Administrator Handbook

The Somewhat Definitive Guide

Dr. Andreas Lang-Nevyjel <lang-nevyjel@init.at>
Natalia Hösl <hoesl@init.at>
Christoph Sieghart <sieghart@init.at>

Administrator Handbook: The Somewhat Definitive Guide

by Dr. Andreas Lang-Nevyjel, Natalia Hösl, and Christoph Sieghart

Release: 1.3
Copyright © 2014 init.at informationstechnologie GmbH

Abstract

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

Table of Contents

1. Core concepts 1
2. Installation 3
    Clusterserver 3
        Prerequisites 4
        Database setup 4
        Accessing the webfrontend for the first time 5
        Needed Server processes for nodeboot 5
    Node Setup 5
    Functionality 5
    Configurations 5
3. Basic Administration - adding devices 7
    Populate the device tree 7
        Create devicegroup 7
        Create device 9
        Bulk modification/deletion 12
    Network configuration 15
        Networks 16
        Device Network 17
    Virtual routing 21
    Configuration setup 29
        General checks configuration 29
        Device configuration 36
    Domain name tree 37
    Category tree 41
    Device variables 48
    User definition 52
    Monitoring setup 65
    Adding nodes 69
4. Configuring Nodes 72
    Overview 72
    Foo 72
5. Installing and distributing packages 73
    Installing and distributing packages 73
    RMS - Resource Management Systems 73
        SGE 73
6. Debugging and Error hunting 74
    General information 74
    Show errors 74
    Node information 75
    Important logfiles 75
7. Monitoring 76
8. Troubleshooting 77
    Problems accessing the webfrontend 77
9. Frequently Asked Questions 78
    Q1: Why does my font in RRD Graphics is not readable? 78
Glossary 79

List of Examples

3.1. Using populate_ramdisk.py to create a ramdisk 70
3.2. foo 71
6.1. Using check_scripts.py to view running services 74
6.2. Using lse to display the last error 74
6.3. Retrieving information from nodes 75
7.1. Monitoring configuration to check diskspace 76

Chapter 1. Core concepts

Abstract

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

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

Remove this block!

cluster-server	Responsible for writing config files and general coordination of the cluster. Listens on port **TCP/8004**. cluster-server.py is a general purpose server that handles various tasks like writing /etc/hosts, generating a valid DHCP configuration, configuration of the BIND nameserver, feeding LDAP and / or YP servers, ...
mother	Creates the tftpboot /ethernet structure, monitors the installation progress of the nodes. Listens on ports **TCP/8000 TCP/8001**. Mother provides access to IPMI as well.
cluster-config-server	Generates the files for the Clusternodes (based on the config stored in the Database) to make the nodes distinguishable.
package-server	Provides repositories available for installation by the package-client. Listens on port **TCP/8007**.
package-client	Install required software by using the locally available package management commands. **zypper**, **yum** or **apt-get**.
hoststatus	A small program written in C that transmits node status messages to the cluster-server. The hoststatus is written in C to be easily includable in the initial ramdisk.It listens on port TCP/2002. hoststatus is in the package child
meta-server	This daemon is responsible for restarting services that might have crashed or were otherwise killed. This functionality should be taken over by systemd. `/var/lib/meta_server` contains the relevant information about which services should be running.
logging-server	Creates the structure needed for receiving logs via rsyslog or syslog-ng.
logcheck-server	Do log rotation and deletes logs older than a specified time range.
collclient.py	Client part of the host-monitoring collserver.py.
ccollclientzmq	Frontend program to talk to collrelay.
rms-server.py	Provides integration of **CORVUS®** with SGE. The commands **sns** and **sjs** rely on it.

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

Clusterdatabase: The database where all the Configuration data of the cluster installation is stored its generally referred to as the "clusterdatabase".

Throughout this document we might refer to *Server side scripts* and *Node side scripts*. Server side scripts are generally services or scripts that are running on the clusterserver. Most of them need database connectivity to function properly. On the other hand "node-side script" are daemons or scripts that are generally run on a node of the cluster. Node-side scripts don't require access to the cluster Database.

Target states are associated with nodes: boot_into_production (boot and load config from the clusterserver boot_into_produciton_clean (boot, don't load config from the clusterserver) install_into_production(boot, copy image to disk, load config from clusterserver) install_into_production_clean(boot, copy_image to disk, do not load config)

Chapter 2. Installation

Clusterserver

CORVUS® packages are available for

- Debian Squeeze
- Debian Wheezy
- Ubuntu 12.04
- CentOS 6.2
- CentOS 6.4
- openSuSE (12.3, 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.

DEB based

- deb http://www.initat.org/cluster/DEBs/FULL_NAME/base CODENAME main
- deb http://www.initat.org/cluster/DEBs/FULL_NAME/cluster CODENAME main
- deb http://www.initat.org/cluster/DEBs/FULL_NAME/extra CODENAME main

RPM based

- http://www.initat.org/cluster/RPMs/suse_REPL/base
- http://www.initat.org/cluster/RPMs/suse_REPL/cluster
- http://www.initat.org/cluster/RPMs/suse_REPL/extra

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
- at
- 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:

- nginx
- uwsgi
- host-monitoring
- at

Prerequisites

CORVUS® is using Django as its database layer 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.

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
```

Database setup

The necessary commands to create the database are printed via the following command:

```
/opt/python-init/lib/python/site-packages/initat/cluster/manage.py sqlcreate
```

Test if you can connect to the newly created database successfully. After filling in the information populate the database by calling: **/opt/cluster/sbin/create_database.sh** This command has to be run

as root and supports the option --clear-migrations which will clear all migration directories of the DB-Schema migration facility of the **CORVUS®** (this option is mainly useful as a development switch, should not be needed in production environments).

Note

Backupg1s of your clusterdatabase are kept for 14 days by **CORVUS®** The backups can be found under `/opt/cluster/share/db_backup` The backups are created by the cluster-server process at 2:00 am.

During the run of create_database.sh one has to enter the basic account data of the admin-user, the password entered here is the one needed for the webfrontend.

Accessing the webfrontend for the first time

Nearly every aspect of **CORVUS®** 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 web-frontend:

- 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 these processes simply issue the command **/opt/cluster/sbin/check_scripts.py --system memcached nginx uwsgi-init -a** which should give an output similar to

