TWO IN ONE

Administrator Handbook

The Somewhat Definitive Guide

DI Dr. Andreas Lang-Nevyjel <lang-nevyjel@init.at>
Georg Bahlon <bahlon@init.at>

Administrator Handbook: The Somewhat Definitive Guide

by DI Dr. Andreas Lang-Nevyjel and Georg Bahlon

Release: 1.5 -This Document was generated 2014-12-05

Abstract

This document is the official documentation for **CORVUS®** and **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.

1. About 1
1.1. Not the same but part of it 1
1.2. What is **CORVUS®** and what is it for? 1
1.3. What is **NOCTUA®** and what is it for? 1
1.4. Much more than just a webfrontend! 1
1.5. System Requirements 4
1.5.1. Common requirements 4
1.5.2. Database requirements 4
2. Installation 5
2.1. Basics 5
2.1.1. Operating Systems 5
2.1.2. Repository 5
2.1.3. Versions 5
2.1.4. Install repository 6
2.1.5. Debian repositories 6
2.1.6. Ubuntu repositories 7
2.1.7. OpenSUSE and SLES repositories 8
2.1.8. CentOS repositories 9
2.1.9. **CORVUS®** Packages 9
2.1.10. Database setup 10
2.1.11. Required services 10
2.1.12. Portnumber for accessing the webfrontend 11
2.1.13. Needed Server processes for nodeboot 11
2.2. Installation on virtual machine 11
2.2.1. KVM libvirt/qemu 11
2.3. Upgrade **CORVUS®** / **NOCTUA®** 12
2.4. Node Setup 12
2.5. Functionality 12
2.6. Configurations 12
3. Webfrontend 13
3.1. First connection 13
3.2. Areas 13
3.2.1. Menu area (1) 14
3.2.2. Tree area (2) 15
3.2.3. Main area (3) 17
3.2.4. Cluster server information 17
4. User and group management 19
4.1. Create user or group 19
4.2. Create group form 19
4.3. Create user form 20
4.4. Permission System 20
4.4.1. Permission 20
4.4.2. Permission level 23
5. Package installation 25
5.1. Preparing installation for package install 25
5.1.1. Server settings 25
5.1.2. Client settings 25
5.1.3. Server config file /etc/sysconfig/package-server 25
5.1.4. Client config file /etc/sysconfig/package-client 26
5.2. Install packages 27
5.2.1. Install packages using package manager 27
5.2.2. Install packages using directory upload 27
5.3. Delete packages 28
6. RMS - Resource Management System 29
6.1. Introduction of RMS 29
6.1.1. Environment variables 29
6.1.2. Installation of RMS 30
6.1.3. Basic cluster configuration 30

6.1.4. RMS web front-end 31
6.2. Job management system in SGE 32
6.2.1. SGE commands 33
6.2.2. Job submission via command line 35
6.2.3. Job submission via web front-end 35
7. Virtual Desktop 37
7.1. Prerequisites 37
7.2. Connect to virtual desktop 38
7.3. Change settings 39
8. SNMP discovery 41
8.1. Automatically network discovering with SNMP 41
8.1.1. Setup of SNMP discovery 42
8.1.2. Auto discover network device 42
9. Device localisations 45
9.1. Setup localisation 45
9.1.1. Upload and edit user images 46
9.1.2. Edit uploaded photos 48
9.2. Livestatus integration in maps 49
9.2.1. Binding livestatus burst to maps 49
9.2.2. Display livestatus burst 49
10. Monitoring 51
10.1. First simple check setup 51
10.2. Extended monitoring setup 51
10.3. Evaluation of monitored data 52
10.4. Overview Configuring Nodes 52
10.5. Foo 52
10.6. Monitoring 52
10.7. Discovery server 53
11. Parameterizing checks 55
11.1. Types of checks 56
11.1.1. Fixed 56
11.1.2. Parameterized 56
11.2. Examples 56
11.2.1. Different port 56
11.2.2. Different warning value 56
11.3. Advantages of parameterizing 56
12. Graphing 59
12.1. Introduction to Graphing 59
12.1.1. Principles of RRD 60
12.1.2. Data collection and graphing 60
12.1.3. How to display RRD graphs? 60
12.1.4. RRD frontend 61
12.1.5. RRD tree components 65
12.1.6. Summarized graphs 66
12.1.7. Compound graphs 67
13. Debugging and Error hunting 69
13.1. General information 69
13.2. Show errors 69
13.3. Node information 70
13.4. Logging 70
13.4.1. Automatic log mail delivering system 70
14. Frequently Asked Questions 71
14.1. Miscellaneous questions 71
14.1.1. Bad looking font in RRD Graph 71
14.1.2. Server Error (500) 71
14.1.3. Unable to connect 71
14.1.4. An error occurred 71
14.1.5. My changed config don't be applied. 72

14.1.6. Discovery not defined in routing 72
14.1.7. Slow Network topology graph 72
14.1.8. Lost login password 72
14.1.9. "Please wait..." after add location gfx 73
14.1.10. Weird mouse events on virtual desktop 74
Glossary 75

Chapter 1. About

1.1. Not the same but part of it

Although you hold only one Documentation in your hands or read it on screen, **CORVUS®** and **NOCTUA®** is not exact the same software. In fact **NOCTUA®** is part of **CORVUS®**. But why only one documentation for two different peace of software? Because of many overlapped parts and functions it makes no sense to write two documentations. Instead of two nearly the same documentations repeating many of the content we decided to provide only one documentation with clear visible content.

1.2. What is CORVUS® and what is it for?

Purpose of **CORVUS®** is the HPC Cluster Management. It allows administrators to setup and manage a huge number of nodes in clusters or even more than one cluster at once.

Especially in cooperation with **NOCTUA®** it offers both, management and monitoring of your cluster, two essential parts every admin of HPC sooner or later have to think about.

1.3. 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 *free Software*, licensed under the *GNU GPL* 2.0 License.

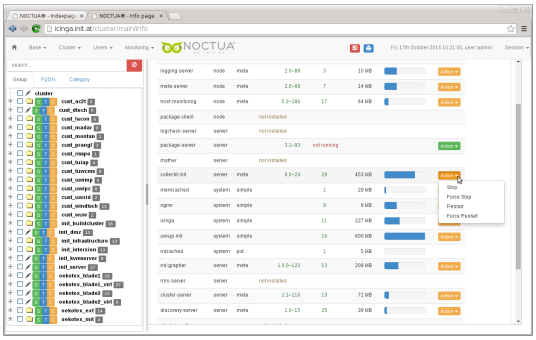
You can find more Information about free software and it's benefits at http://www.fsf.org

1.4. Much more than just a webfrontend!

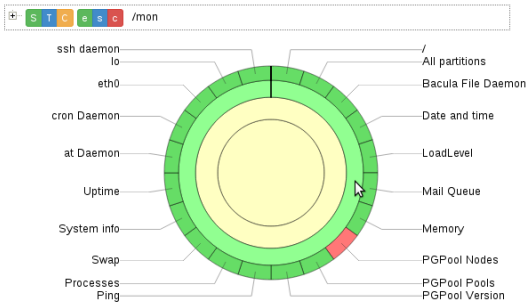
Even if the main task of **CORVUS®** and **NOCTUA®** is to ease configuration and administration of icinga, there are some special and unique features other software solutions do not have implemented.

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.

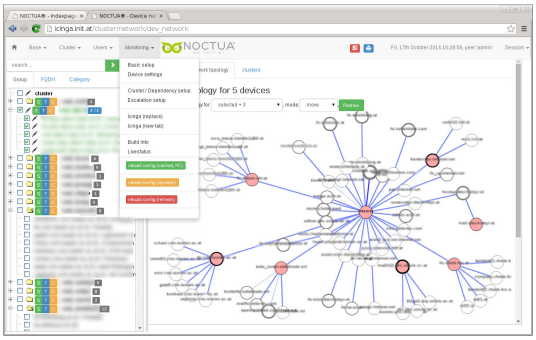


Livestatus - Graphical realtime indication tool to display monitored data of devices or cluster. No more need to interpret date.



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

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



Weather map [WIP] - Fancy graphical realtime image map to simply show data flow of devices.

no illustration available

Localisation [WIP] - Integration of collected data into google Maps™ . Facility to see cluster/ device status at a glance.

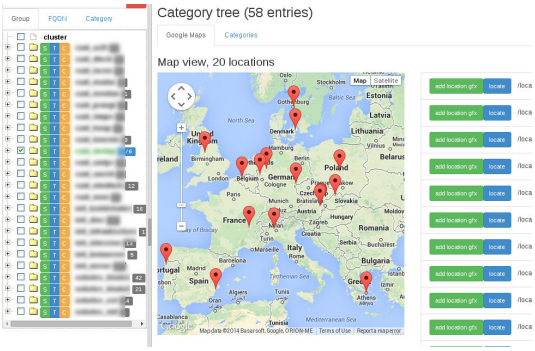
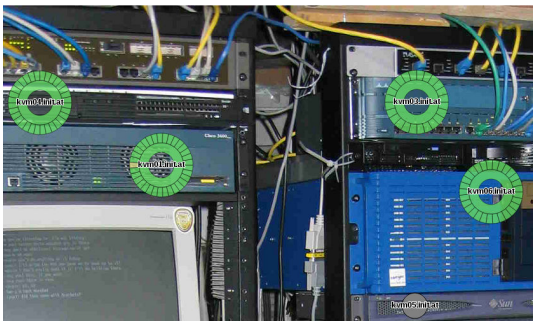
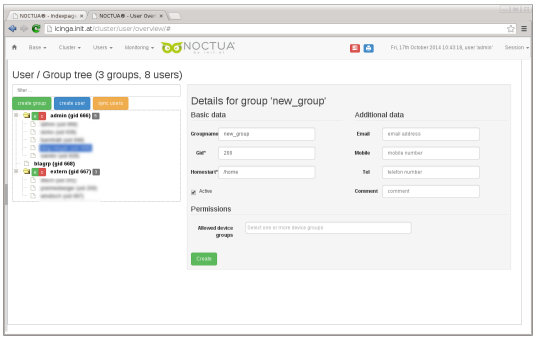


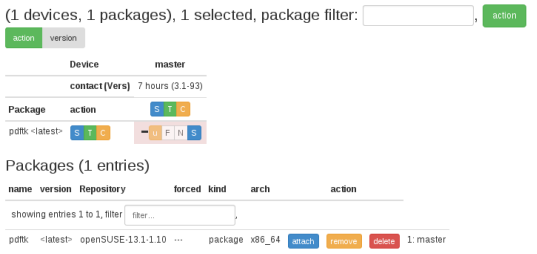
Image Maps [WIP] - Option to upload user images or photos from cluster infrastructure, plant layout or server racks. Status data of cluster and devices can be displayed as overlay on image.



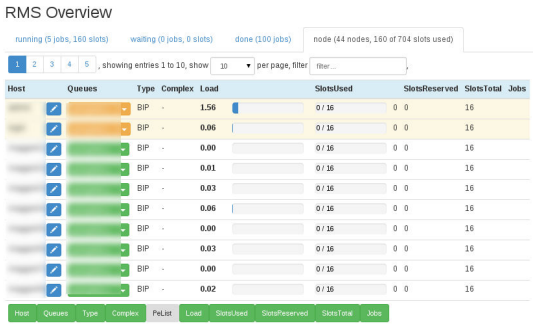
Usermanagement - Administrator tool to manage groups, users and corresponding permissions.



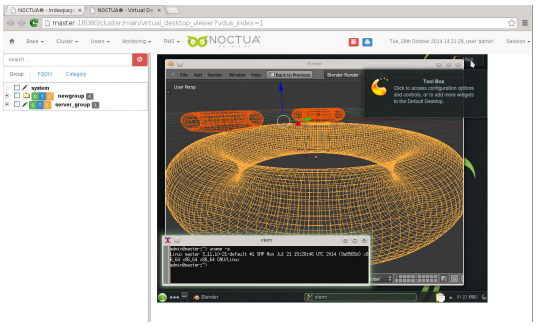
Packageinstall - Option to install system packages over the web front-end. (Operating system package management will take care for the exact procedure of installation.)



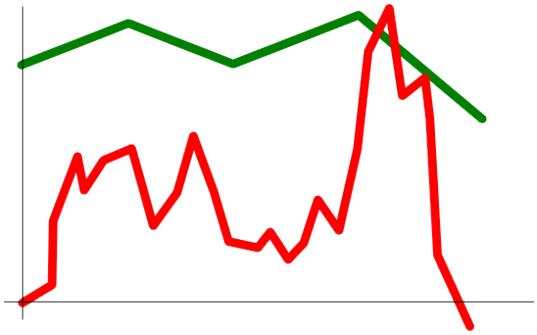
RMS - Resource Management System - Job management system for user to easy handle their jobs via the web front-end. Also



Virtual desktops - Automatic virtual desktop management system. Full access to remote machines displayed inside of your favorite browser.



License Management - License Management System to keep track of used, unused or locked licenses. With graphical output and history function.



1.5. System Requirements

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

1.5.1. Common requirements

Like every other software, **CORVUS®** and **NOCTUA®** has also certain system requirements. Because **CORVUS®** and **NOCTUA®** is free Software, everybody who has enough programming knowledge could be able to port it to other free 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 Chapter 2, *Installation*

1.5.2. Database requirements

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

CORVUS® and **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), another well tested one would be MySQL.

Chapter 2. Installation

2.1. Basics

2.1.1. Operating Systems

CORVUS® 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 at all.

If you are running one of the supported operating systems add the repository matching your installation to your package management software. Refer to the manual of your package management software how to do that.

2.1.2. Repository

There are 2 main and 1 extra repository you have to deal with to install either **CORVUS®** or **NOCTUA®**.

cluster	Main repository for **CORVUS®**. It includes among others `mother`, `cluster-server`, `package-server`, `package-client`, `discovery-server`, `collectd-init`
monit	This is the main repository for **NOCTUA®**. It includes among others `noctua`, `init-monit`
extra	This is the extended repository which contains packages **CORVUS®** and **NOCTUA®** are based on like `python-modules`, `icinga`, `nginx`, `uwsgi`

2.1.3. Versions

Repositories are available for oldstable (1.x), stable (2.x) and for master (devel) versions of above mentioned operating systems. The system running on your machines and the version of **CORVUS®** or **NOCTUA®** you want, determine which repository you must use in your package manager.

devel or **master**	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.

2.x or **stable**	The current stable version for productive environment. Most features and functions are included and there are no knowing bugs.
1.x or **oldstable**	This is the oldest available software version. No new features or functions are included and updates or changes will be done only for security issues.

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

2.1.4. Install repository

There are two different ways to add new repositories for monitoring software by **init.at™** 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 2.1. For debian based systems

Debian wheezy	`/etc/apt/sources.list.d/`
Debian squeeze	`/etc/apt/sources.list.d/`
Ubuntu 12.04	`/etc/apt/sources.list.d/`

Table 2.2. For suse based systems

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

Table 2.3. For red-hat based systems

CentOS	`/etc/yum.repos.d/`

2.1.5. Debian repositories

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

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`

devel (master)

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

```
deb http://www.initat.org/cluster/DEBs/debian_wheezy/cluster-devel wheezy main
deb http://www.initat.org/cluster/DEBs/debian_wheezy/monit-devel wheezy main
deb http://www.initat.org/cluster/DEBs/debian_wheezy/extra wheezy main
```

stable (2.x)

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

```
deb http://www.initat.org/cluster/DEBs/debian_wheezy/cluster-2.0 wheezy main
```

```
deb http://www.initat.org/cluster/DEBs/debian_wheezy/monit-2.0 wheezy main
deb http://www.initat.org/cluster/DEBs/debian_wheezy/extra wheezy main
```

oldstable (1.x)

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

```
deb http://www.initat.org/cluster/DEBs/debian_wheezy/cluster-1.0 wheezy main
deb http://www.initat.org/cluster/DEBs/debian_wheezy/monit-1.0 wheezy main
deb http://www.initat.org/cluster/DEBs/debian_wheezy/extra wheezy main
```

Of course, all above is true for debian squeeze, just replace wheezy with squeeze and you are done.

Repository directory

The second way to add repositories to the system is to put `*.list` files into your repository directory.

