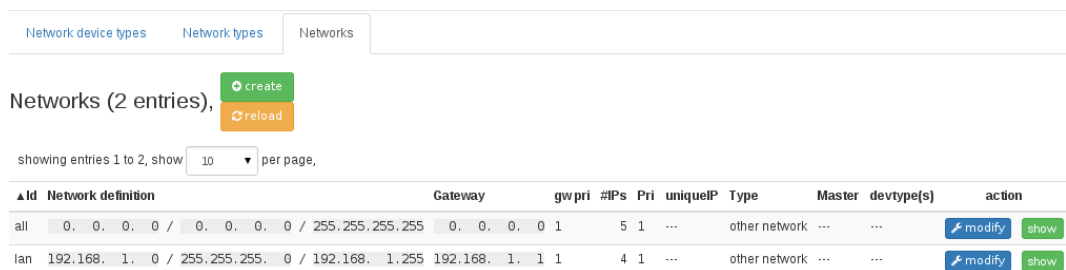
### 16.1.15. No "IP address" dropdown in device network

16.1.1 Why there is no "IP address" dropdown in device network??

5.1.

There is no "IP address" dropdown button visible in device network until at least one network is defined in Base  Networks  Networks

**Figure 16.6. Right networks setup**



Id	Network definition	Gateway	gw pri	#IPs	Pri	uniqueIP	Type	Master	devtype[s]	action
all	0. 0. 0. 0 / 0. 0. 0. 0	255.255.255.255	0. 0. 0. 0	1	5	1	other network	...	...	<a href="#">modify</a> <a href="#">show</a>
lan	192.168. 1. 0 / 255.255.255. 0	192.168. 1.255	192.168. 1. 1	1	4	1	other network	...	...	<a href="#">modify</a> <a href="#">show</a>

Two defined networks

### 16.1.16. Could not connect to server: Connection refused

16.1.1 I get server error on port 5432, how can i resolv the problem?

6.1.

After running `icsw service status` script, a python error with Port `5432` occurs. Generally, if you get error messages with the port number **5432**, the reason is likely that your postgres server (listens by default on port 5432) is down.

```
[...]
django.db.utils.OperationalError: could not connect to server: Connection refused
Is the server running on host "localhost" (::1) and accepting
TCP/IP connections on port 5432?
```

To check if your postgres database server is running type in one of the following commands, depend on your os:

**rcpostgres status**

**service postgresql status**

**systemctl status postgresql.service**

Make sure to start the postgres server at boot time by enabling your operating system start scripts. Replace `status` with `start` to start the database server.

Also make sure the service starts after rebooting the *system*.

### 16.1.17. Internal Server Error

16.1.1 I get an internal server error in web front end, how can i resolv the problem?

7.1.

After server installation and database setup, it could happen that you get an **Internal Server Error**. Try restarting your **uwsgi-init** service with this command: **rcuwsgi-init restart**.

The restart of uwsgi service results in an running **yuglify** process which generates all static files. After all static files were generated, you should get access to your web front-end.

## 16.1.18. License warning appears on every page

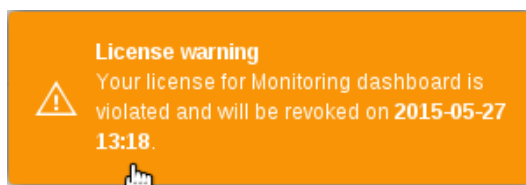
**16.1.1** I get an license warning message on every single page, how can i resolve this?  
**8.1.**

A license warning message appears on the top right side, each time a new page is loaded.

This warning means that one of your licenses is either out of date and for this reason in grace time, or used devices/services/user for this license exceeds its license limitation. In both cases the license is violated and from that moment the grace period begins to run. Grace period for licenses is **2 weeks** long.

In grace period, functionality of the software is as usual but you will see this license violation warning on each new loaded page

**Figure 16.7. License warning**



License violation warning

You have to get a new license or expand your existing one to avoid the license violation message. Please contact us by mail <support@init.at> or by phone +43-1-522 53 77 to request for a extended licenses.