```
Name       type           pids         runlevels           Memory status
-------------------------------------------------------------------------
memcached  system running 4246         levels 3, 5   47.56 MBytes running
nginx      system running 15271-15279  no runlevels   1.99 MBytes running
uwsgi-init system running 22070/22071  no runlevels 233.19 MBytes running
```

The Webfrontend can be accessed via http://SERVERNAME/cluster

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.

Node Setup

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

Functionality

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

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. Basic Administration - adding devices

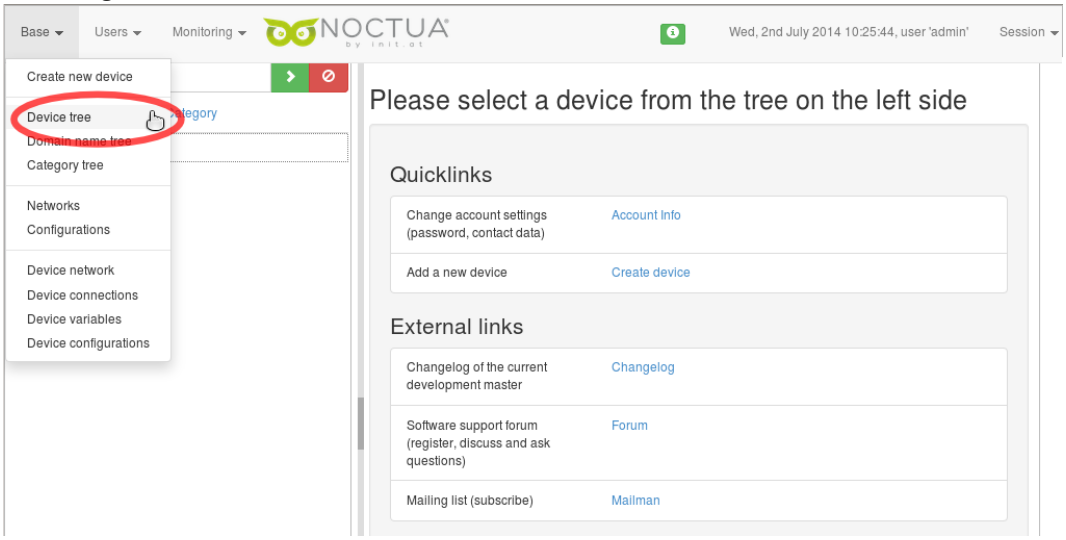
Adding devices can be split in 3 basic steps

- Add nodes to the device tree
- Configure the network settings of the added nodes
- Connect the nodes to the existing cluster (virtual routing)

Populate the device tree

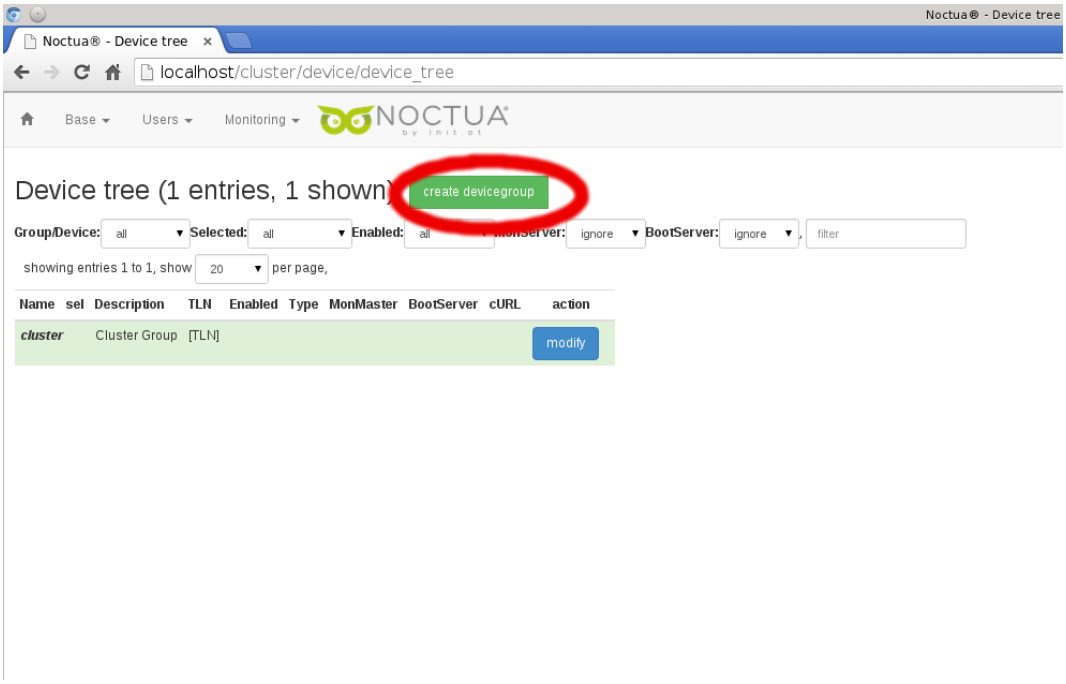
Create devicegroup

Menu option „Base“ → „Device tree“ gives the possibility to create and modify the device tree of monitoring software Noctua.

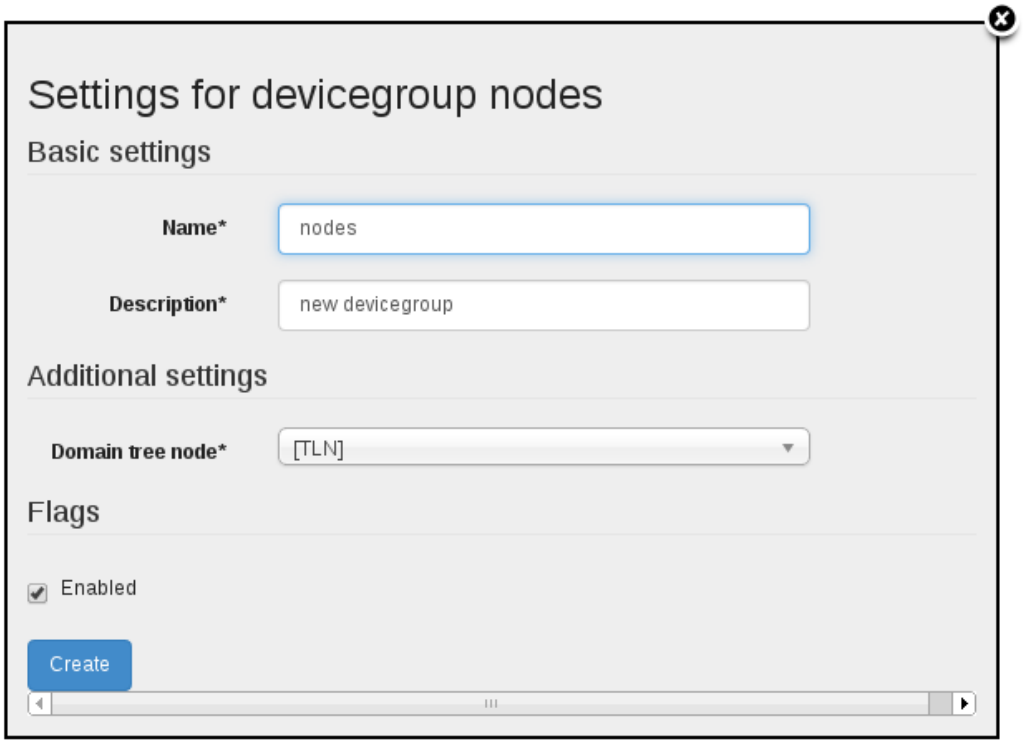


At first it is necessary to create one or many device group, which contain the devices to be monitored. The „cluster“- group is issued during the installation process. It is the main devicegroup of „Noctua“ and can not be deleted.

To create a new devicegroup, click the „create devicegroup“-button on the main view of device tree.



In appeared windows define the Name, Description and Domain tree node of the new created devicegroup. By default the new devicegroup is automatically enabled. This option can be changed later by „modify devicegroup“-option. Click „Create“ - button to complete the process.