Name of the directory in debin is`/etc/apt/sources.list.d/`

With **wget** you are able to download the `*.list` file into your repository directory. Go into `/etc/apt/sources.list.d/` and enter this command to download repository file.

```
wget    http://www.initat.org/cluster/repository_files/initat_debian_wheezy_devel.list
```

> **Important**
>
> Please pay attention to broken link line if you want to copy and paste the command line. make sure there is no ″-″character and only one coherent line.

2.1.6. 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

devel (master)

```
deb http://www.initat.org/cluster/DEBs/ubuntu_12.04/cluster-devel precise main
deb http://www.initat.org/cluster/DEBs/ubuntu_12.04/monit-devel precise main
deb http://www.initat.org/cluster/DEBs/ubuntu_12.04/extra precise main
```

stable (2.x)

```
deb http://www.initat.org/cluster/DEBs/ubuntu_12.04/cluster-2.0 precise main
deb http://www.initat.org/cluster/DEBs/ubuntu_12.04/monit-2.0 precise main
deb http://www.initat.org/cluster/DEBs/ubuntu_12.04/extra precise main
```

oldstable (1.x)

```
deb http://www.initat.org/cluster/DEBs/ubuntu_12.04/cluster-1.0 precise main
deb http://www.initat.org/cluster/DEBs/ubuntu_12.04/monit-1.0 precise main
deb http://www.initat.org/cluster/DEBs/ubuntu_12.04/extra precise main
```

Repository directory

Almost the same procedure like for debian must be done in ubuntu. The only difference is the link.

With **wget** you are able to download the `*.list` file into your repository directory. Go into `/etc/apt/sources.list.d/` and enter this command to download repository file.

```
wget    http://www.initat.org/cluster/repository_files/initat_ubuntu_1204_devel.list
```

2.1.7. 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 cluster devel version

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

SUSE 13.1 and monit devel version

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

SUSE 13.1 and extra packages

```
[init-extra]
name=init-extra
enabled=1
autorefresh=0
baseurl=http://www.initat.org/cluster/RPMs/suse_13.1/extra
type=rpm-md
```

You need, as mentioned before, at least 2 main and 1 extra repository. Either you create new files with your favorite editor or you can use **zypper** command.

Zypper command for adding repositories is:

```
zypper ar http://www.initat.org/cluster/RPMs/suse_13.1/cluster-devel cluster-devel
```

You can use the same pattern for other suse versions, only replace *suse_13.1* with your desired version, for example suse_12.3.

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/

2.1.8. CentOS 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 **CORVUS®** / **NOCTUA®**

CentOS 6.5 devel branch

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

[initat_extra]
autorefresh=1
enabled=1
type=rpm-md
name=initat_extra
baseurl=http://www.initat.org/cluster/RPMs/rhel_6.2/extra

[initat_monit]
autorefresh=1
enabled=1
type=rpm-md
name=initat_monit
baseurl=http://www.initat.org/cluster/RPMs/rhel_6.2/monit-devel
```

2.1.9. CORVUS® Packages

Before installing packages make sure to remove the following conflicting packages:

- nginx
- uwsgi
- memcached

Install the package containing the **at** daemon before attempting to install **CORVUS®** packages.

- cluster-backbone-sql
- webfrontend
- host-monitoring
- nginx-init
- uswgi-init
- cluster-server
- mother
- package-server
- cluster-config-server

from the newly added repositories. These packages pull in all the necessary dependencies to use **CORVUS®**. Ignore the output from the package post install scripts on how to populate the database. Setup the following processes to be started at your default runlevel:

Note

done by setup_noctua

- nginx
- uwsgi
- host-monitoring
- at

2.1.10. Database setup

Refer to the documentation of your Database System on how to create users and databases.

The database access data is stored in `/etc/sysconfig/cluster/db.cf` , a sample file is provided under `/etc/sysconfig/cluster/db.cf.sample` . If you want to connect via local socket leave DB_HOST empty. Fill in the user and database information.

Every daemon and process from **CORVUS®** 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
```

2.1.11. Required services

Nearly every aspect of **CORVUS®** and **NOCTUA®** is administrated via the webfrontend, so the next steps after the initial database setup are done there. The following system processes are needed to access the webfrontend:

- nginx (the web-server, started via /etc/init.d/nginx start)
- uwsgi (serves the application, started via /etc/init.d/uwsgi-init start)
- memcached (for storing session data, started via /etc/init.d/memcached start)

In order to check if these processes are running simply issue the command **/opt/cluster/sbin/check_scripts.py --system memcached nginx uwsgi-init** which should give an output similar to

```
Name       type   status
------------------------
memcached  system running
nginx      system running
uwsgi-init system running
```

2.1.12. Portnumber for accessing the webfrontend

The Webfrontend for **NOCTUA®** can be accessed via http://SERVERNAME/cluster or by http://IP_ADDRESS:**18080**/cluster/

The Webfrontend for **CORVUS®** can be accessed via http://SERVERNAME/cluster or by http://IP_ADDRESS:**8080**/cluster/

2.1.13. Needed Server processes for nodeboot

For booting nodes you have to have an *NFS* server up and running. The entries in your /etc/exports file will be added automatically by the cluster software.

2.2. 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 install a virtual machine with already installed **CORVUS®** or **NOCTUA®**. We distribute two popular 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.

2.2.1. KVM libvirt/qemu

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

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 and add or modify
4. Define your new virtual machine

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

2.3. Upgrade CORVUS® / NOCTUA®

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 banch 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 OpenSUSE
apt-get update; apt-get upgrade handbook-init	Refresh repositories and do single package upgrade in debian

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

2.4. Node Setup

The only thing you have to do is to set the nodes to boot from network(*PXE*).

2.5. Functionality

Go to /etc/sysconfig/cluster/cluster_license and set the required license entries to enabled="yes".

2.6. Configurations

Under Setup > Configurations Add the various configurations: cluster_server mother package_server

Under Setup > Device Settings > Configuration attach the configurations to the host server.

Chapter 3. Webfrontend

3.1. First connection

Most configuration in **CORVUS®** administrators have to do, will be accessed over a standard html compatible browser like Mozilla Firefox™ or Google Chrome™. Once **CORVUS®** 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:18080/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, **CORVUS®** takes his own during installation procedure generated password you never seen before and next time you can not log in.

If you running the

```
setup_noctua.sh
```

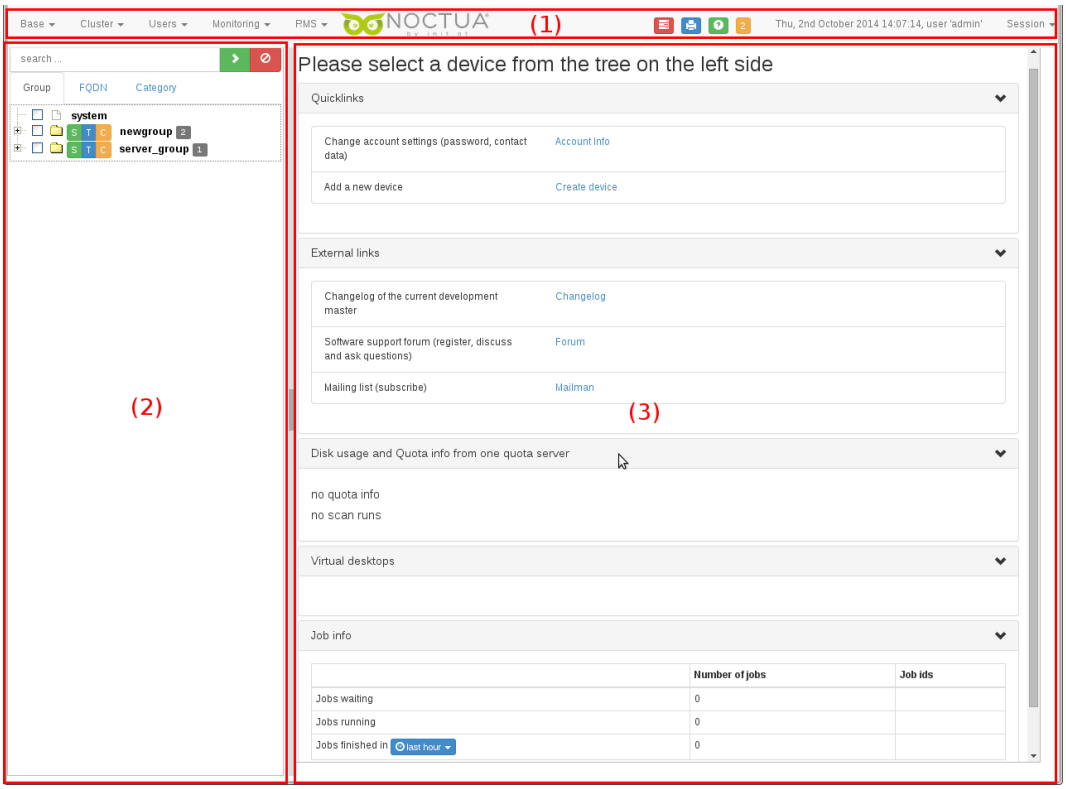
script manually, a new password will be generated. In this case you must look for the password in your shell output.

3.2. Areas

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

- Menu area (1)
- Tree area (2)
- Main area (3)

Figure 3.1. Three areas



Areas you'll see after login

3.2.1. 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

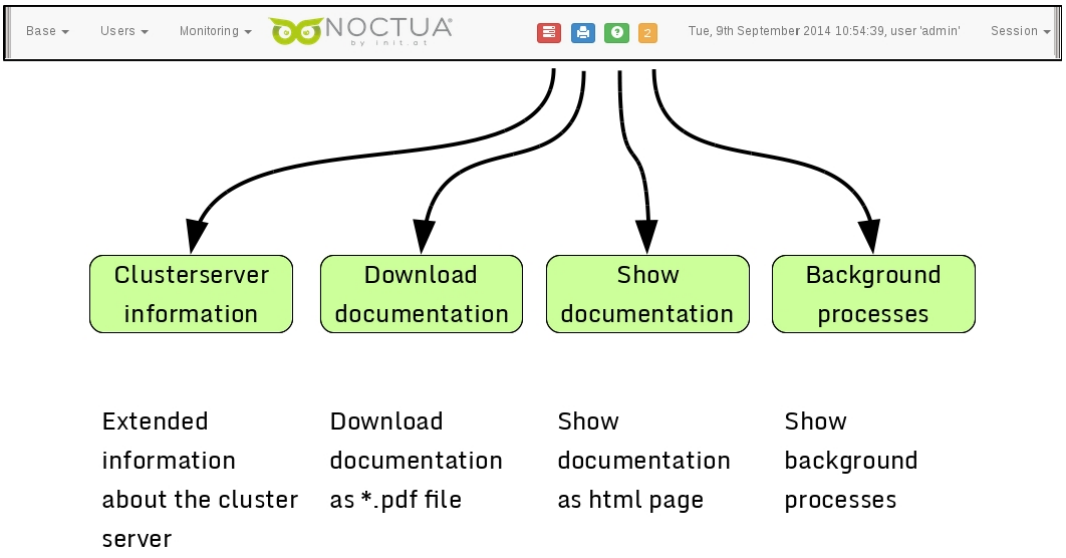
CORVUS® offers some additional menus:

1. RMS - Resource management System
2. Cluster

Buttons

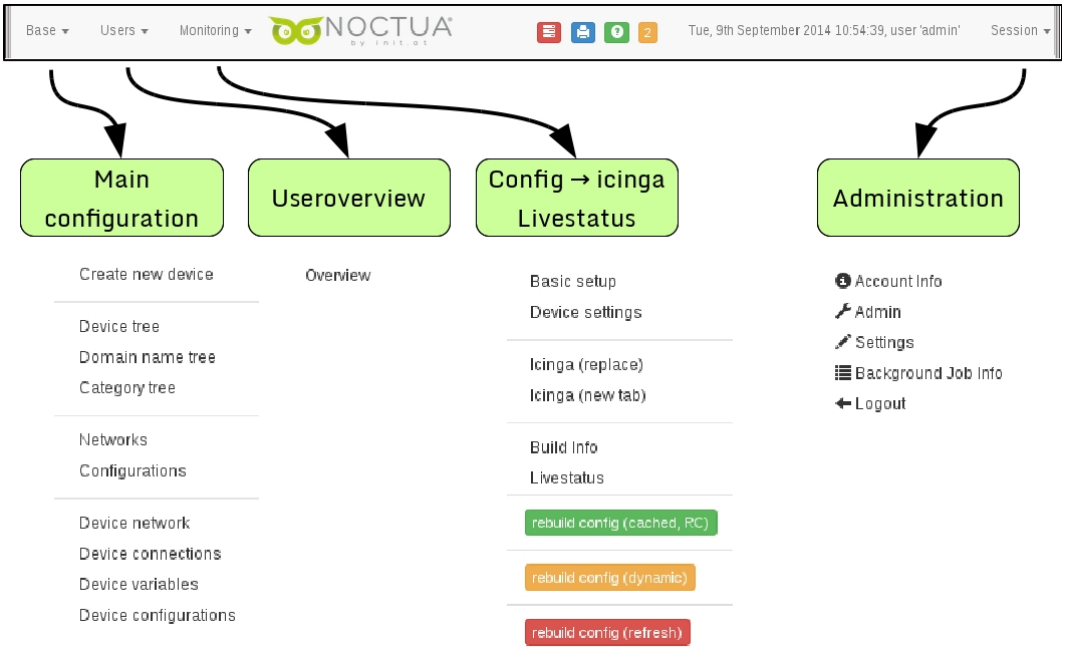
1. cluster server information
2. show cluster handbook as pdf
3. show index
4. number of background jobs

Figure 3.2. Menu buttons



Buttons and submenus for **NOCTUA®**

Figure 3.3. Menus



Menus for **NOCTUA®**

3.2.2. Tree area (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.

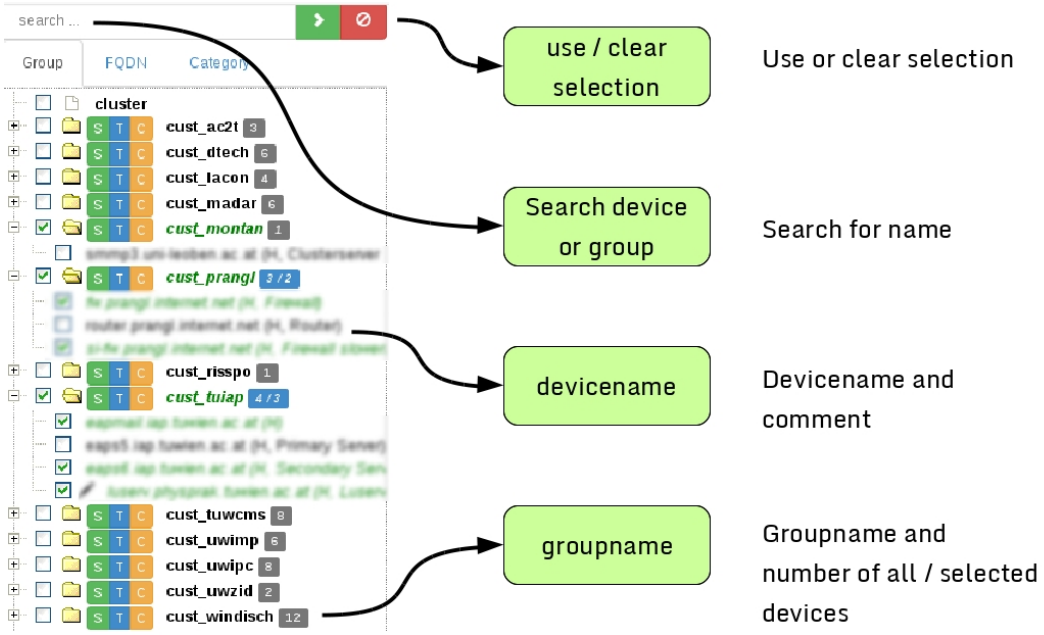
1. Searchfield
2. use selection Button (green with arrow)
3. clear selection Button (red with circle)