An other method to get again into a valid license limitation is to **lock** licenses for some devices.

## 16.1.19. Reverse domain tree node order

**16.1.1** How can i reverse the domain tree node order?  
**9.1.**

Suppose you have a couple devices with following domain tree node order:

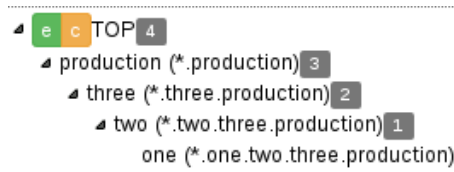
**Figure 16.8. Device domain name**

```
test_01.one.two.three.production (new device)
test_02.one.two.three.production (new device)
test_03.one.two.three.production (new device)
test_04.one.two.three.production (new device)
test_05.one.two.three.production (new device)
```

Original device domain name

Domain tree node structure for above device domains looks like this:



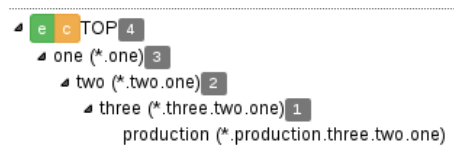
**Figure 16.9. Domain name tree**

Original domain name tree

Now for example you want to revert the order of the device domains.

First step you have to do is to change the domain name tree itself. Navigate to **Base** **Domain name tree** and modify the entry called **one** and choose as parent the top level node [TLN].

Do the same with the other entries until your tree looks like this:

**Figure 16.10. Reverted Domain name tree**

Reverted domain name tree

All you have to do now is to change your domain tree node for your devices. Select your desired devices in the device tree sidebar on left side and navigate to **Base** **Device tree**. Push the **modify selected** button, select the **DTN** checkbox and choose your reverted domain tree node from the list.

**Figure 16.11. Modify devices**

Information

Change settings of 5 devices

Basic settings

☐ DeviceGroup

Additional settings

☒ DTN

Domain tree node\*

Select the domain tree node (for FQDN)

[TLN]  
one  
two.one  
three.two.one  
production.three.two.one

☐ Bootserver

☐ MonitorServer

Security

☐ pwd

Flags


☐ EnabledFlag

☐ PerfDataFlag

☐ store  
RRD  
data

☐ IPMI  
capable

Modify many

Modify devices to use your reverted  DTN

The result should looks like this:

**Figure 16.12. Reverted device domain names**

- ☒ test\_01.production.three.two.one (new device)
- ☒ test\_02.production.three.two.one (new device)
- ☒ test\_03.production.three.two.one (new device)
- ☒ test\_04.production.three.two.one (new device)
- ☒ test\_05.production.three.two.one (new device)

---

# Glossary

Django	Django is a free and open source web application framework, written in Python
DTN	Domain Tree Node, is the tree structure of fully qualified domain names.
FQDN	Fully qualified domain name, for example <b>noctua.init.at</b>
GNU GPL	Free software license named GNU General Public License, more Information at [ <a href="http://www.gnu.org/licenses/">http://www.gnu.org/licenses/</a> ]
icinga	Industry standard software for monitoring devices.
IPMI	Intelligent Platform Managemnt Interface
Network	A computer network or data network is a telecommunications network which allows computers to exchange data.
network topology central node (peer)	Is the central node other devices are connected to
nginx	Small and fast http server similar to apache
open source software	Often titled as OSS, is software with available source code. Popular open source software license is the GPL
Parameterized check	Some option values of this check (command) can be accessed by parameter.
PXE	Preboot Execution Environment
rrd-tool	RRDtool is the OpenSource industry standard, high performance data logging and graphing system for time series data.
SGE	The "Son of Grid Engine", community project of Sun Grid Engine. [ <a href="https://arc.liv.ac.uk/trac/SGE">https://arc.liv.ac.uk/trac/SGE</a> ]
TLN	Top Level Node, is the top level domain the node belongs to.
VM	Virtual Machine, general name for virtual systems running completely in an host environment like KVM or similar.