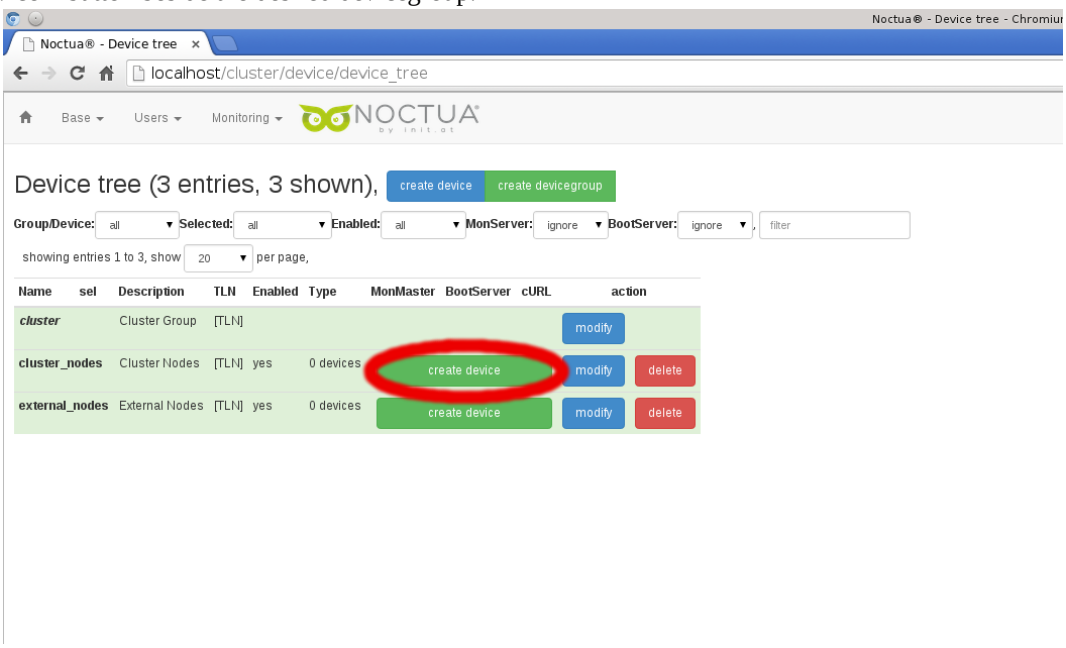


Each issued devicegroup is directly shown on the main Device tree view.



Create device

Once created, devicegroups can be filled with devices. To create a new device, hit the „create device“- button beside the desired devicegroup.



In „Device settings“ - window following options are available ('*' - required option):

Base settings

- Name*: name of device

- Comment: description of device
- Device type*: choose device type from drop-down list
- Device group*: use proposed devicegroup or select another from drop-down list, which contains all existing groups from device tree.

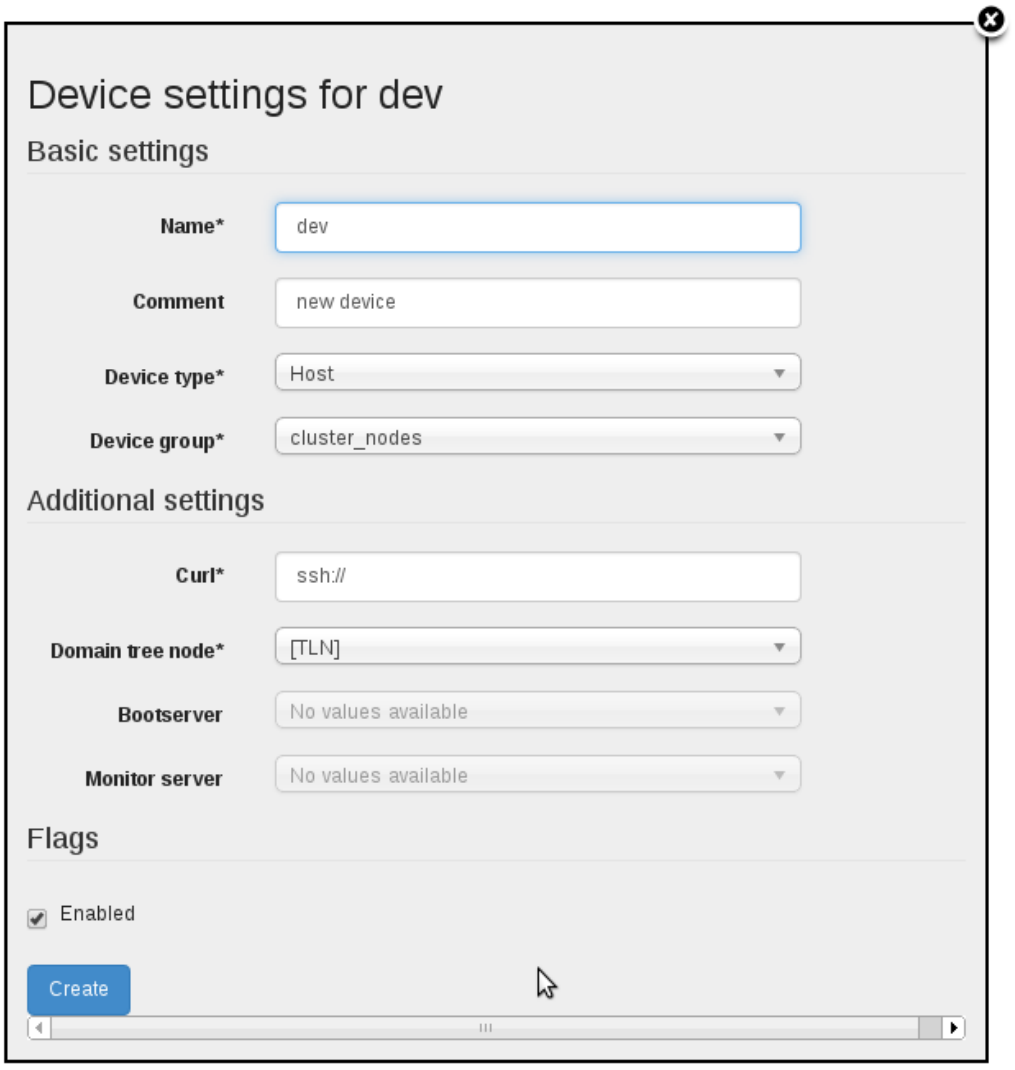
Additional settings

- Curl*:
- Domain tree node*: select the domain name from presented drop-down list. The target domain name has to be defined earlier under "Menu"-> Domain name tree menu option. Default value for this field is "TLN" ("Top level node").
- Bootserver: if there are more then one defined bootserver, select the necessary one from drop-down list.
- Monitor server: in case of many monitoring servers in network, here can be chosen the one of them.

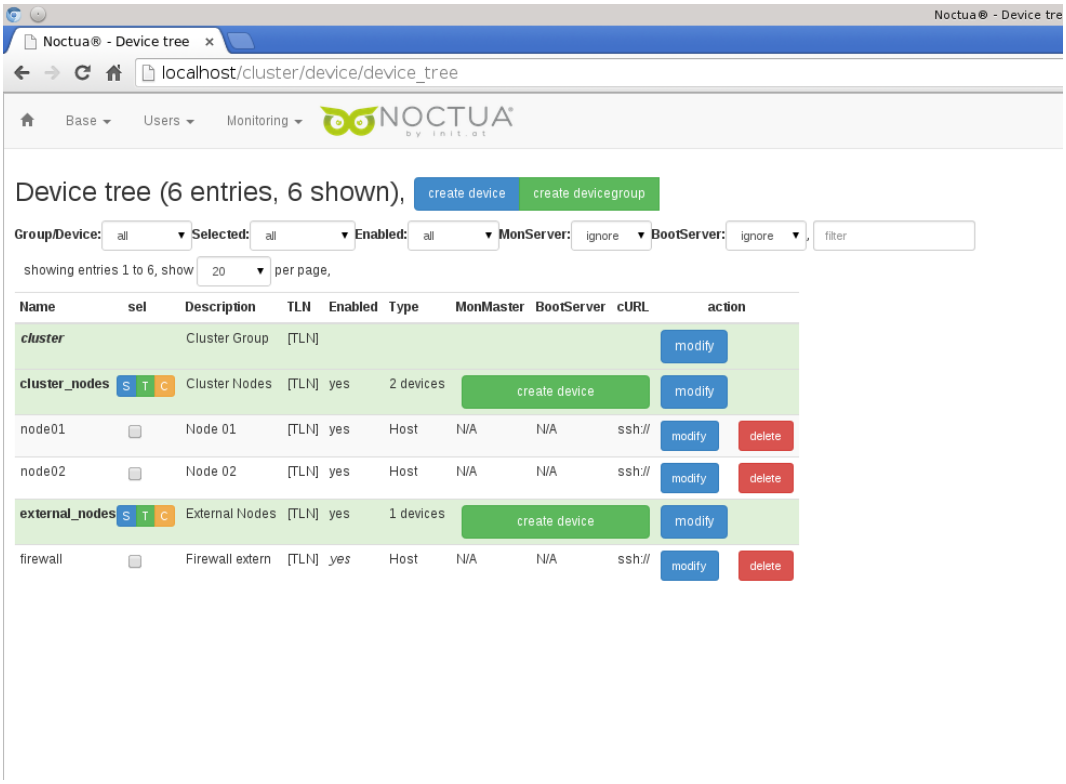
Flags

- Enabled: each device is automatically enabled. Clear this checkbox, if the device has to be disabled.

Confirm the attributes of new device with click on "Create" - button.

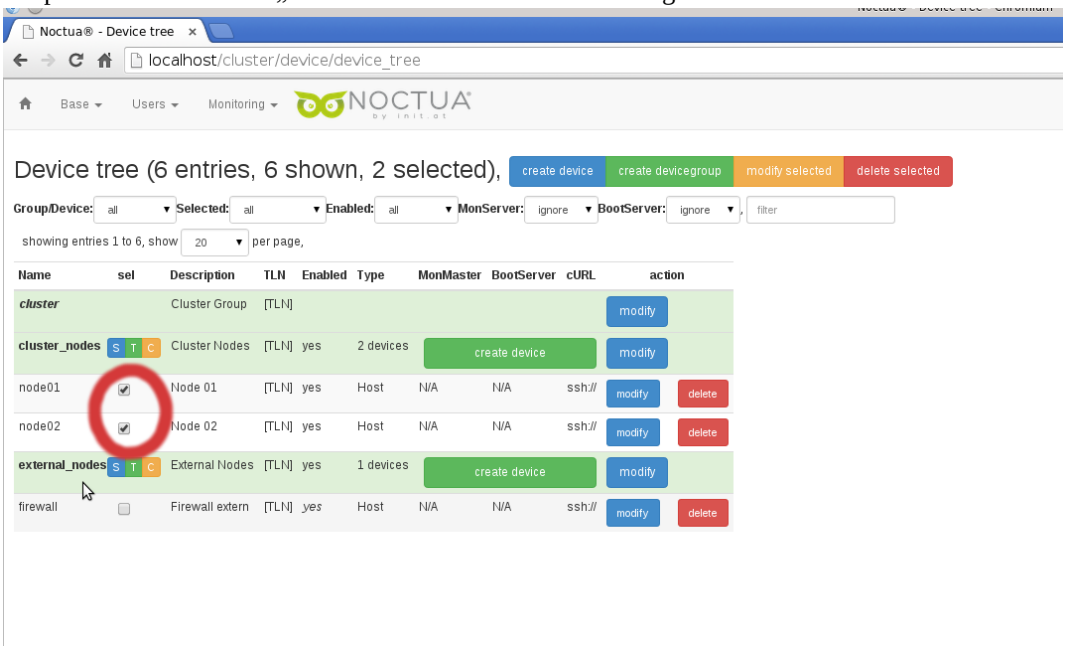


On the device tree view all defined devicegroup and devices are listed in alphabetic order. For each device there are two buttons for „Modify“ and „Delete“ - options. The devicegroup can not be deleted, while it includes any device.