4. Group

5. FQDN (Full Qualified Domain Name)

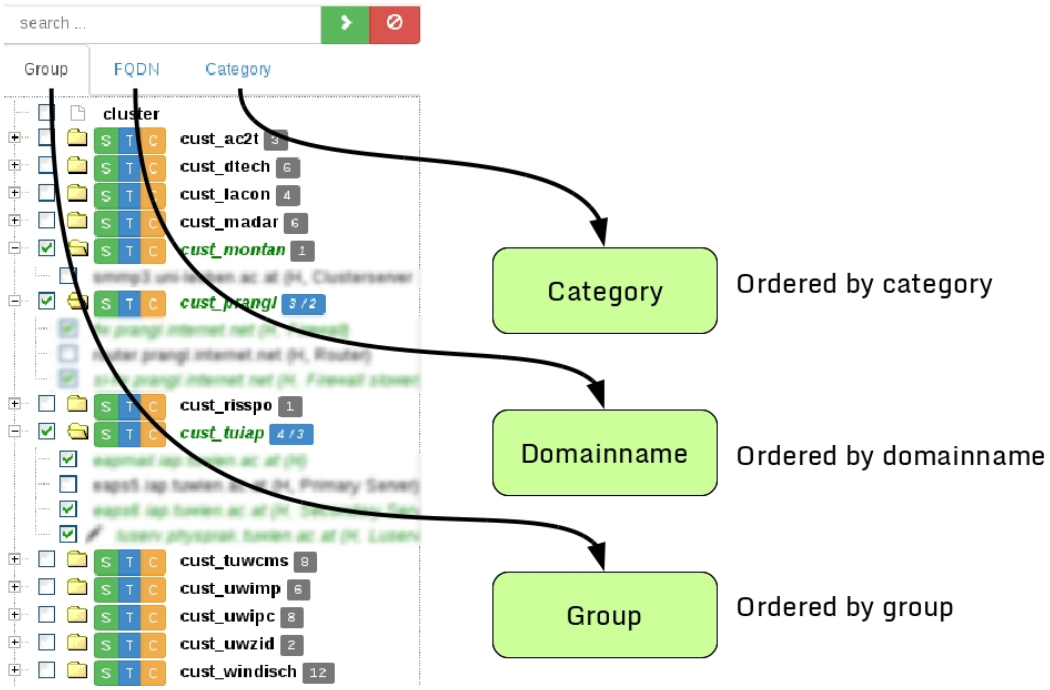
6. Category

Figure 3.4. Devicetree



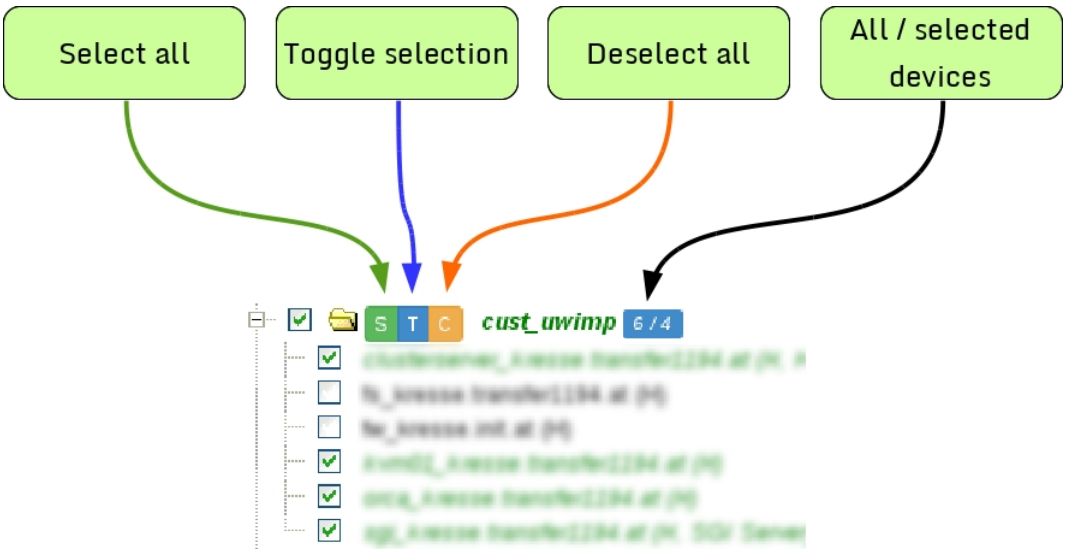
NOCTUA®

Figure 3.5. Devicetree



NOCTUA®

Figure 3.6. Selection buttons



Buttons to select, deselect or toggle selection

3.2.3. 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 3.7. Possible main area

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

3.2.4. Cluster server information

The cluster server information button shows two overview tabs, one tab with information about **definied cluster roles** and one with information about **server**

Cluster roles definied

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

It is a matter of services providing special functionality to the server.

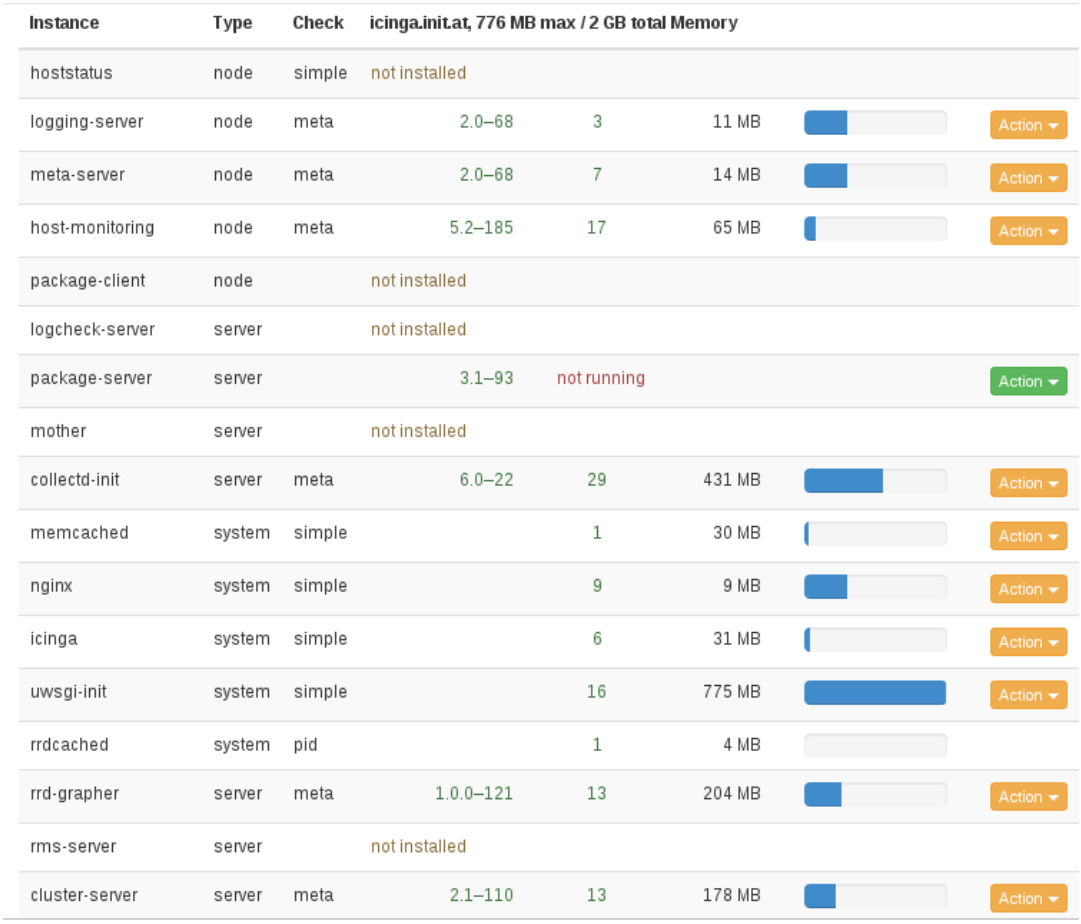
One Server checked

Inside this tab, there is a table showing following information:

Server information

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

Figure 3.8. Cluster server information



Backgroundinformation about running or stopped services

Chapter 4. User and group management

4.1. Create user or group

After installation of **CORVUS®** or **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:

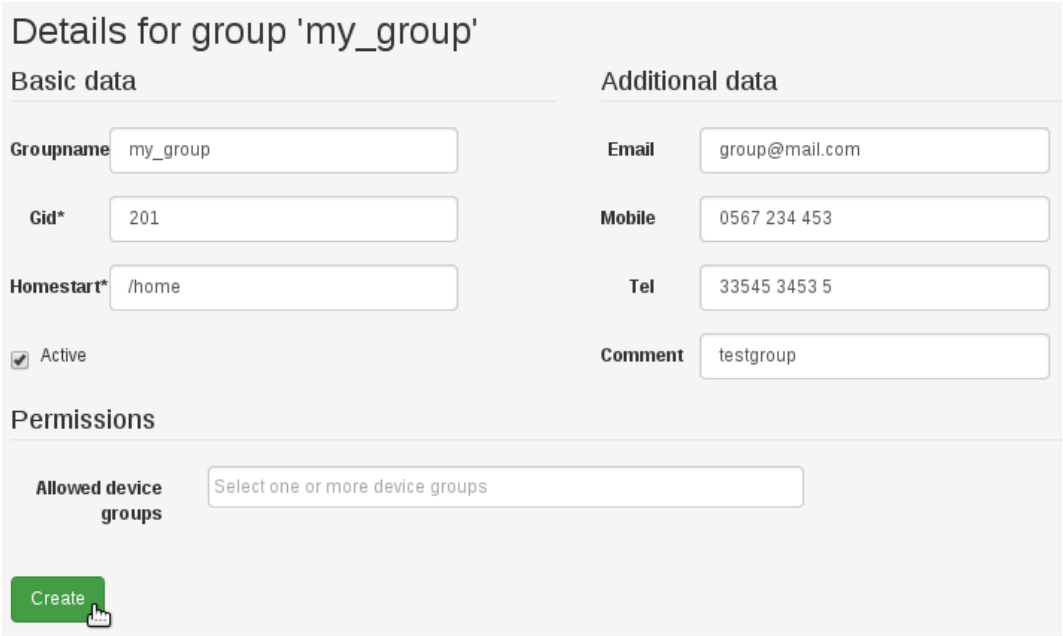
Figure 4.1. Userbuttons

Create user, group or sync users Button

4.2. 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 4.2. Group create form



Basic settings form to create group

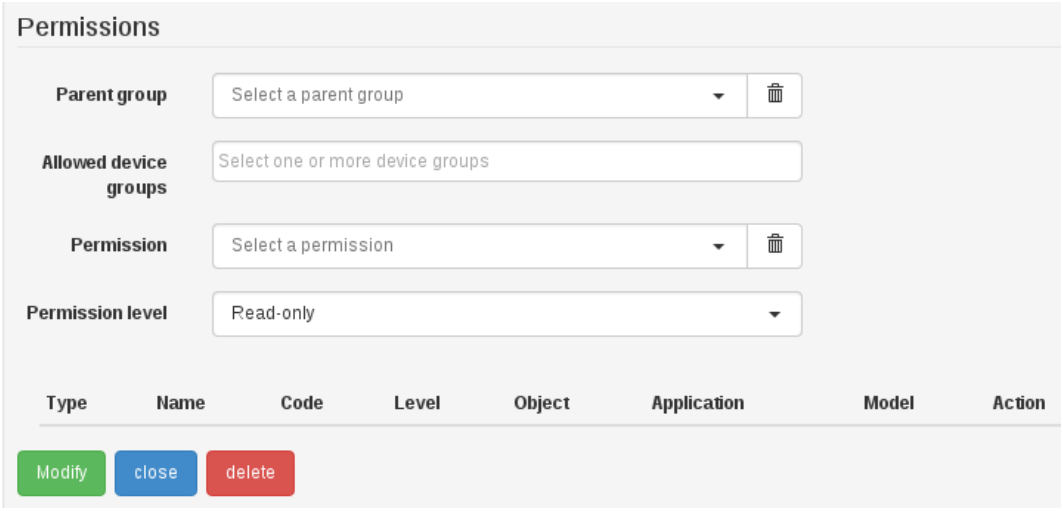
The form is self-explanatory, but some input should be mentioned anyway:

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 4.3. Extended permissions



More complex permission settings

4.3. 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

4.4. Permission System

4.4.1. 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: **Session**  **Background Job Info**

config

modify global configurations (G)	Shows additional menu button: **Base**  **Configurations**

device

Access to device graphs (G/O)	Shows **graphs** tab for selected devices. Depends on possibility to choose devices (**acess all devices**)
change disk setup (G/O)	Shows disk tab for selected devices. Depends on possibility to choose devices (**acess all devices**)
Change basic settings (G/O)	Change basic settings (General) for selected devices. Depends on possibility to choose devices (**acess all devices**)
Change boot settings (G/O)	Shows new top-menu named **Cluster**
Change configuration (G/O)	Show **Config** tab for selected devices. Depends on possibility to choose devices (**acess all devices**)
Change device category (G/O)	Show **Category** tab for selected devices. Depends on possibility to choose devices (**acess all devices**)
Change device connection (G/O)	Shows new top-menu: **Base**  **Device connections**
Change device location (G/O)	Show **Location** tab for selected devices. Depends on possibility to choose devices (**acess all devices**)
Change device monitoring config (G/O)	Shows 3 new tabs for selected devices: • Livestatus • Monconfig • MonHint
Change network (G/O)	Shows new top menu content: **Base**  **device network** Depends on possibility to choose devices (**acess all devices**)
Change variables (G/O)	Show **vars** tab for selected devices and new top menu: **Base**  **Device variables**. Depends on possibility to choose devices (**acess all devices**)
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.

group

Group administrator (G/O)	...

image

Modify images (G)	...

kernel

Modify kernels (G)	...

mon_check_command

Change monitor settings (G)	Shows new top menu content under: **Monitoring**  **Basic Setup / Build Info**

network

modify global network settings (G)	...
show network clustering (G)	...

package

access package install site (G)	Shows new top menu under: **Cluster**  **Package install**. Additional software packages can be choosen and installed by this menu button.

partition_fs

modify partitions (G)	...

user

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

4.4.2. 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.

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

Chapter 5. Package installation

Installation of packages via the webfrontend is another helpful feature provided by **CORVUS®** and **NOCTUA®**. It offers you to install software packages on one or many systems over the webfrontend, without needs to login on each local machine and install packages manually with the command line.

Your **CORVUS®** or **NOCTUA®** operates as central package installation entity, stores its repositories in the database and can also distribute its repositories to connected nodes.

It's a huge ease for user with less experience to do software installation with a few clicks instead of typing long and cryptic terminal commands.

In this section you can learn how to setup this feature, how to configure and how to use it.

5.1. Preparing installation for package install

Two important services for this function are: **package-client, package-server**

> First of all you have to install both, **package-client** on the client machine and **package-server** on your server machine

Before you are able to install packages by the webfrontend, you have to configure your machines appropriate. Not only the server-side configuration but also the client-side configuration is essential to make installation and distribution of packages working.

5.1.1. Server settings

1. On top menu, go to **Session**  **Settings**. Enable the Button for **package installation** (package) and reload the page.

2. Click your server device from device-tree on the left side, go into "Config" tab, click on the blue arrow button and activate "package_server" on dropdown menu.

3. Start the package-server by navigating to **cluster server information** and open the lower dropdown menu with click on the arrow. Push "Action" button for **package-server** and choose start.

So far, your server is ready for package installation. Also the clients/nodes have to be prepared for package installation.

5.1.2. Client settings

1. Make sure **package-client** service is installed and running on the nodes/clients. To check the status of package-client use **check_cluster** command. Status of package-client should be "running".

2. Last step to setup package-installation is to enter your server (package-server) IP-address (or hostname) in `/etc/packagserver` on the client machine.

5.1.3. Server config file /etc/sysconfig/package-server

The main configuration file for package-server is `/etc/sysconfig/package-server`. It content should be self-explanatory and looks like this:

Table 5.1. package-server config options

options	default value	description
PID_NAME=	package-client/package-client	Name of PID files
KILL_RUNNING=	True	...
USER=	idpacks	Username
GROUP=	idg	Groupname
GROUPS=	['idg']	$$
LOG_DESTINATION=	uds:/var/lib/logging-server/py_log_zmq	Destination of log files
LOG_NAME=	package-server	Name of log file
SERVER_PUB_PORT=	8007	Server port for communication with client
NODE_PORT=	2003	Client port for communication with server
DELETE_MISSING_REPOS=	False	Capability to deleting missing repos

5.1.4. Client config file /etc/sysconfig/package-client

The main configuration file for package-client is `/etc/sysconfig/package-client`. It content should be self-explanatory and looks like this:

Table 5.2. package-client config options

options	default value	description
PID_NAME=	package-client/package-client	Name of PID files
KILL_RUNNING=	True	...
COM_PORT=	2003	Client port for communication with server
SERVER_COM_PORT=	8007	Server port for communication with client
LOG_DESTINATION=	uds:/var/lib/logging-server/py_log_zmq	Destination of log files
LOG_NAME=	package-client	Name of log file
NICE_LEVEL=	15	Nice level the log daemon running at
MODIFY_REPOS=	False	Capability to modify repositories
PACKAGE_SERVER_FILE=	/etc/packageserver	$$
PACKAGE_SERVER_ID_FILE=	/etc/packageserver_id	$$

Important

Set "MODIFY_REPOS=False" to forbid repository modification.

5.2. Install packages

There are two common ways to install additional packages.

- Package installation with operating system package manager
- Package installation with package upload in directory

Usually the first method is recommended for standard installation of available packages. All software and packages your running system provides, can be installed via "Package install". It starts your system package-manager in background (apt-get, yum, zypper) and install selected packages on selected nodes.

5.2.1. Install packages using package manager

1. In top menu, go to **Cluster**  **Package install**.
2. Push the **Rescan** button to update you repositories.
3. Go to **Package search** tab and search for the packages you want to install on the system.
4. If there are some results, list all matching packages with the **show** button. In below appeared list choose your desired package version by pushing one of the the right buttons (take exact/take latest).
5. Go to **Install** tab, select devices the package should be installed for and push "attach" button.
6. On top, a new button "action" appears. Push the button, choose "Target state" install and submit your settings. The package will be installed automatic on your selected nodes.

5.2.2. Install packages using directory upload

If your system do not provide some packages you really want to install, there is an other way to go. In this special case you can either download fitting binary packages from external sources and place it in the right directory or you can compile and build your own package from sourcecode.

Upload binary packages

1. Upload your package into your upload directory on your server: `/opt/cluster/system/packages/`
2. Execute the update script `update_repo.sh` in `/opt/cluster/system/packages/` to refresh your repositories.

   This script does:

```
#!/bin/bash
cd /opt/cluster/system/packages
createrepo .
yum clean all
yum makecache
```

3. Maybe you have to "Sync to clients"/"Clear caches" to get the new repositories on all nodes.
4. Now, if you search after uploaded package you should get some results. To install uploaded package follow the same procedure as install packages from system package manager.

Compile, make and upload packages from source

1. Download source files and extract it.
2. Compile your software as usual and install it (.configure ; make; make install).
3. Once your package is installed, use **make_package.py** to create a new *.rpm package.
4. Run the **update_repo.sh** to refresh your repositories.
5. Maybe you have to "Sync to clients"/"Clear caches" to get the new repositories on all nodes.
6. On top, a new button "action" appears. Push the button, choose "Target state" install and submit your settings. The package will be installed automatic on your selected nodes.

5.3. Delete packages

To delete packages do following steps:

1. In top menu navigate to **Cluster**  **Package install**, and choose the **Install** tab.
2. Select packages and nodes to delete it from.
3. Push the **Action** button and choose **erase** from **Target state** dropdown menu. To finish deletion click on the **Submit** button.

Chapter 6. RMS - Resource Management System

An essential aspect in **CORVUS®** is the job management system. Main reason for using clusters is a higher computing power to calculate jobs. The calculation of data will be splitted into pieces and every node or slot can calculate each piece separately, this results in a higher speed of calculation. The organisation of slots, cluster and jobdistribution is done by the *SGE - son of grid engine*. SGE provides special commands and tools to control jobs distributed to the nodes.

The RMS is the coupling between the SGE and our web front-end. With enabled RMS you are able to manage jobs without any using of SGE commands.

6.1. Introduction of RMS

Like mentioned before, the RMS is a powerful addon for managing jobs on clusters. It consists of packages and services working together to provide management functions for transmitted jobs.

Important parts of RMS are:

SGE part

- SGE - Son of Grid Engine
- Commandline tools like:
  - qdel
  - qstat
  - qacct

  Look command-reference or manual page of `sge_intro` to show complete list of commands.

  man sge_intro

init.at part

- RMS-server.py - Server between SGE and Webfrontend
- Webfrontend
- Commandline tools like:
  - sjs
  - sns

Both commands sjs and sns are links to `/opt/cluster/bin/sgestat.py`.

6.1.1. Environment variables

Environment variables for setting up RMS can be found under `/etc/`

- `/etc/sge_cell`

  Name of SGE

- `/etc/sge_server`

  Hostname or IP address of sge server.

- `/etc/sge_root`

  Directory sge installs to.

6.1.2. Installation of RMS

To get RMS working it is not enough only to install the package, you must also edit some config files and build the SGE part manually. Below step by step how to install RMS will help you installing RMS and run the required services.

Even it should be obvious, before you are able to install RMS make sure you already installed noctua and its dependencies.

1. Install rms-tools:

   zypper ref; zypper in rms-tools

2. Set environment variables in `/etc/sge_cell`, `/etc/sge_server` and `/etc/sge_`

   Setting of environment variables must be done **before** compiling SGE!

3. Download the latest version (Latest version for 2014.09.25 is 8.1.7) of SGE package from https://arc.liv.ac.uk/trac/SGE

   wget http://arc.liv.ac.uk/downloads/SGE/releases/8.1.7/sge-8.1.7.tar.gz

4. Extract `sge-8.1.7.tar.gz` archive to `/src/`, change into extracted directory and run our buildscript placed under `/opt/cluster/sge/build_sge6x.sh`.

   tar xzf sge-8.1.7.tar.gz

   cd /src/source/

   /opt/cluster/sge/build_sge6x.sh

   If your system can not compile and output some error messages, make sure you already installed necessary build-tools and development packages. Dependent of your operating system package names and count could differ.

5. Now directories under `/opt/sge62` exists and service **sge_qmaster** is running.

   Test if sge_qmaster is running:

   ps aux | grep sge_qmaster

6. Set $PATH variables by running script located under `/etc/profile.d/batchsys.sh`

   . /etc/profile.d/batchsys.sh

7. Run followed scripts:

   /opt/cluster/sge/create_sge_links.py and **/opt/cluster/sge/modify_sge_config.sh**

6.1.3. Basic cluster configuration

COMMING SOON ...$$

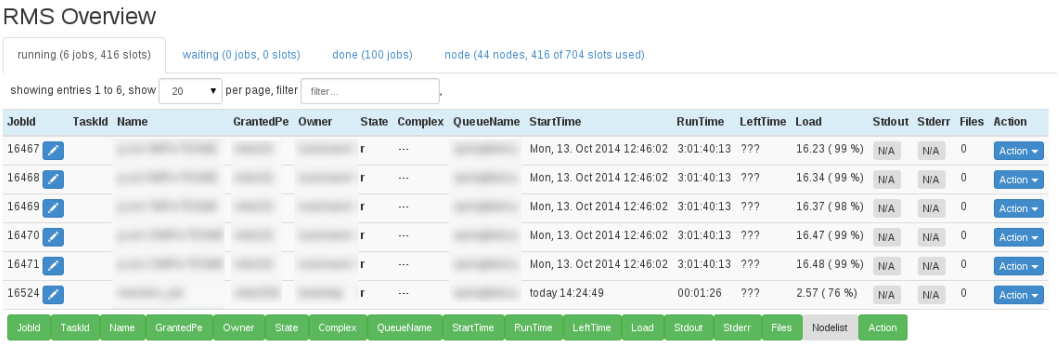
6.1.4. RMS web front-end

RMS overview provides 4 tabs. Not only for displaying information but also to control jobs. There are a couple of green buttons on the bottom of overview page to hide or unhide columns.

Running jobs

The first tab of RMS overview displays current **running** jobs in the grid engine. You can get some background informations like jobids, owner, runtime or nodelist of each job. On the right side there is an action button to delete or force delete running jobs.

Figure 6.1. RMS running jobs

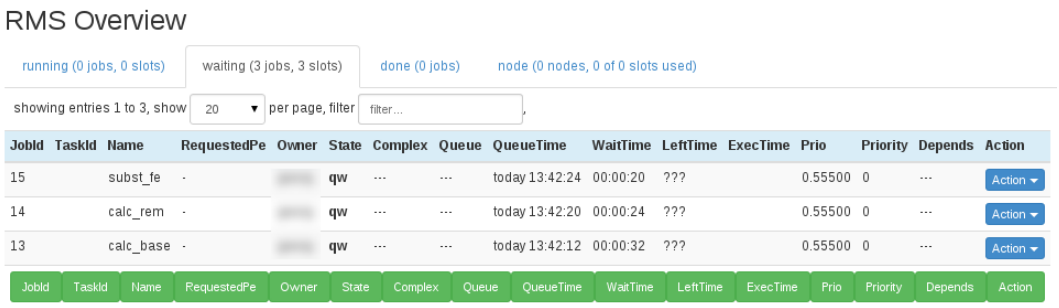


Current running jobs with disabled nodelist column

Waiting jobs

The second tab of RMS overview displays the current **waiting** jobs. This are jobs waiting in the SGE queue for execution. Among other infos, it shows the "WaitTime", "Depends" and the "Left-Time".

Figure 6.2. RMS waiting jobs

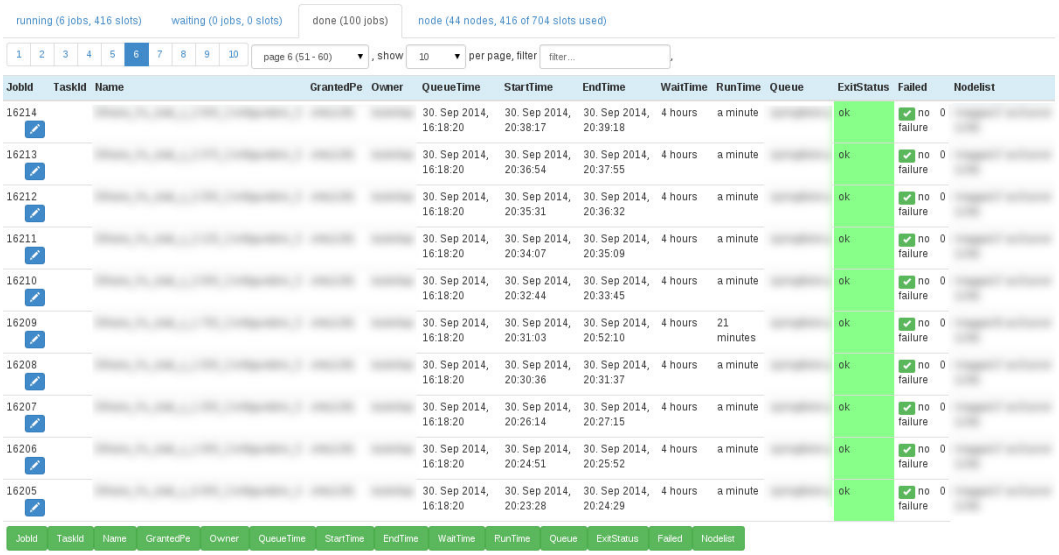


Current waiting jobs

Done jobs

The third tab of RMS overview displays **done** jobs and specific columns like "ExitStatus", "Failed" or "RunTime".

Figure 6.3. RMS done jobs

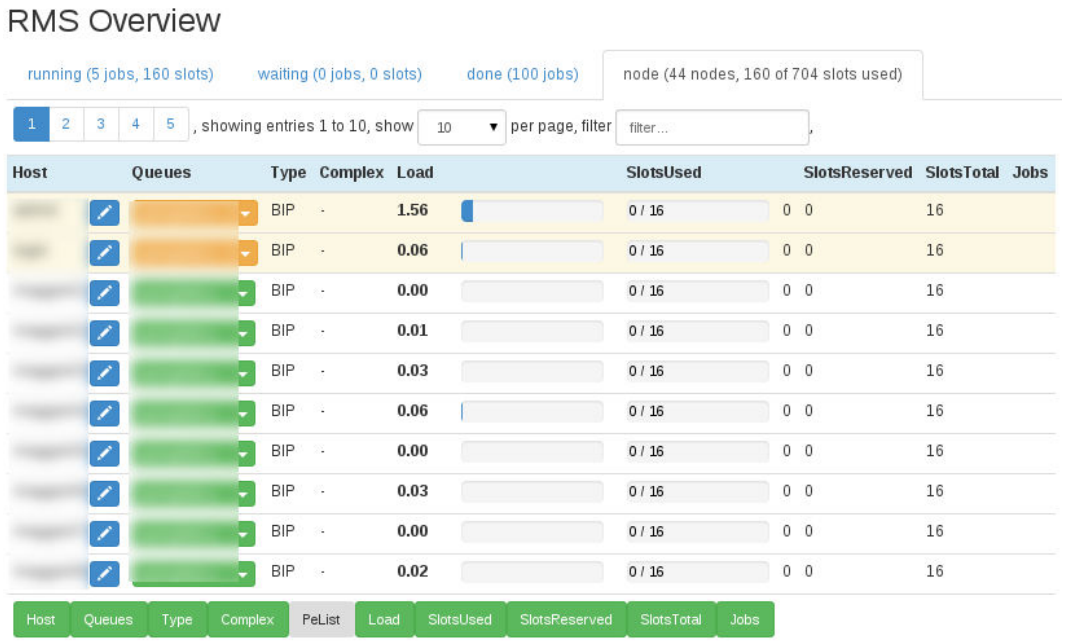


Done jobs

Nodes

The fourth tab of RMS overview displays the **nodes** itself. You can enable or disable queues or, if it exists, display graphs of choosen nodes.

Figure 6.4. RMS nodes



Node overwiew

6.2. Job management system in SGE

For direct usage of the SGE, there are a couple of commands:

6.2.1. SGE commands

Commands the SGE provides are:

qacct

qacct extracts arbitrary accounting information from the cluster logfile.

qalter

qalter changes the characteristics of already submitted jobs.

qconf

Queue Configuration, allows the system administrator to add, delete, and modify the current Grid Engine configuration, including queue management, host management, complex management and user management.

qdel

Provides a means for a user/operator/manager to delete one or more jobs.

qevent

qevent provides a means of watching Grid Engine events and acting on jobs finishing.

qhold

Qhold holds back submitted jobs from execution.

qhost

qhost displays status information about Grid Engine execution hosts.

qlogin

qlogin initiates a telnet or similar login session with automatic selection of a suitable host.

qmake

qmake is a replacement for the standard Unix make facility. It extends make with an ability to distribute independent make steps across a cluster of suitable machines.

qmod

qmod allows the owner(s) of a queue to suspend and enable queues, e.g. all queues associated with his machine (all currently active processes in this queue are also signaled) or to suspend and enable jobs executing in the queues.

qmon

qmon provides a Motif command interface to all Grid Engine functions. The status of all, or a private selection of, the configured queues is displayed on-line by changing colors at corresponding queue icons.

qping

qping can be used to check the status of Grid Engine daemons.

qquota

qquota provides a status listing of all currently used resource quotas (see sge_resource_quota(5)).

qresub

qresub creates new jobs by copying currently running or pending jobs.

qrls

qrls releases holds from jobs previously assigned to them e.g. via qhold(1) (see above).

qrdel

qrdel provides the means to cancel advance reservations.

qrsh

qrsh can be used for various purposes such as providing remote execution of interactive applications via Grid Engine comparable to the standard Unix facility rsh, to allow for the submission of batch jobs which, upon execu- tion, support terminal I/O (standard/error output and standard input) and terminal control, to provide a batch job submission client which remains active until the job has finished or to allow for the Grid Engine-controlled remote execution of the tasks of parallel jobs.

qrstat

qrstat provides a status listing of all advance reservations in the cluster.

qrsub

qrsub is the user interface for submitting an advance reservation to Grid Engine.

qselect

qselect prints a list of queue names corresponding to specified selection criteria. The output of qselect is usu- ally fed into other Grid Engine commands to apply actions on a selected set of queues.

qsh

qsh opens an interactive shell (in an xterm(1)) on a low loaded host. Any kind of interactive job can be run in this shell.

qstat

qstat provides a status listing of all jobs and queues associated with the cluster.

qtcsh

qtcsh is a fully compatible replacement for the widely known and used Unix C-Shell (csh) derivative tcsh. It pro- vides a command-shell with the extension of transparently distributing execution of designated applications to suitable and lightly loaded hosts via Grid Engine.

qsub

qsub is the user interface for submitting a job to Grid Engine.