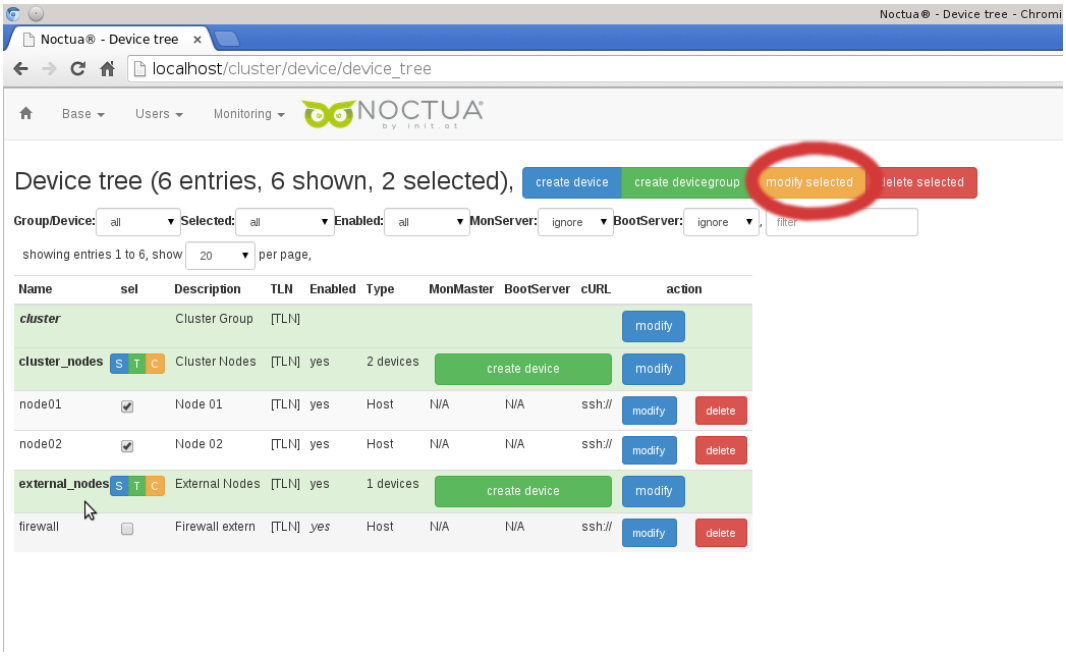


Bulk modification/deletion

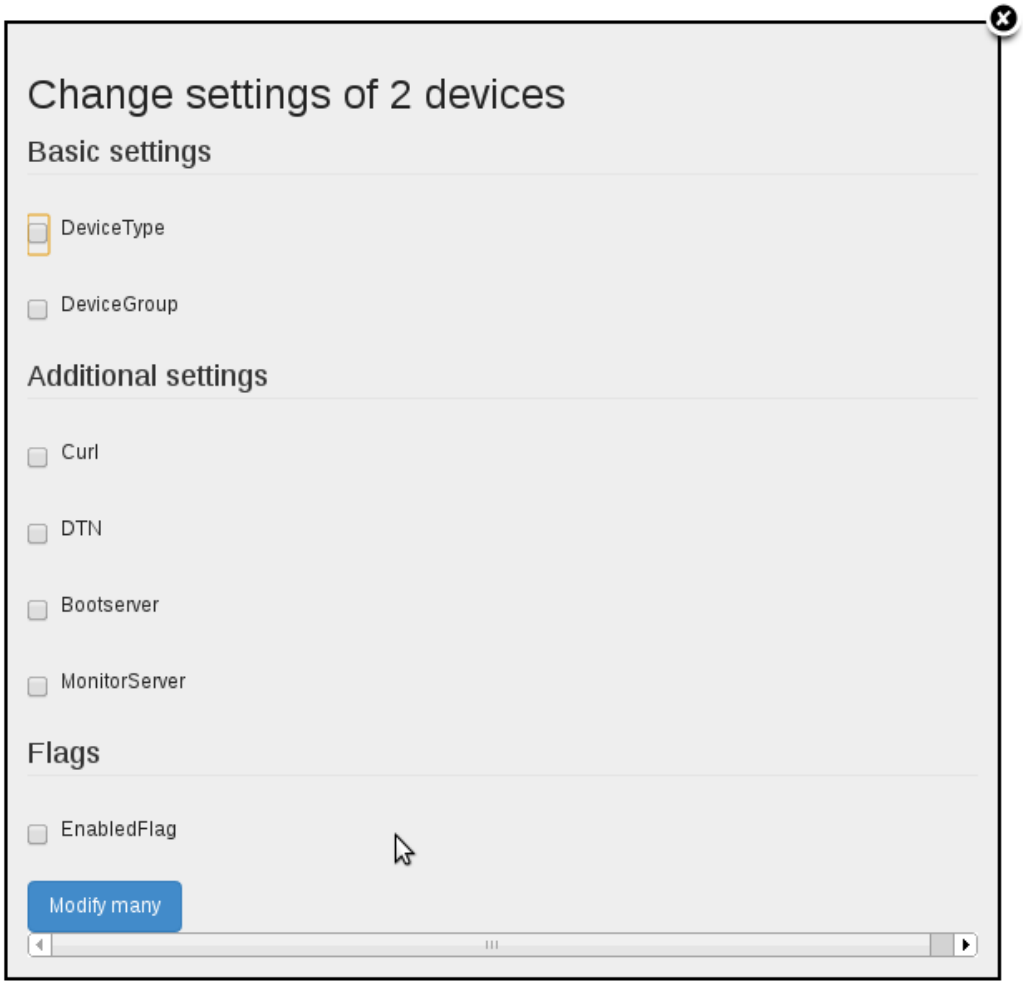
The main device tree view provides bulk modification and deletion of devices. To use this feature at first pick the devices with „sel“ checkbox as shown on the image:



After at least one checkbox was chosen, two new buttons „modify selected“ and „delete selected“ appear on the top-right site of the main view. Select „modify selected“ for bulk modification or „delete selected“ for bulk deletion of chosen devices.

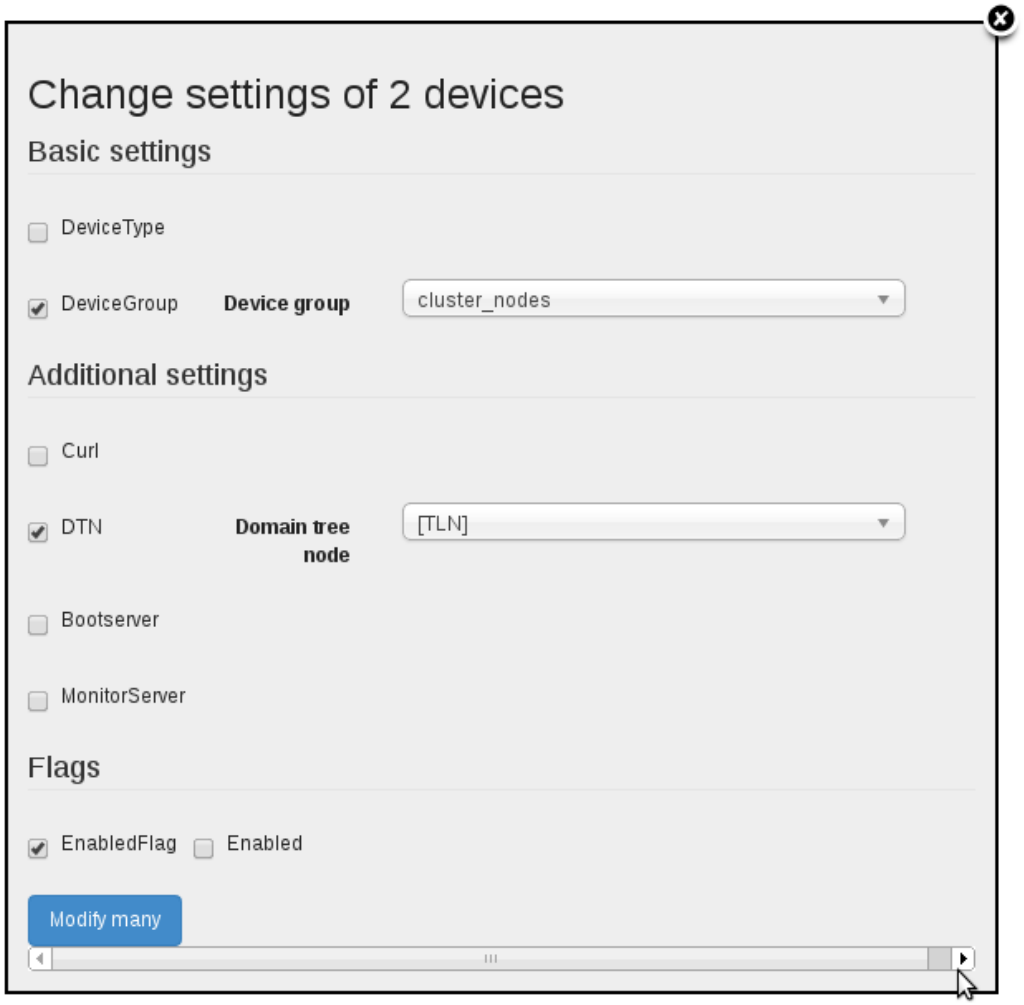


In case of modification it is possible to change many properties of selected device. Choose the necessary attributes for modification.

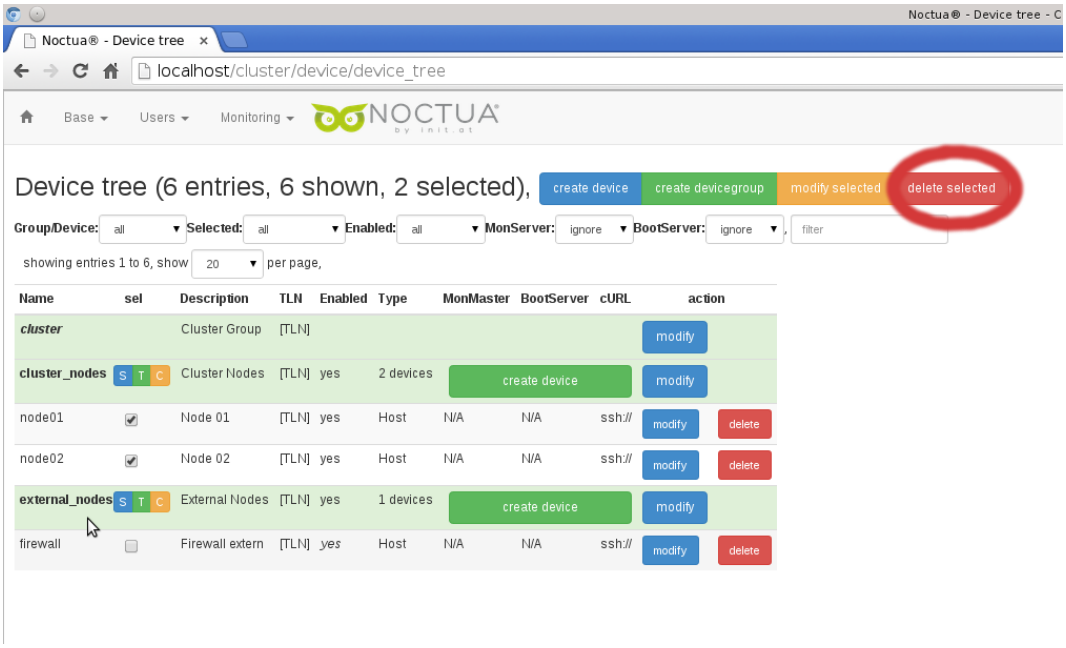


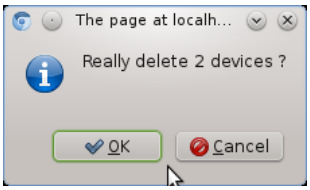
Once an attribute was selected, the list-box which arises to the right of the attribute make possible to input new value.

Important is also the „EnableFrag“-option. It gives the facility to enable or disable the status of many devices in one step. To complete the modification process, select „modify many“- button on the bottom of the window.

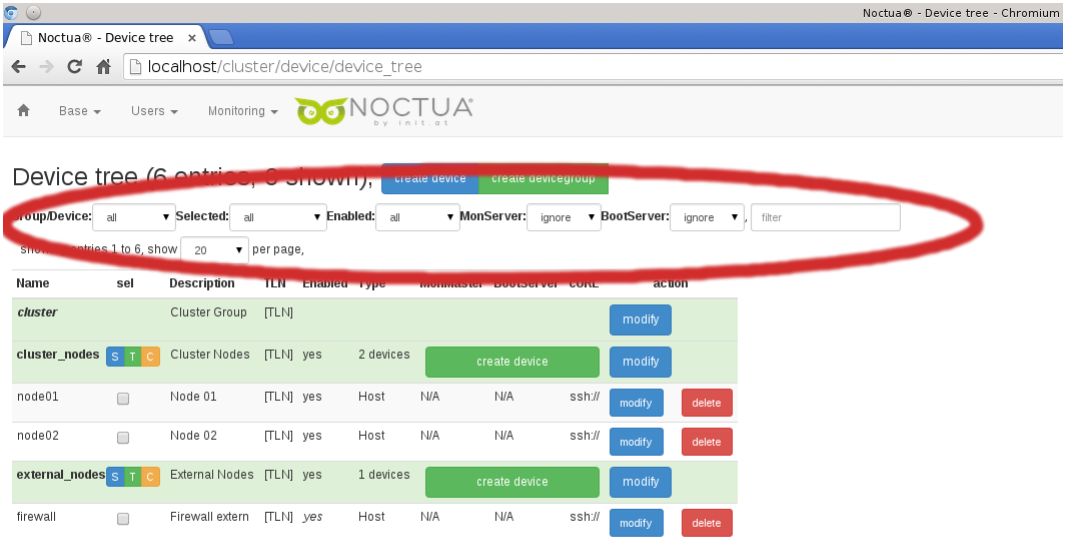


For bulk deletion enter the „delete selected“ - button on the main device tree view. In appeared window confirm you decision again.





For keeping the clear view to the device tree expecially in case of many monitored devicegroups and devices, use the filter option on the filter panel of the main view. **Group/Device** drop-down menu restricts the view to devicegroup or devices, **Selected** – to selected or unselected devices, **Enabled** – to enabled or disabled devices. If in Noctua more then one monitoring server or boot server are defined, the filter options „MonServer“, „BootServer“ reduce the device tree scheme to one of them. The filter text-box accepts normal text and regular expressions.