> For more information please take a look into the well written man pages of each *"q"* command.

6.2.2. Job submission via command line

Common way to submit jobs to the cluster is to use grid engines *"q"* commands. Assumed that your cluster configuration is correct, running jobs on cluster is as easy as running jobs on local machines.

Following steps have to be done to transfer jobs to queue:

- test

TEST TEST2

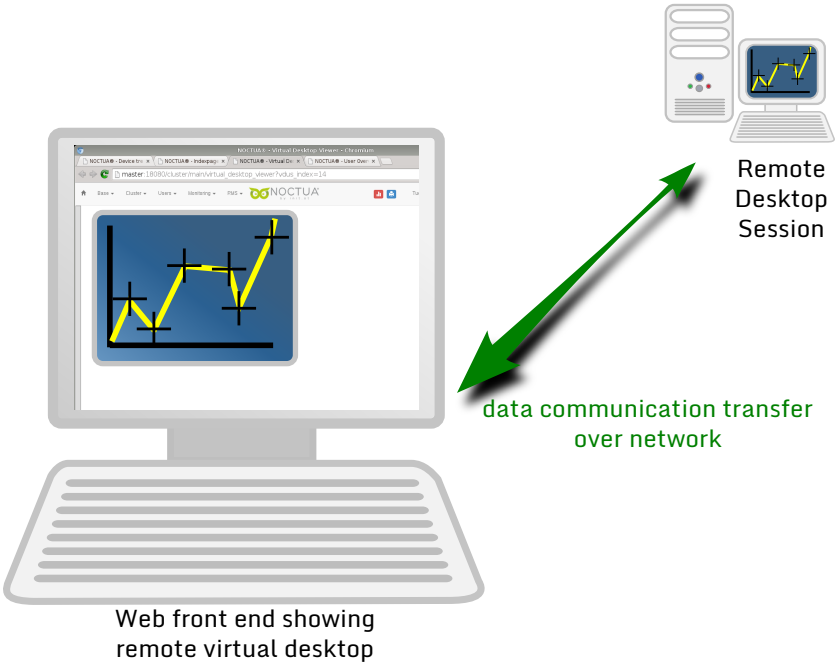
1. First...
2. Second...

6.2.3. Job submission via web front-end

Chapter 7. Virtual Desktop

Virtual desktop is a technology to transfer display output from remote graphic cards to your local machine graphic card.

Figure 7.1. VNC functionality



Basic illustration to explain vnc technology

User are often forced to work on remote machines because of computation power, license issues or simply geographical distance. In this cases, user usually have to start their remote desktop manually via a command line or similar tools.

With our virtual desktop technology there is no need to manually start anything. The back-end of **CORVUS®** takes care of sessions, ports, passwords etc., makes the relevant settings and saves it in the global database for you. Not only the settings and configurations will be done automatically by the back-end but also in cooperation with the web front-end it provides the display output.

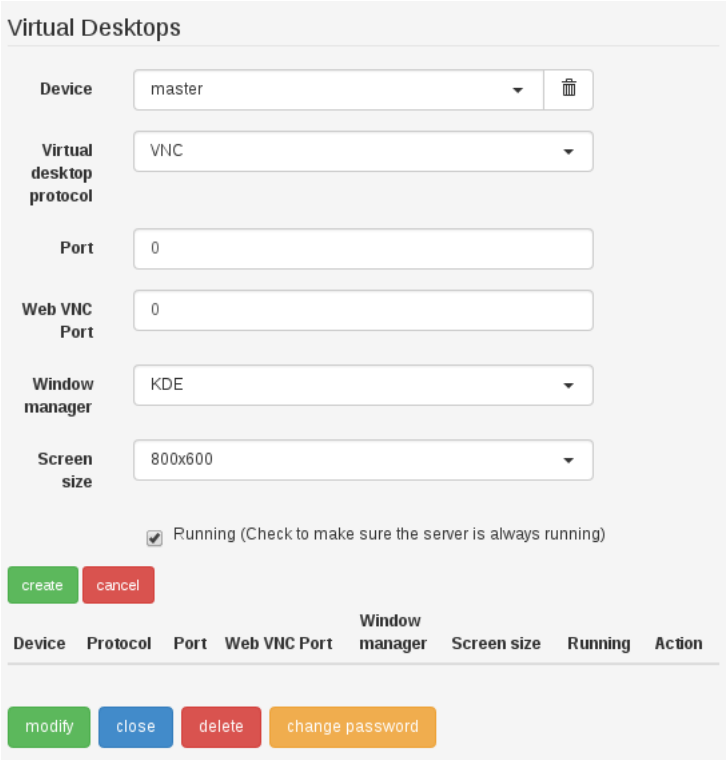
That way you are able to access and work on remote machines via the web front-end on your favorite browser.

7.1. Prerequisites

> **Important**
>
> There must be the same **user** with the same **user_id** within the user management system of **CORVUS®** and on the machine which starts the vnc-server session.
>
> If it is not true, virtual desktop session can not be used.

To activate the virtual desktop technology, first of all you have to define a **Virtual Desktop session** in **User Management**. In the main menu on top of the page navigate to **Users** **Overview**, do left mouse click on the admin user.

Figure 7.2. Virtual desktop session



Before using virtual desktops you have to define a session for it.

Virtual desktop settings

Device — Please insert text here...

Virtual desktop protocol — Protocol which will be used for virtual desktop session

Port — Portnumber of connecting client - if set to "0", port will be random

Web VNC Port — Portnumber of vnc server

Window manager — Window manager system for systems with more than one window manager

Screen size — Preset of virtual desktop size. It's the windowsize the virtual desktop will be displayed into.

Running — Checkbox to make sure the server is always running

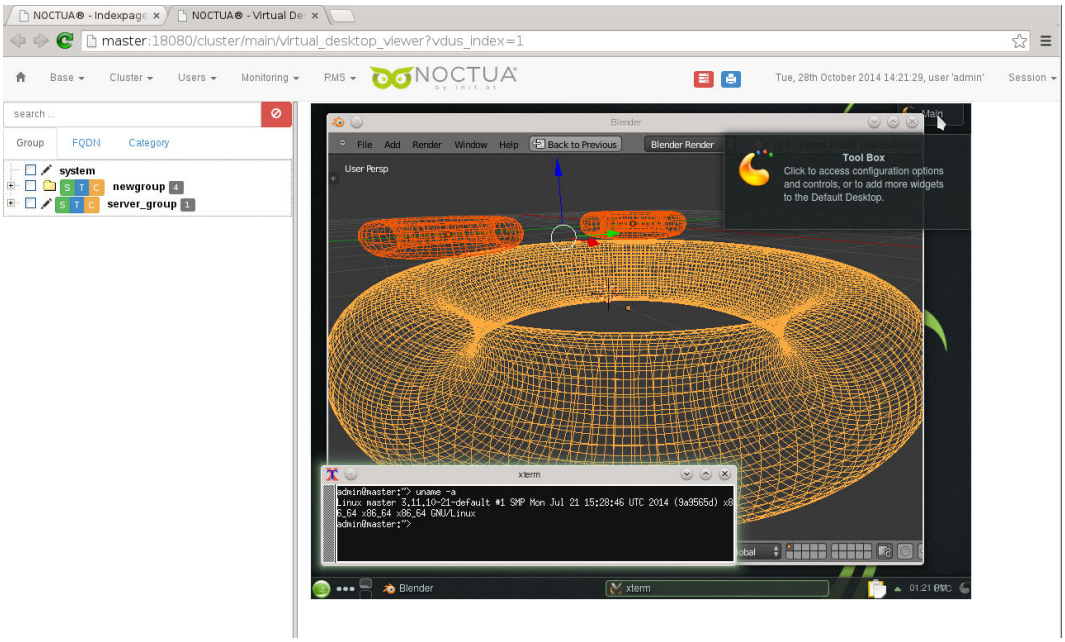
After at least one virtual desktop session is defined, the back-end takes control of the further process. It looks continuously every 5 minutes for a running vnc-server. After discovering a running vnc-server, there will be new entries and buttons in virtual desktop tab.

Now you have the choice to view your remote desktop in the main home page or in a new browser tab.

7.2. Connect to virtual desktop

Connection to remote desktop is as simple as login to your local system, even more simple like this. Just push one of the buttons and enjoy your virtual desktop inline or in new opened tab.

Figure 7.3. Virtual KDE Session



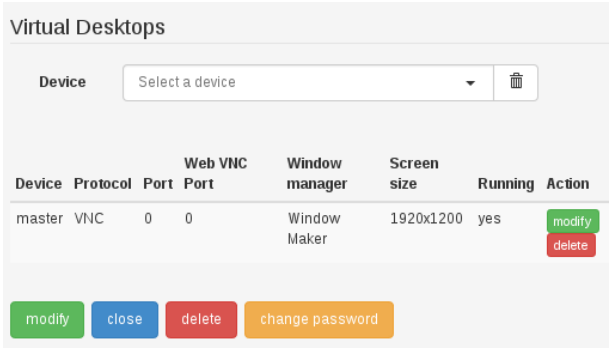
KDE session inside web front end with started 3D software and xterminal

7.3. Change settings

To change your window manager or change the virtual desktop screen size, simply navigate to **Users**  **Overview** and choose the user of virtual desktop session.

Scroll down to section "Virtual Desktops", change setting and push the modify button to change settings.

Figure 7.4. Modify settings for vnc



> **Important**
>
> Keep in mind that running processes and programms will be halt after modifying virtual desktop settings.

Chapter 8. SNMP discovery

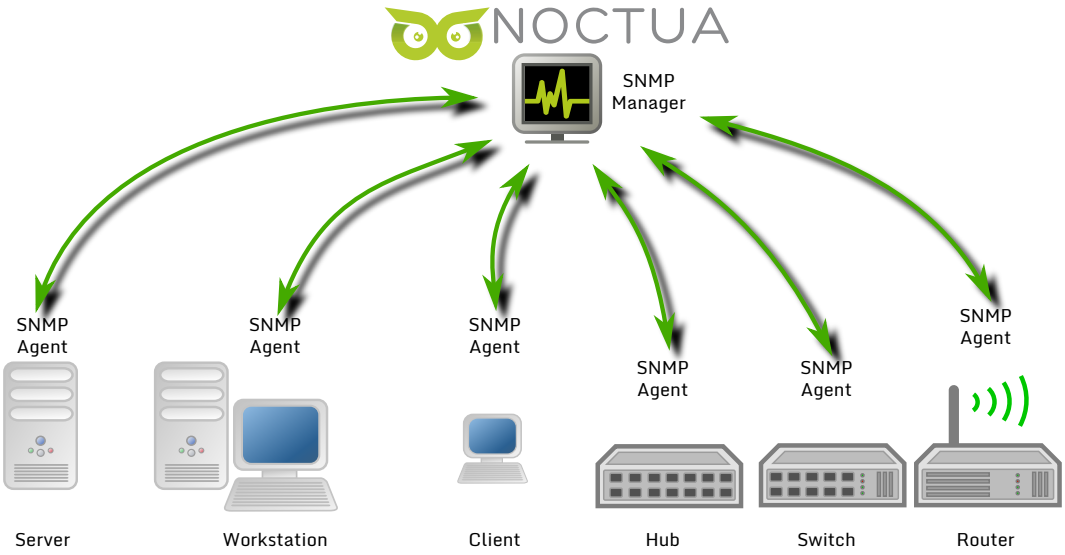
The **S**imple **N**etwork **M**anagement **P**rotocol is a official RFC internet-standard-protocol which is designed to handle variables of devices like switches, router, server, workstation, printer, bridges, hubs, and more.

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 8.1. SNMP Agents and Manager



Agents and Manager

8.1. 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.

8.1.1. Setup of SNMP discovery

To activate SNMP discovery for one device, simply select the checkbox **Enable perfdata, 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.

8.1.2. Auto discover network device

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

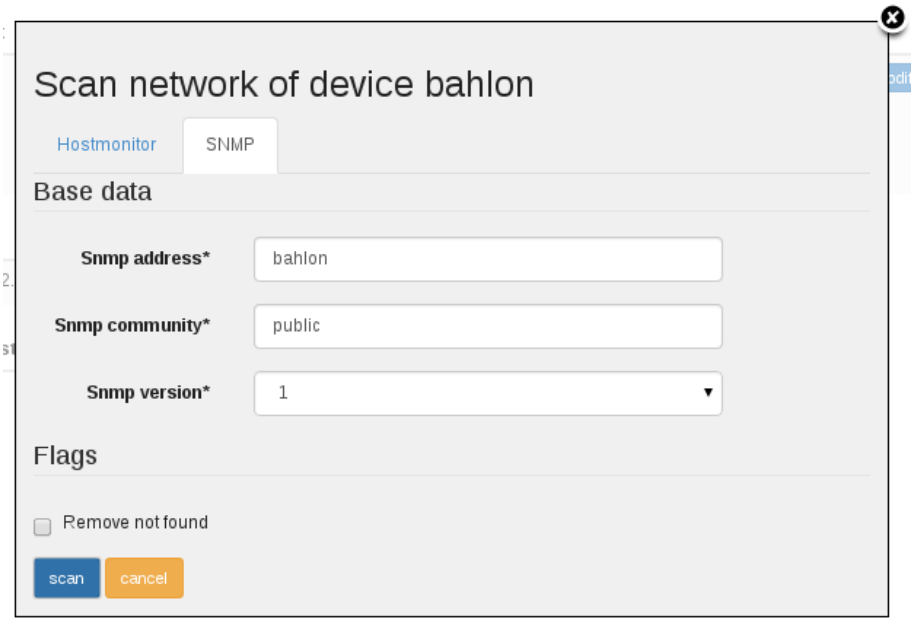
There are no SNMP schemes yet in the settings window. Now perform a SNMP scan with left click on the orange **update network** button.

Figure 8.2. Auto discover SNMP

Button to auto discover network

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

Figure 8.3. SNMP scan settings



SNMP scan settings

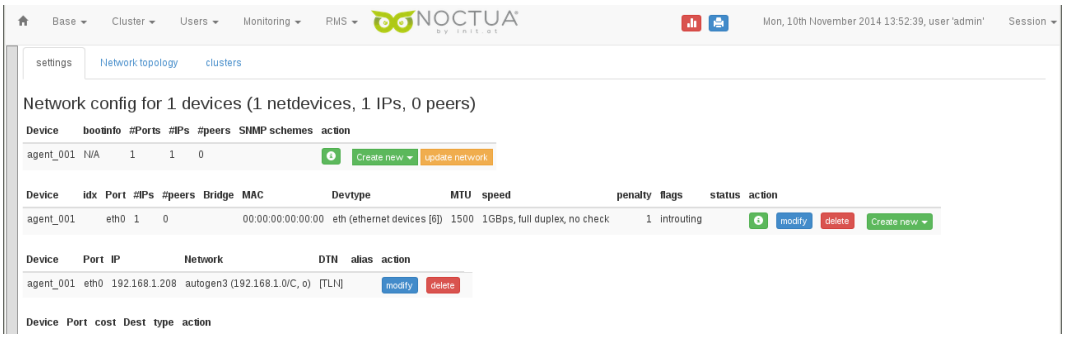
Settings

Snmp address	The IP address of the device, a valid domainname or a valid host name.
Snmp community	SNMP security settings, either public or private
Snmp version	Number of snmp version, either 1 or 2
Remove not found checkbox	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.

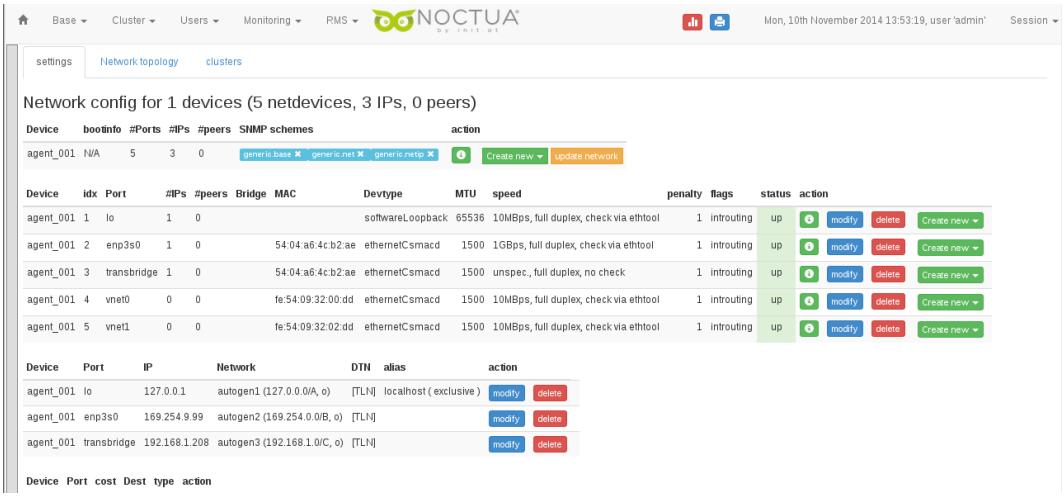
After performing SNMP scan, you will get some new network confic entries for the scanned device.

Figure 8.4. Before SNMP scan



Network config for one device before scan

Figure 8.5. After SNMP scan



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.

Chapter 9. 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.

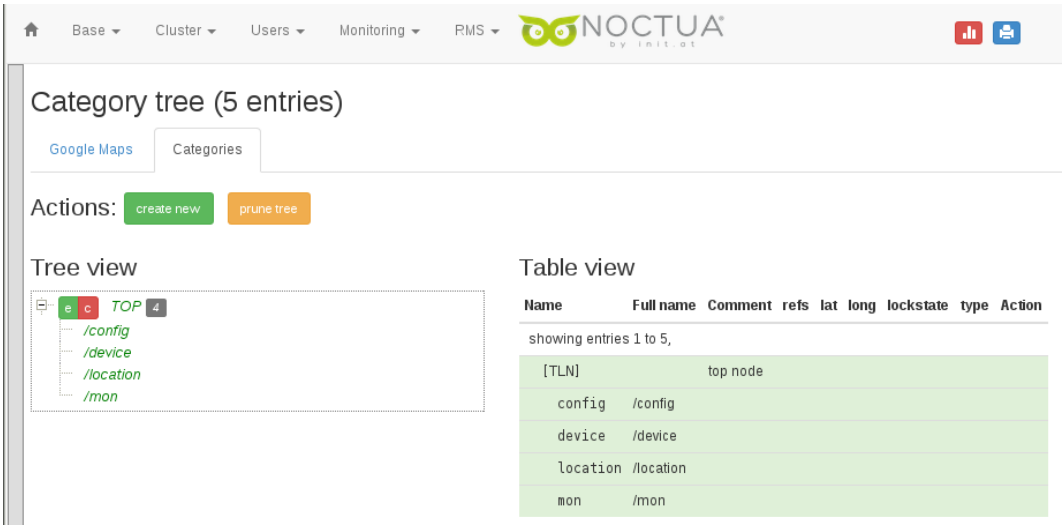
No matter if structural or physical locations, both of them have to be configured the same way.

9.1. 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 9.1. Location setting



Location settings inside category tree

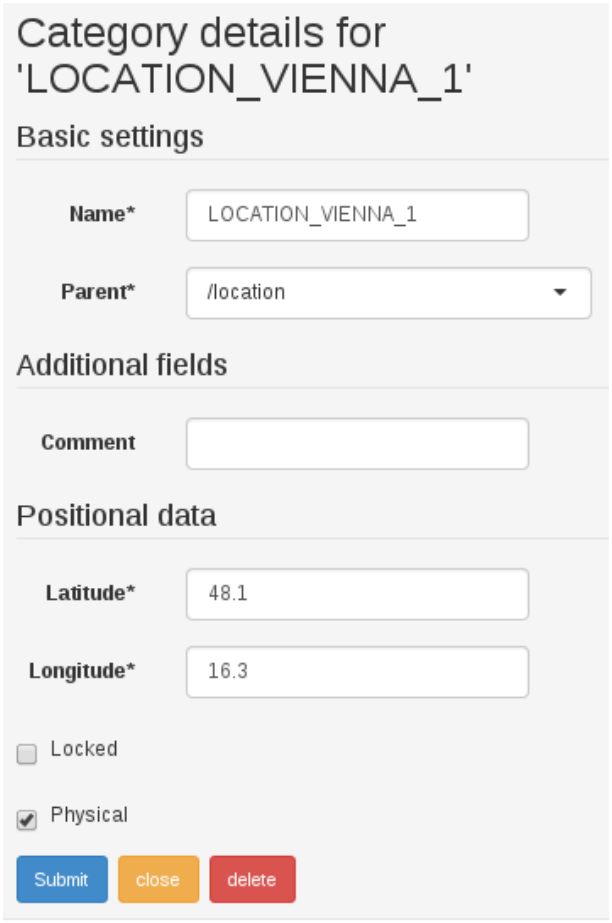
Left click on **create new** button, a new window appears below. Enter a new category name and choose *location* as parent category.

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

Advanced location settings

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

Figure 9.2. Advanced location setting



Advanced location settings

9.1.1. 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.

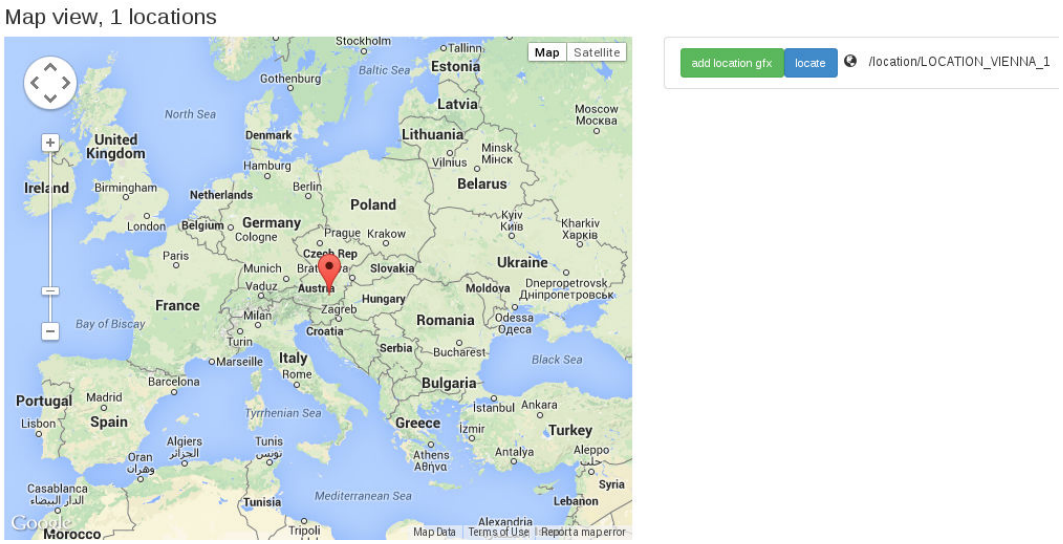
The blue **locate** button zooms the map in. With the green **add location gfx** button you are able to upload user image maps in two steps:

- 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

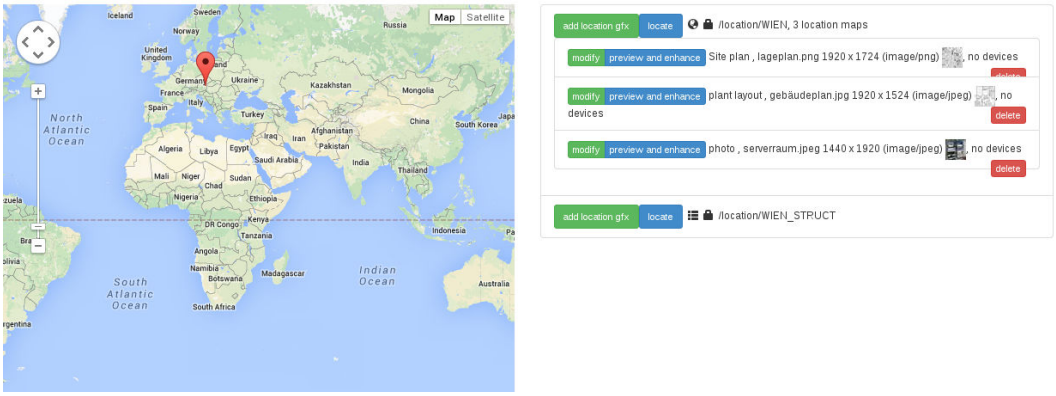
Figure 9.3. Advanced location setting



Advanced location settings

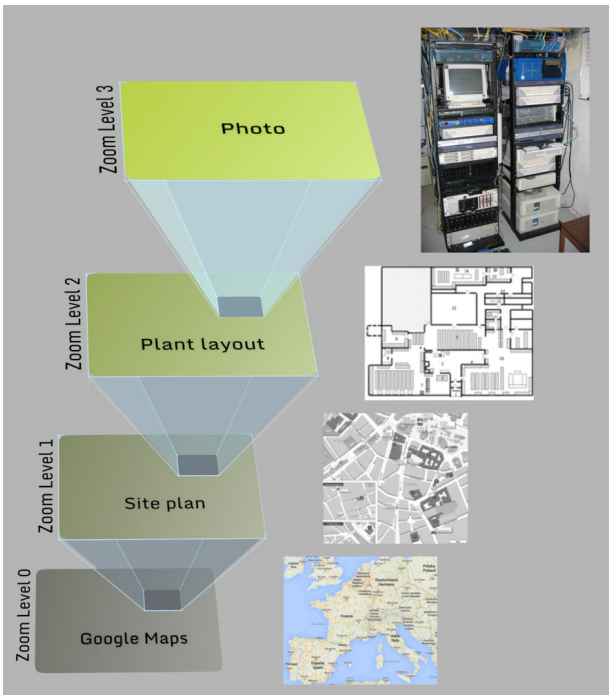
Of course you can add even more than just one user image, so you can create a stepwise zooming from google map to detailed server room photographs.

Figure 9.4. Three user images added to location



Zoom levels with according user image maps

Figure 9.5. Concepts of zoom levels with multiple image maps



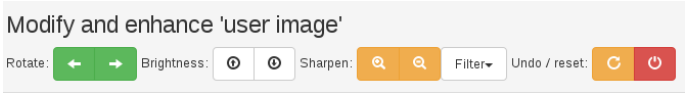
Zoom levels with according user image maps

9.1.2. Edit uploaded photos

CORVUS® 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 9.6. 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

9.2. 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 serverroom for example.

9.2.1. Binding livestatus burst to maps

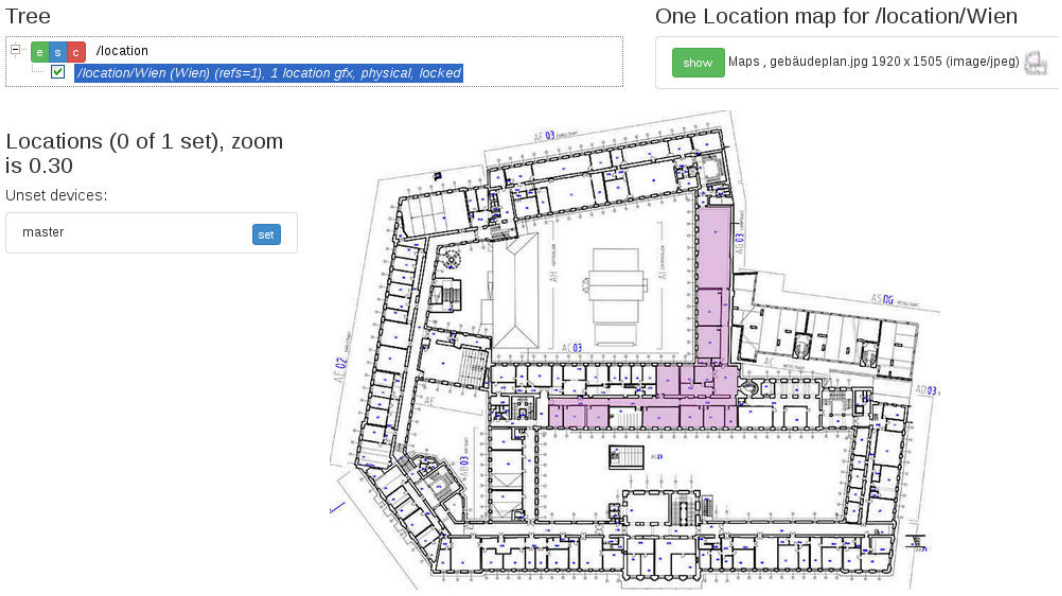
Once you have created new location categories and added some photos or images, you can easily add device livestatus to it.

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 informations 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 9.7. Adding livestatus to image maps



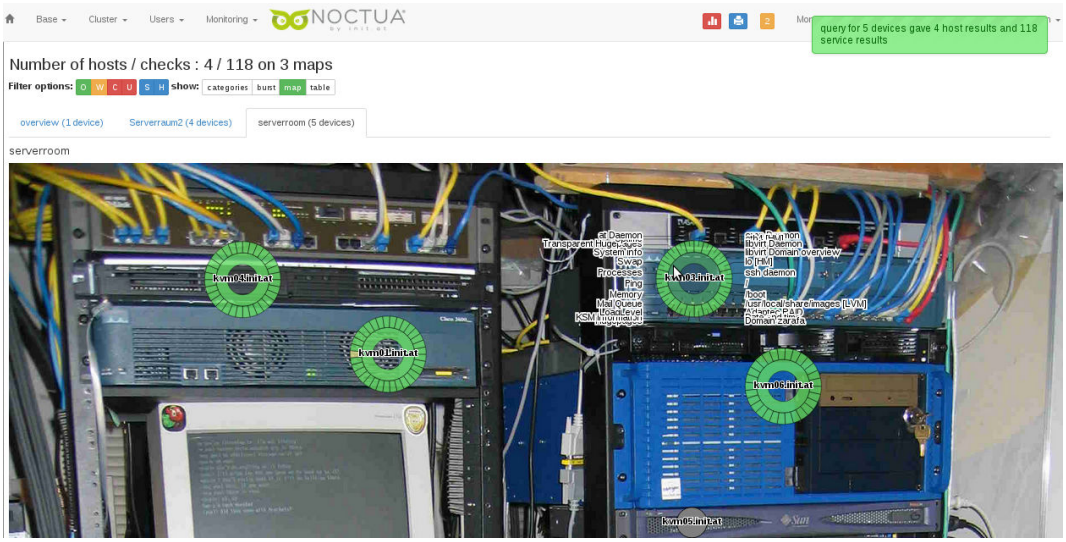
After placing livestatus burst on the right place left click on the **lock** button to prevent the livestatus burst from moving.

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

9.2.2. 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 9.8. Adding livestatus to image maps



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.

There are almost no limits about which device can be monitored. Typical devices are:

- Fileserver
- Cluster
- Webserver
- Switches
- Printer
- Router
- Telephone systems
- Thin clients

10.1. First simple check setup

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.
3. Enable the **check_ping** config to activate the check.

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

10.2. Extended monitoring setup

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.

10.3. Evaluation of monitored data

COMMING SOON$$

10.4. Overview Configuring Nodes

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

1. Variables

2. Monitoring Config

3. Scripts

Variables let your override **CORVUS®** specific settings or pass information into **CORVUS®**. 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 ...

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`*]

10.5. Foo

Now that you have your first node up and running we have to say something about configuring nodes. `START_SCRIPTS` , `INIT_MODS` INIT_MODS specifies which modules are loaded (per TFTP from the server after when the initial ramdisk boots The only module that has to be included in the initial ramdisk is the module required for your network hardware.