Network configuration

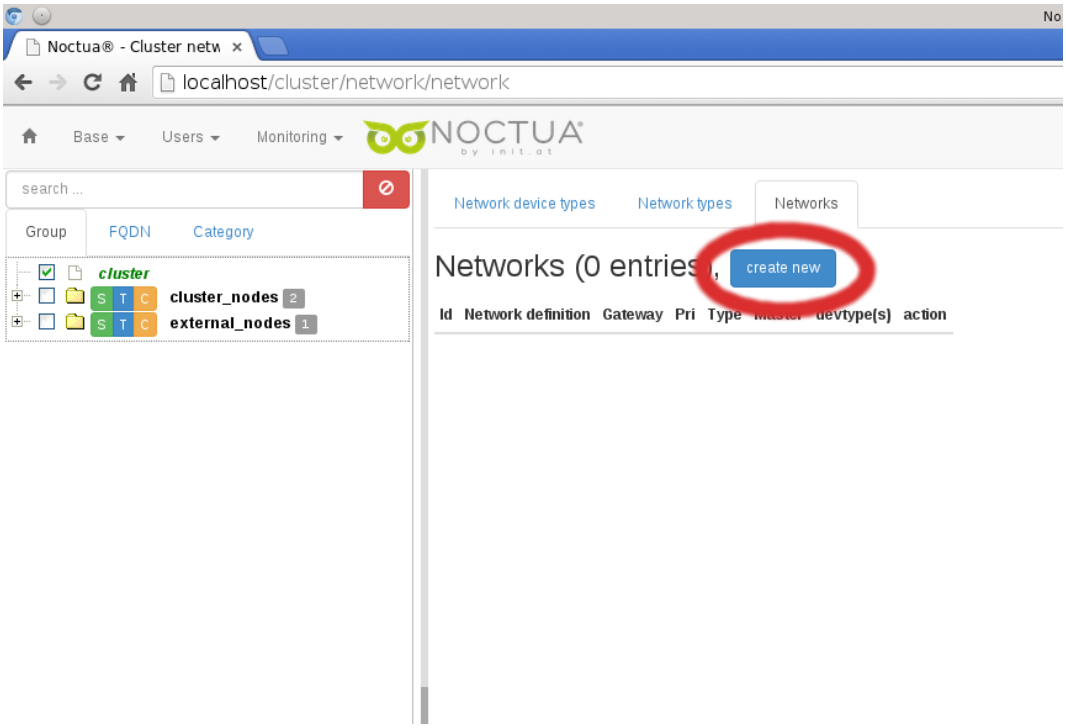
Networks

Before you start with network configuration for each device from monitoring software, it is necessary to define monitoring networks under Base → Networks. The main view from Networks area includes three tabs: „Network device types“, „Network types“ and „Networks“. All tree covers important points of network infrastructure from any monitoring landscape.

Two zones from Networks: „Network device types“ and „Network type“ are predefined from default installation of „Noctua“.

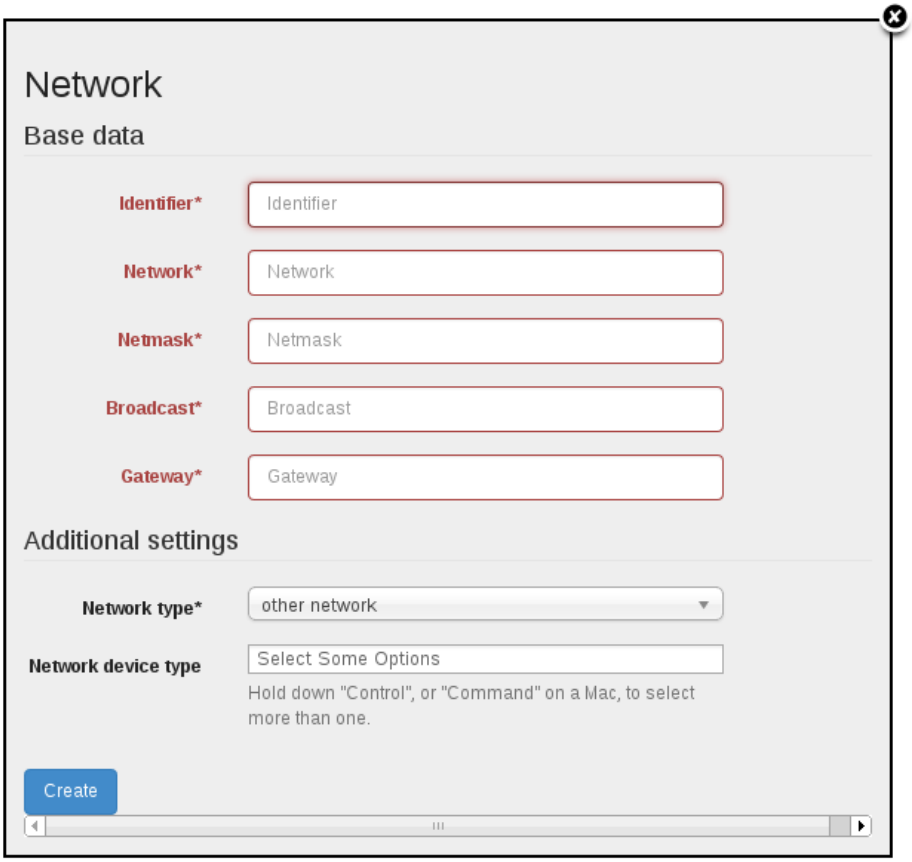
Your main field of application will be the last tab - „Networks“. You have to define networks before you can use these by devices.

For creation of new network select „create new“ button on the third tab „Networks“:

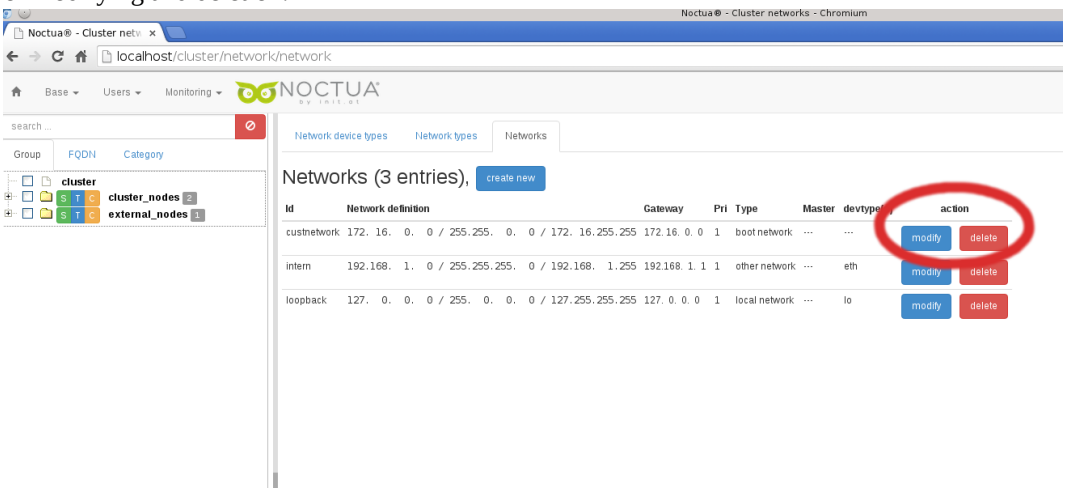


In appeared window define the parameters of new network. Base data includes the regular network's attributes. Additional settings includes the values, which are defined on the two another tabs of network area: „Network type“ and „Network device type“.

More detailed information about „Network type“ and „Network device type“ (PLACE HERE!)



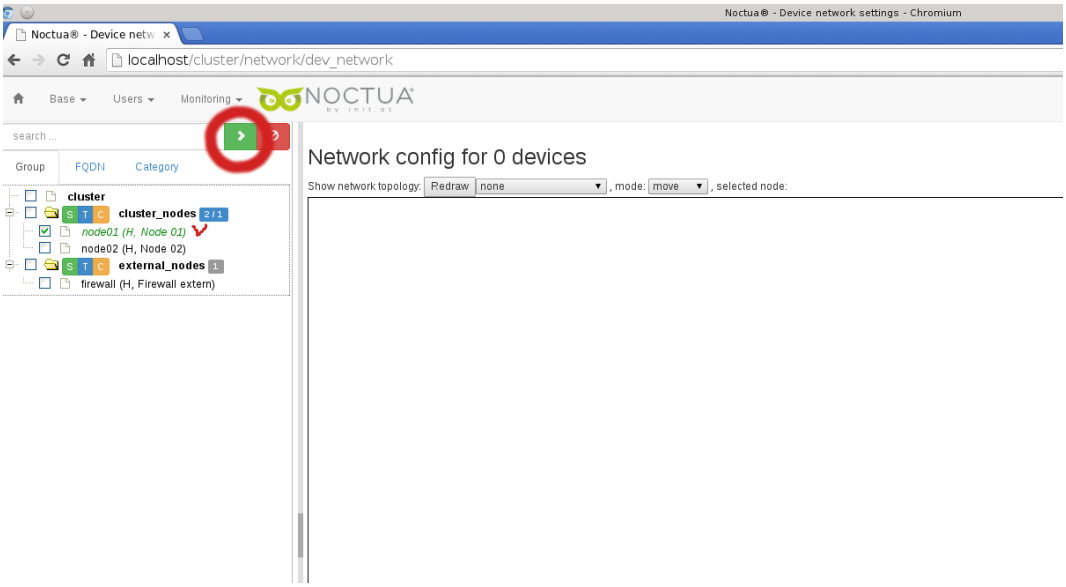
All already created networks you can see in the network list. For each of them there are two buttons for modifying and deletion:



Device Network

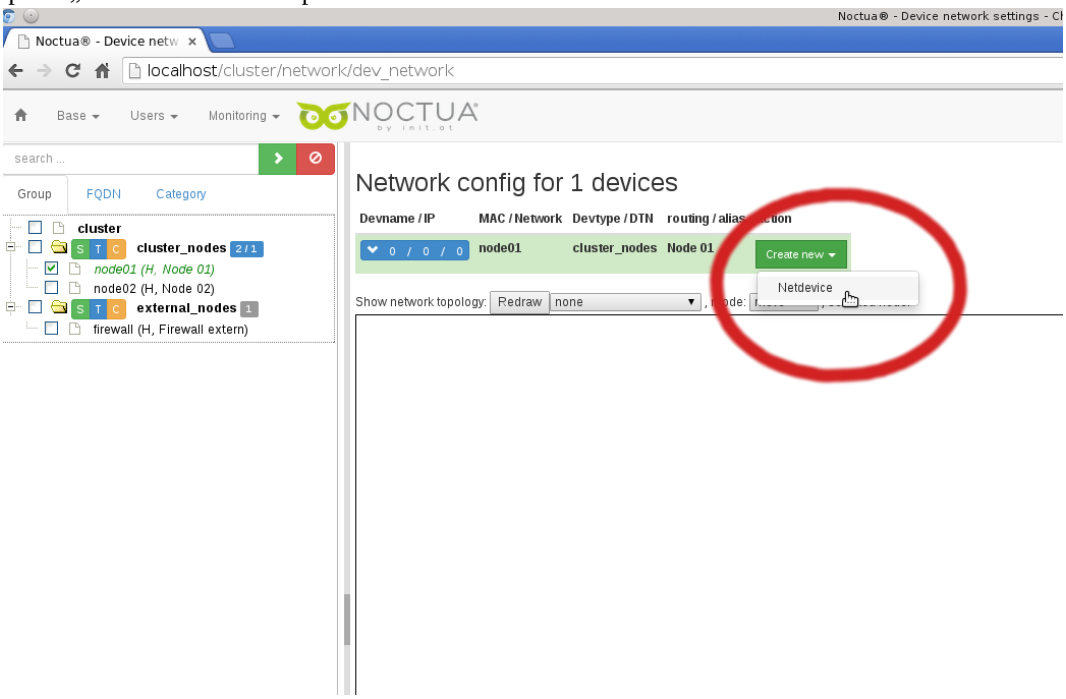
One important step in „Noctua“- configuration is the setup of interfaces for each device. The correct interface's setup of device provides the smoothly connection from Icinga-server to the monitoring client.