10.6. Monitoring

The monitoring configurations support the following syntax. $USER1$ $USER2$ $USER3$ Commands containing @ are special. They are created for all discs, network interfaces, ... @DISC@

Example 10.1. Monitoring configuration to check diskspace

```
$USER1$ df -e 90 -w 80
```

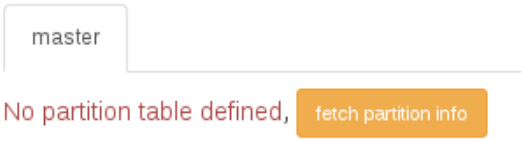
10.7. Discovery server

An other special feature of **CORVUS® / 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")

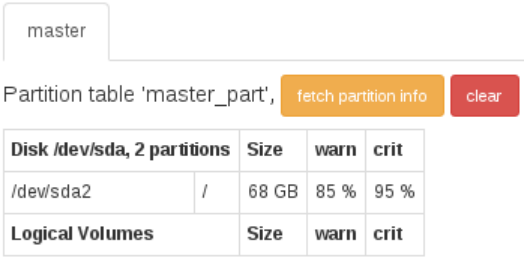
Now you can easily get partition data by pushing the **fetch partition info** button.

Figure 10.1. Before fetching partition information



Before fetching partition information

Figure 10.2. After fetching partition information



After fetching partition information

Chapter 11. 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 **CORVUS®** there are two different methods to create checks (icinga commands).

11.1. Types of checks

11.1.1. 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.

11.1.2. 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.

11.2. Examples

11.2.1. 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.

11.2.2. 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.

11.3. 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.

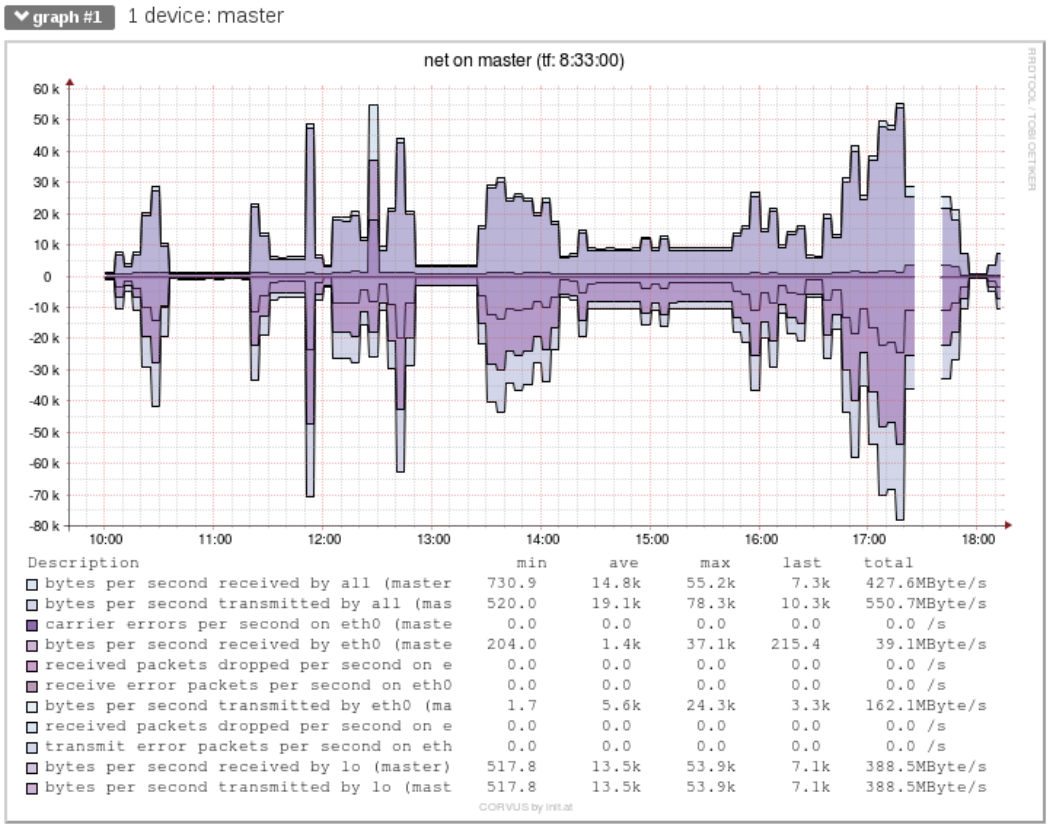
An other advantage of parameterizing is the possibility to react faster in case of alternation established setup.

Chapter 12. Graphing

12.1. Introduction to Graphing

Graphs are one of the most important tools for monitoring devices. They allows you to create graphs of collected data for different timeranges 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 12.1. 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 timelime of displayed graph
total	Total amount of displayed graph

12.1.1. 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.

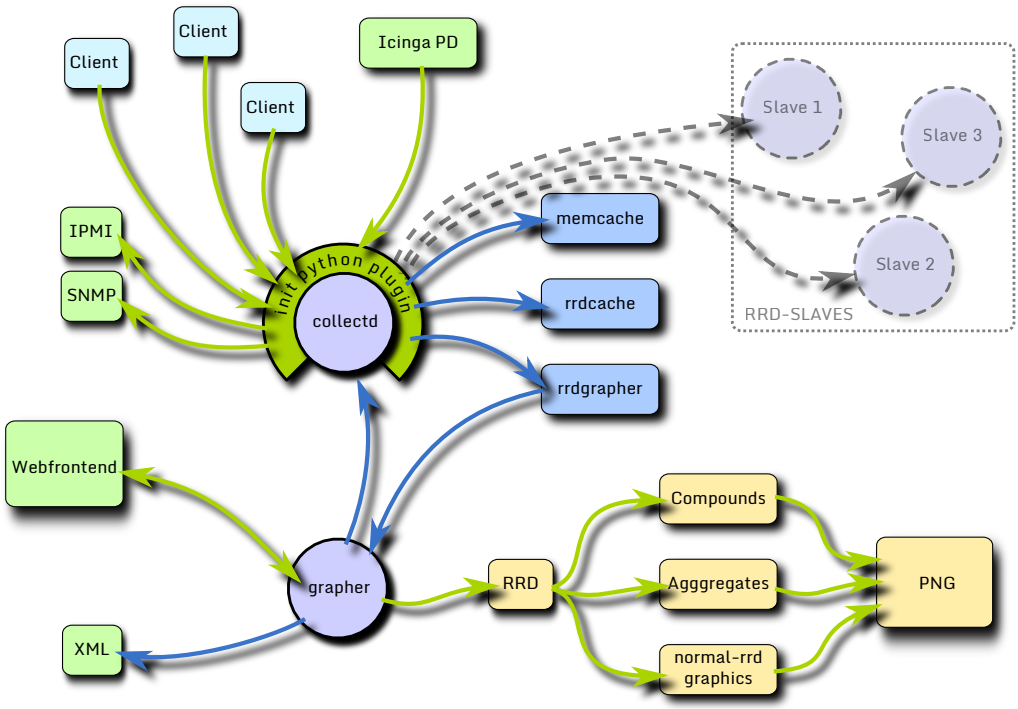
12.1.2. Data collection and graphing

To collect data and draw graphs in **CORVUS®** and **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 **CORVUS®** or **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 12.2. RRD graph cycle



Cycle of rrd graphic dataflow

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

12.1.3. How to display RRD graphs?

Select one ore 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.

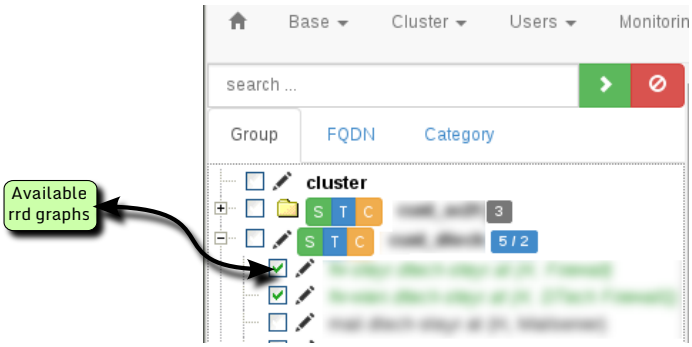
In both cases there will be some new tabs displayed, one of them named **Graphs**

12.1.4. 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. Figure 12.3, “ Available rrd graphs ”

Figure 12.3. Available rrd graphs



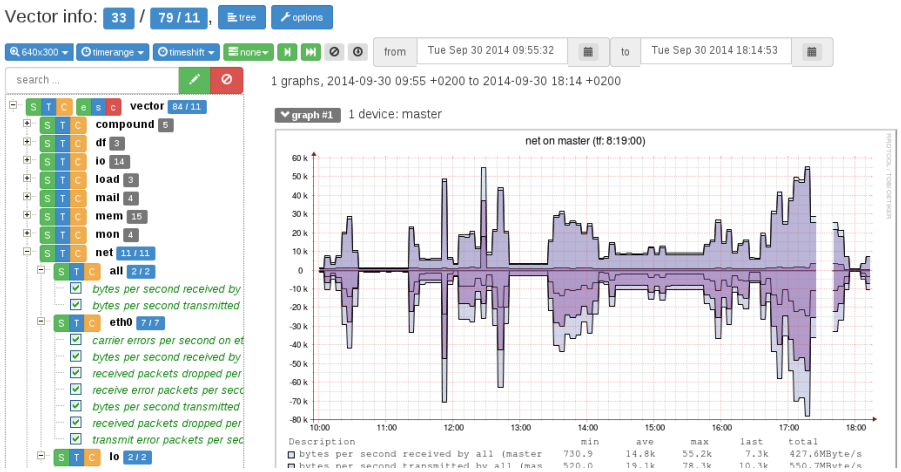
Existing rrd graphs marked with pencil logo.

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 12.4. RRD front-end



Front-end inside of **CORVUS® / NOCTUA®**

Graphic size 

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 

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)
last day	draw the whole last day (00:00 -23:59)
current week	draw the whole current week (sunday -saturday)
last week	draw the whole last week (sunday -saturday)
current month	draw the whole current month
last month	draw the whole last month
current year	draw the whole current year (Jan - Dec)
last year	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 drawed 1 week ago.

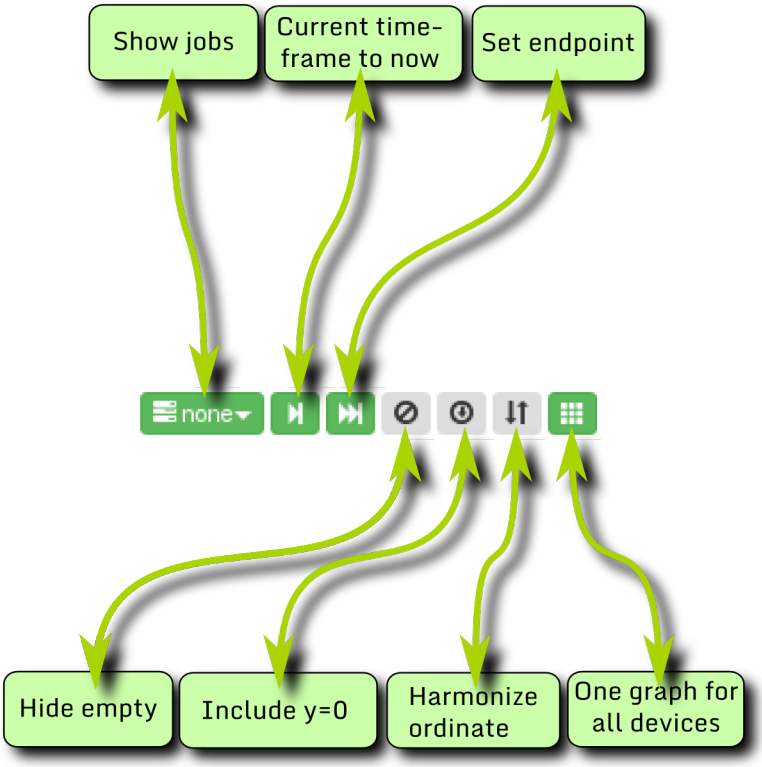
Timeshift

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

1 year	draw one normal graph plus the same graph 1 year (365 days) later (dotted)

Controlbuttons

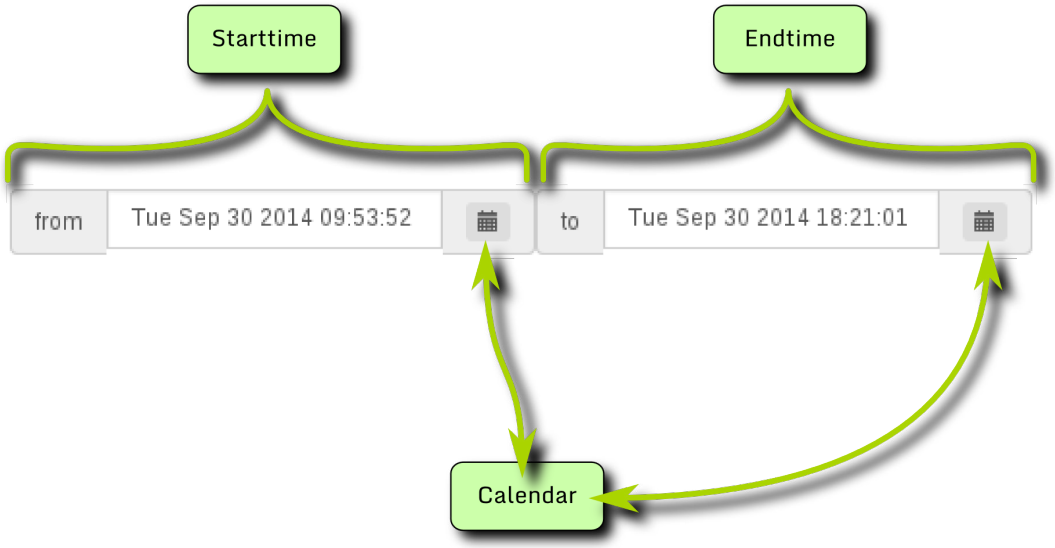
Figure 12.5. Options



Additional options

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

Figure 12.6. Date section



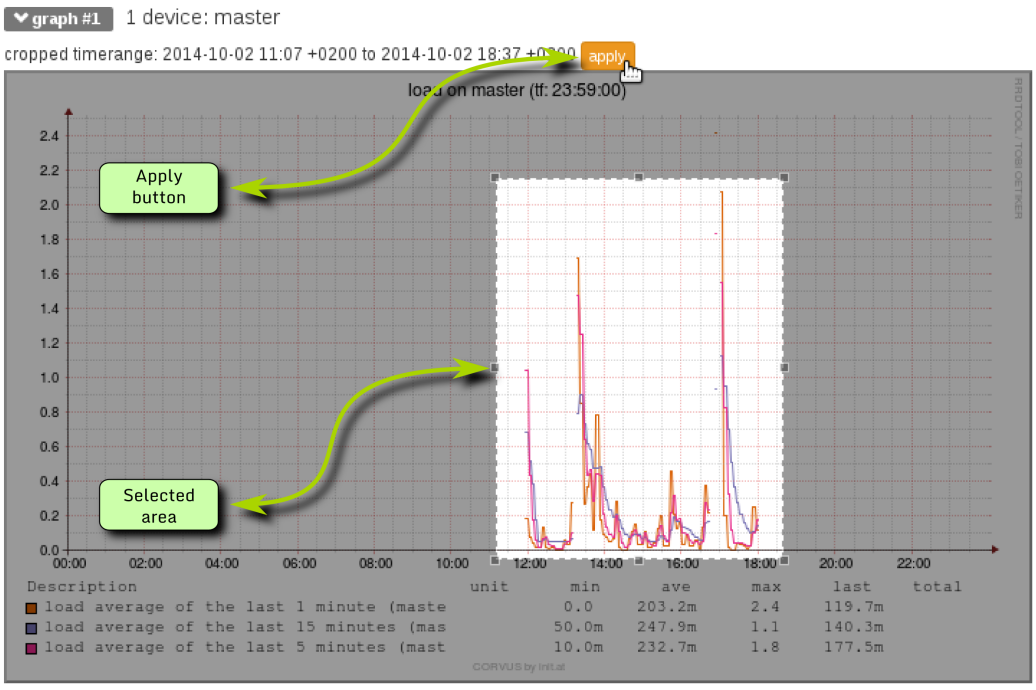
Start and endpoint for drawing date

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 12.7. Zoom area



Zooming area ready for apply

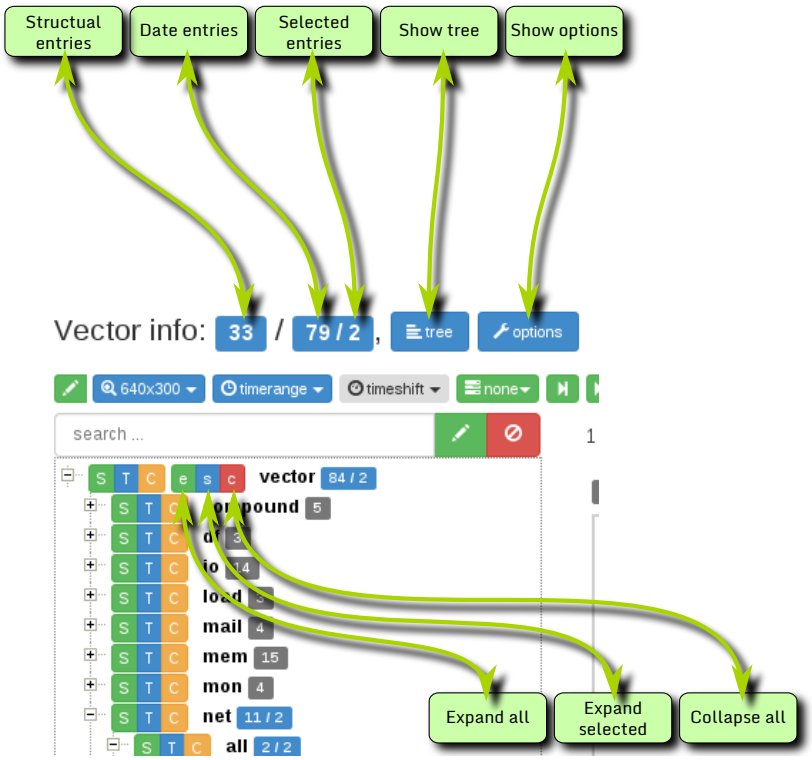
IMPORTANT

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 12.8. Tree

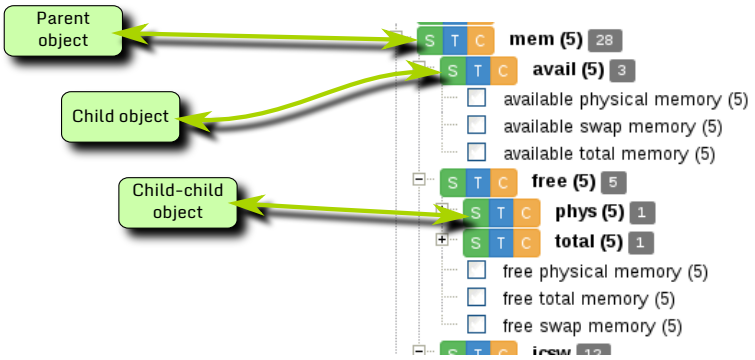


Tree organisation of rrd-data

12.1.5. 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**.

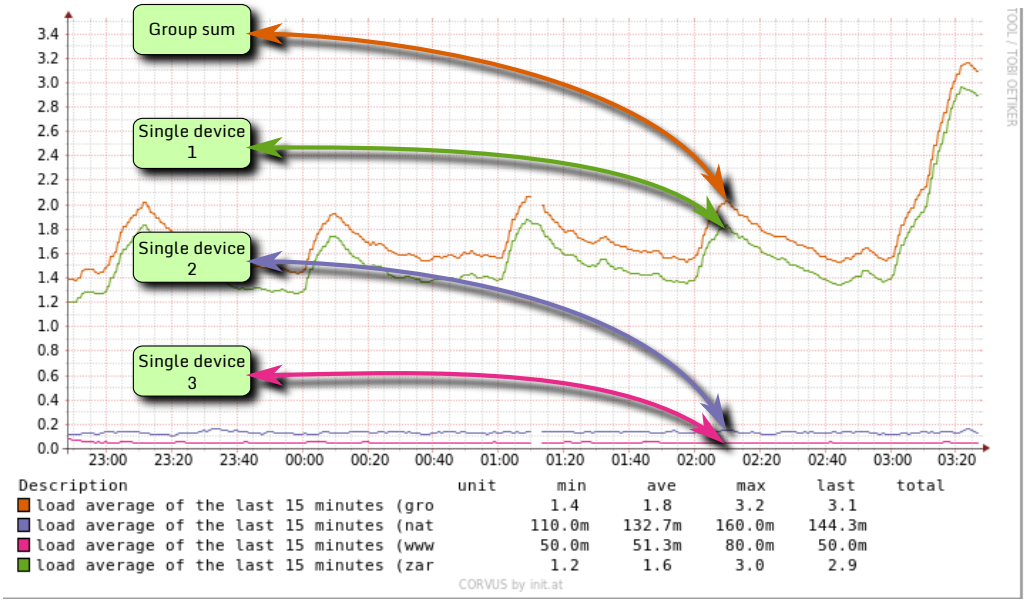
Figure 12.9. Tree parents and childs



Showing tree parents und children

12.1.6. Summarized graphs

Figure 12.10. Aggregation of 3 devices



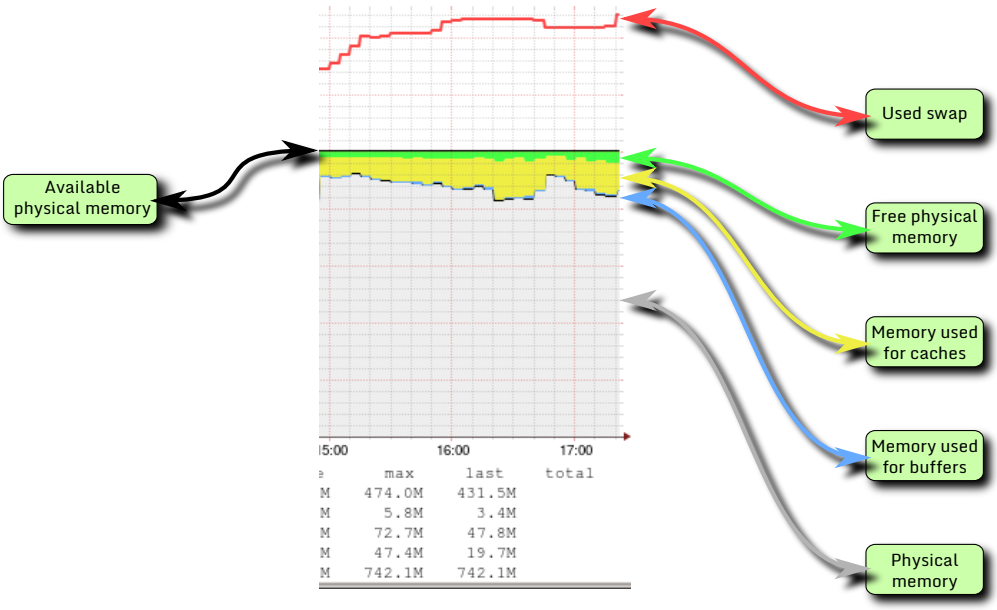
Data aggregation of 3 devices

For parent groups it makes sense to summarize some graphs. This kind of summarization is called *aggregation* in **CORVUS®** and **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.

12.1.7. Compound graphs

Figure 12.11. 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 13. Debugging and Error hunting

13.1. General information

To obtain information about the general status of **CORVUS®** use **check_scripts.py** .

Example 13.1. Using check_scripts.py to view running services

```
clusterserver:~ # check_scripts.py -t --mode show --server ALL

Name                  type   Thread info            status
-------------------------------------------------------------
logcheck-server       server not installed          skipped
package-server        server not installed          skipped
mother                server not installed          skipped
rrd-grapher           server all 9 threads running  running
rms-server            server not installed          skipped
cluster-server        server all 5 threads running  running
cluster-config-server server not installed          skipped
host-relay            server all 13 threads running running
snmp-relay            server all 21 threads running running
md-config-server      server all 21 threads running running
```

13.2. 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 13.2. Using lse to display the last error

```
clusterserver:~ # lse -l 1


Found 40 error records
Error 40 occured yesterday, 17:12:47, pid 11507, uid/gid is (30/8 [wwwrun/ww
  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_typ
```

13.3. 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 13.3. Retrieving information from nodes

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

13.4. Logging

CORVUS® and **NOCTUA®** provides its own logging service. In case of something goes wrong the logging-server writes its logs under `/var/log/cluster/`. 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.

13.4.1. Automatic log mail delivering 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 if there are 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.

```
# 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:

rclogging-server restart

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

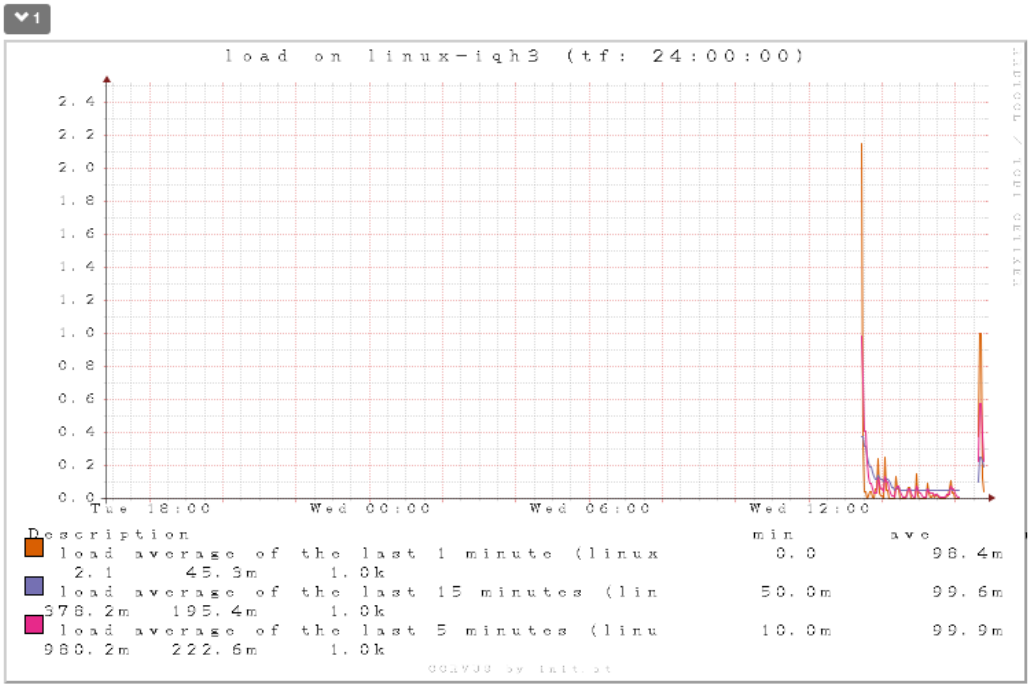
Chapter 14. Frequently Asked Questions

Collection of repeated Questions about **CORVUS®** and **NOCTUA®**.

14.1. Miscellaneous questions

14.1.1. Bad looking font in RRD Graph

Figure 14.1. Bad looking fonts



ugly looking fonts due to wrong font setup

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

14.1.2. Server Error (500)

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.

14.1.3. Unable to connect

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

14.1.4. An error occurred

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"

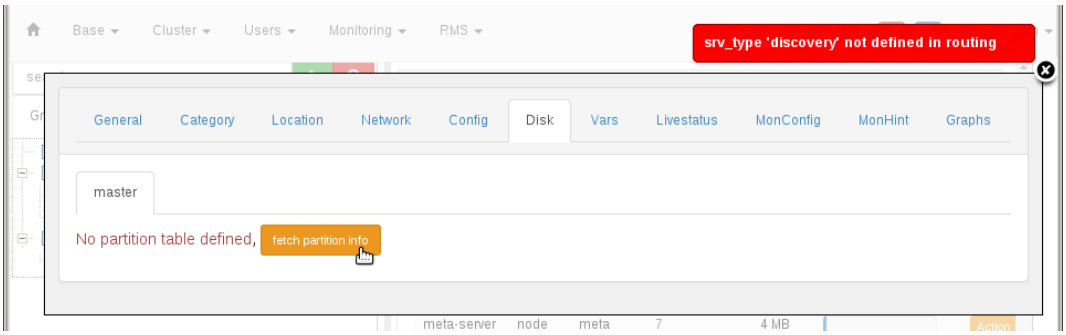
14.1.5. My changed config don't be applied.

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

14.1.6. Discovery not defined in routing

Most likely the discovery-server is not installed or discovery-server service is not running. Make sure the discovery-server is installed and running. Run the **check_cluster.sh** command and look for "discovery-server". If it is not running, start it either by commandline **rcdiscovery-server start** or via the webfrontend under **cluster server information**

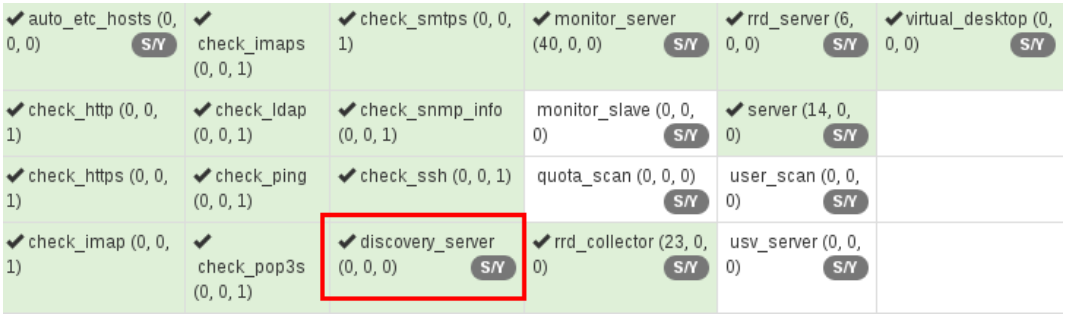
Figure 14.2. discovery-server not defined in routing



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** settings.

Figure 14.3. Discovery-server settings



Enabled discovery server config

14.1.7. Slow Network topology graph

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™.

14.1.8. Lost login password

Sometimes it could be useful to reset a user password. For example if someone forget the password or something else goes wrong.

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

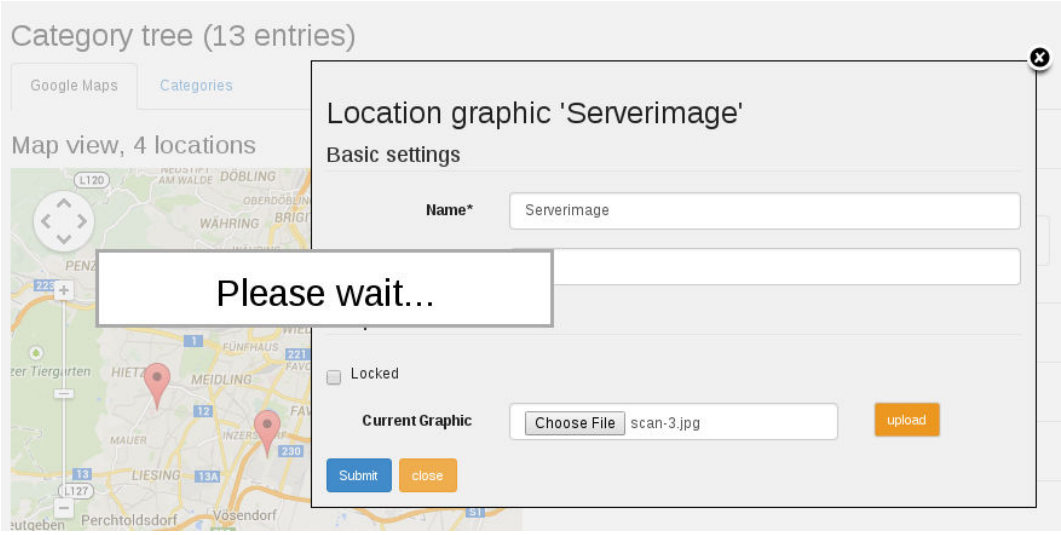
   exit()

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

14.1.9. "Please wait..." after add location gfx

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 resolv this issue.

Figure 14.4. Please wait ...



"Please wait..." message after image upload

14.1.10. Weird mouse events on virtual desktop

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.

Glossary

Django	Django is a free and open source web application framework, written in Python
free Software	Free Software is software licensed under a free license, allowing you to use, change and redistribute the software under the same license.
GNU GPL	Free software license named GNU General Public License, more Information at [http://www.gnu.org/licenses/]
Network	Group of devices connected to each other over data connections.
NFS	Network File System
Parameterized check	Some option values of this check (command) can be accessed by parameter.
PXE	Preboot Execution Environment