Select Base → Device Network to make interface-configuration from any device. In the displayed device tree on the left select devices, which you want to configure, then click the „green arrow“-button (see image) to refresh the view.

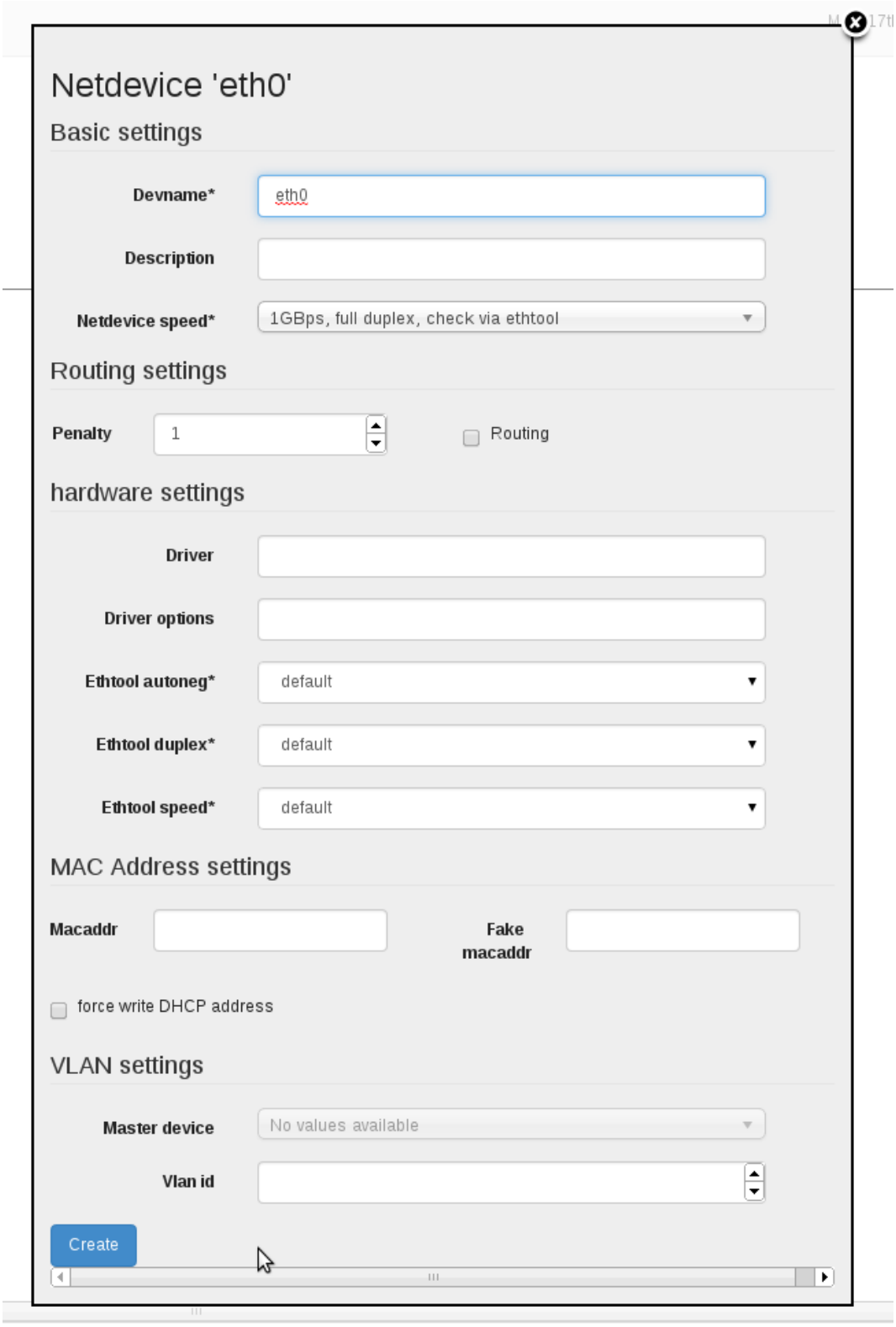


Operation of „green arrow“ - button refaces present network configuration of picked device on the right. In our case there are no configuration for device, therefore the network config setup is blank.

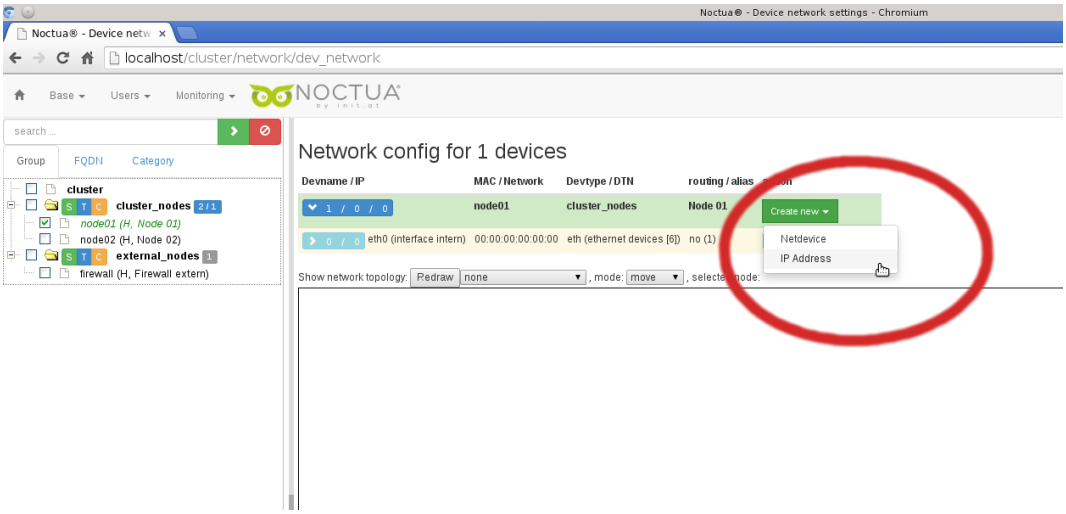
To define new interface, means netdevice in the clustersoftware, select „Create new“ button, then the option „Netdevice“ from drop-drown menu.



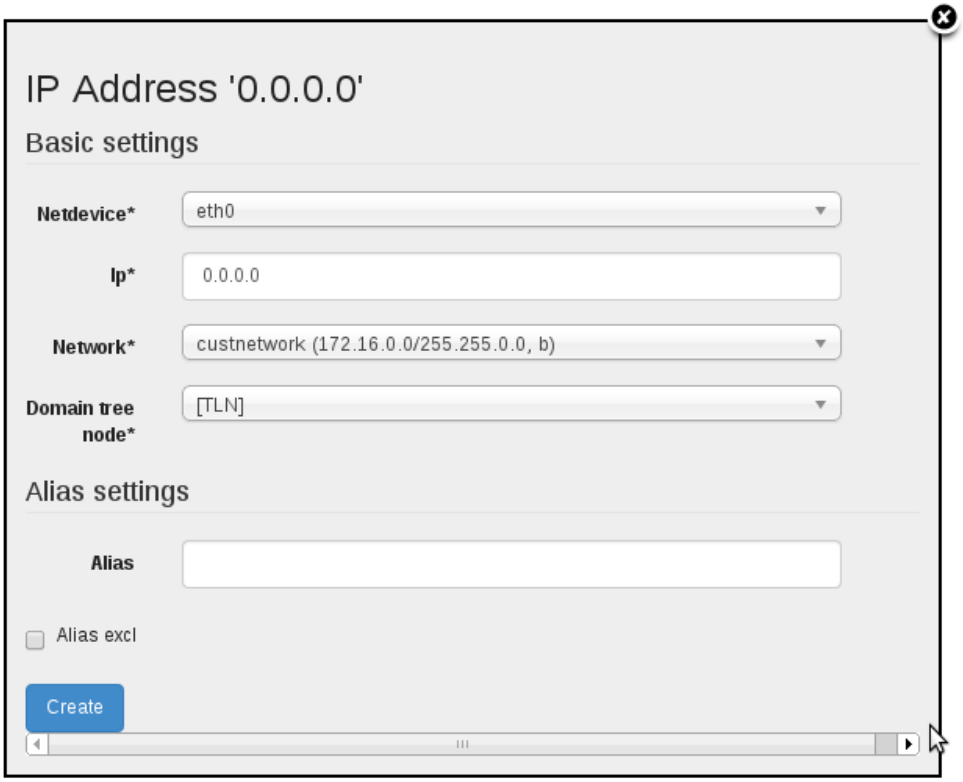
In appeared window set all necessary options ("*"-required options):



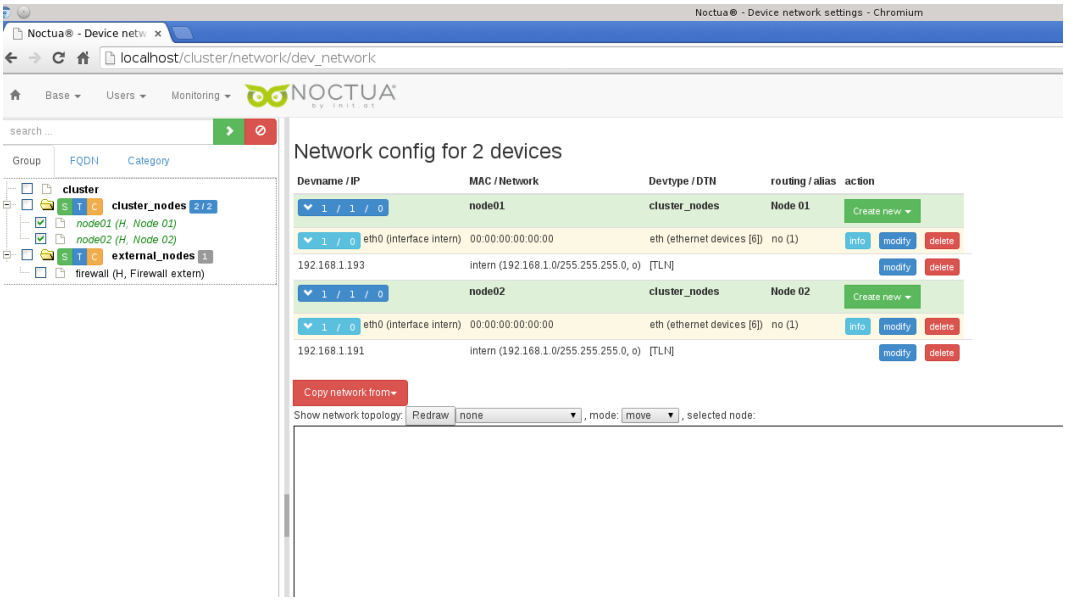
To assign the IP address for new created netdevice, choose in under „Create new“-button drop-down menu the option „IP Address“:



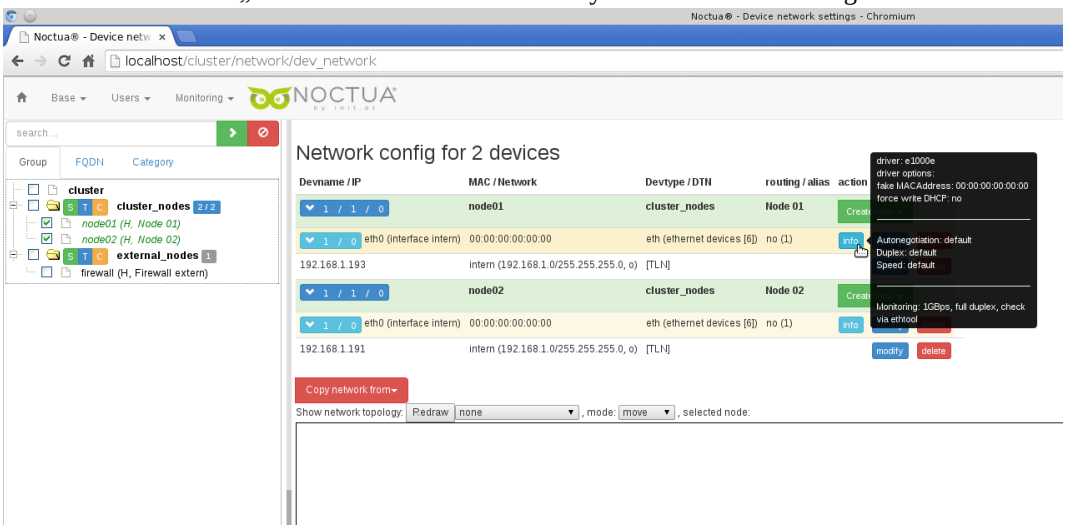
In new released window define the IP address and network of netdevice:



The filled network configuration for some devices can be presented like shown on the following image. For each netdevice and IP-address there are two buttons for modifying and deletion.



You can also use the „info“ button, which activation presents the netdevice's main information. Just move the cursor to the „info“- button to see the summary of the netdevice settings.



Virtual routing

The third step in „Noctua“ - connectivity configuration is specification of the virtual routing. Why virtual? The routing, which will be defined in Noctua must not to be consistent with the physical routing of monitoring landscape. The administrator of „Noctua“ has fully authority and responsibility to decide, how the routes would be constituted.

In simple case the virtual routing can reproduce the physical routing. But it is not a 'must-be' scenario of the routing image of Noctua.

The administrator of „Noctua“ has following rights to change the virtual routing against physical routing:

- right to exclude some physical nodes from virtual routing, which are not interesting for the monitoring/clustering
- right to assign own penalty values for the routes so that the shortly route will be different route as in the physical network

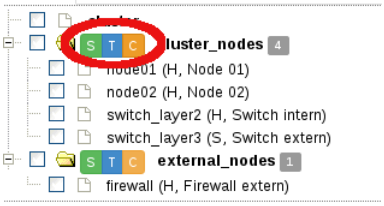
- right to assign some devices as routing devices, which don't have a routing functionality in the real network

The configuration of virtual routing happens under Base → Device network.

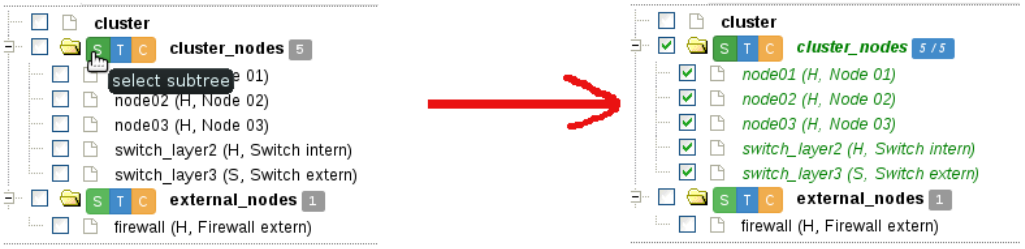
But before we start with the statement, how to configure virtual routing, we would clarify selection instruments in the device tree.

There are three selection buttons, which help to pick/unpick the plenty of devices (see on the image):

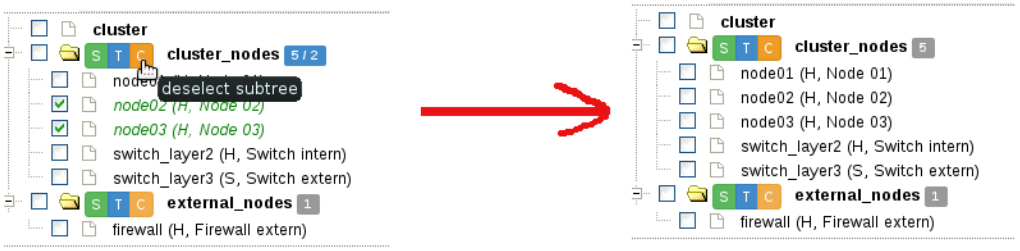
- the green S (for Select) Button: selects all devices and appropriate devicegroup under current subtree
- the blue T (for Toogle) Button: toggles to different devices in the same subtree
- the orange C (for „Clear“) Button: clears the previous selection in the current subtree



Functionality of „S“-Button:



Functionality of „C“-Button:

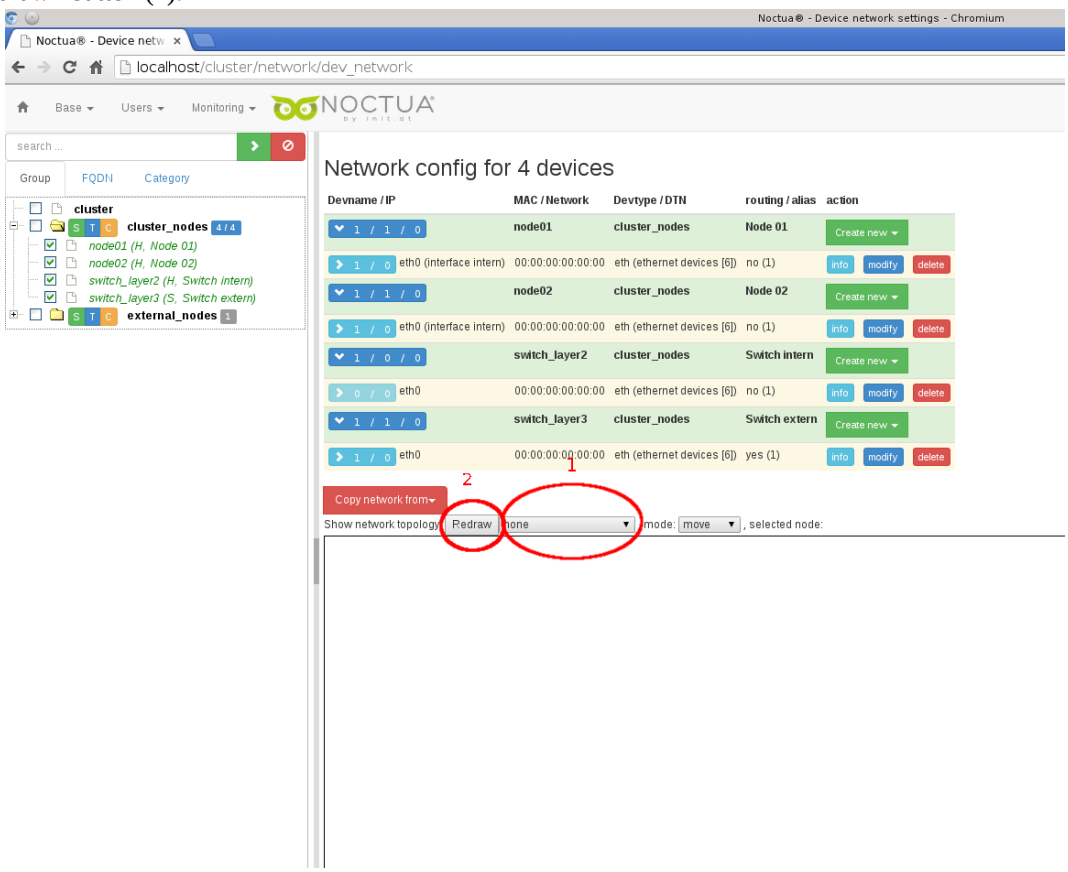


Functionality of „T“-Button:

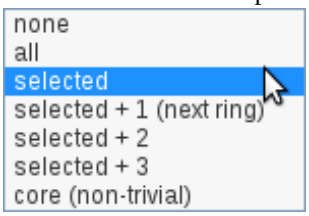


Now we can start with explanation about the main principles and technologies of the virtual routing configuration.

In our network we have defined four nodes, which are not bonded together. It can be demonstrated on the graphic facility of the software. For it select under Base → Device Network the required nodes, enter on the draw-panel in the drop-down menu (1) one selection-option, then click the „Redraw“-button (2).

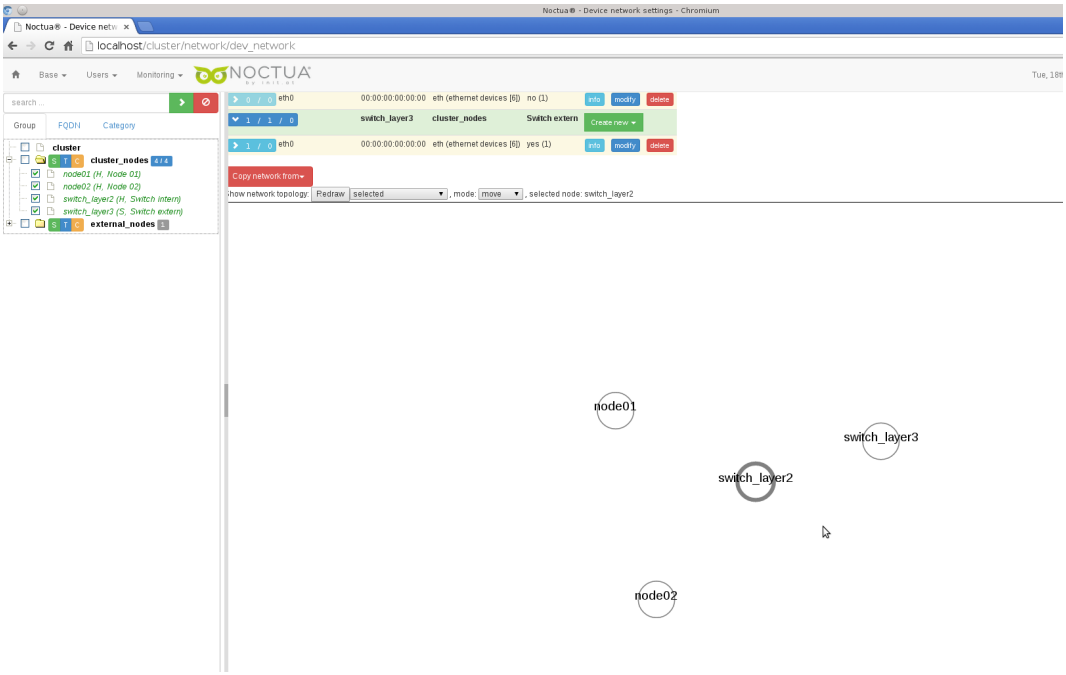


Here are the selection options of the draw facility:

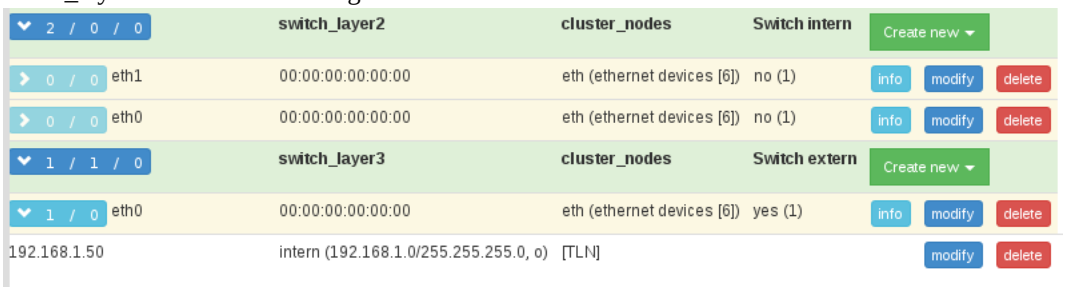


- all : all nodes from monitoring/clustering landscape;
- selected : only on the device tree selected nodes;
- selected + 1/2/3 : only on the device tree selected nodes and next 1/2/3 hops;
- core: only core-nodes (which have more then one route-connection) have been drawn.

Activation of the "Redraw"-button draws the nodes, which are not joined. In this case we have to define the network routing for these nodes. Only after the routing-network exists, the monitoring checks can be executed.



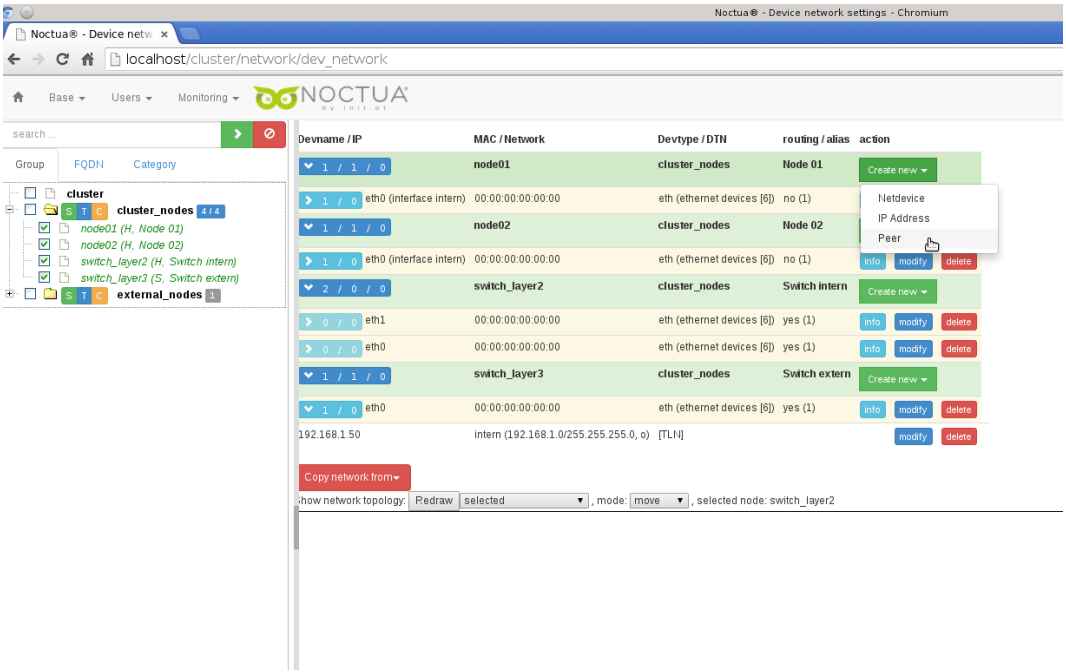
Before you can create some virtual routes from one devices to another, you have to decide, which hosts would be assigned as routable device. In our virtual network we decided to mark as routable two devices: switch_layer2 and switch_layer3. Both of them got the netdevices, but only the switch_layer3 netdevice was assigned with IP-address.



To flag the netdevice as routable device select the Routing – checkbox by creation/modification of the netdevice.

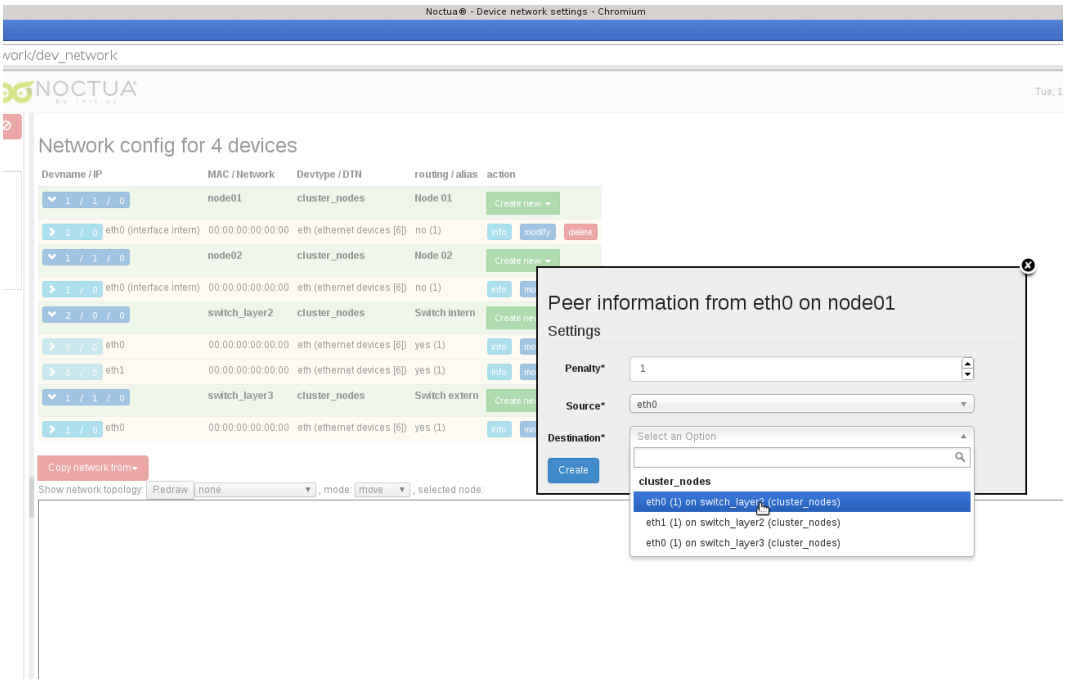


If there are some netdevices im system, which are defined as routing devices, new menu option „Peer“ appears in the 'Create new' drop-down menu on the main „Device Network“ view. Select this option to bind netdevice to the next peer.

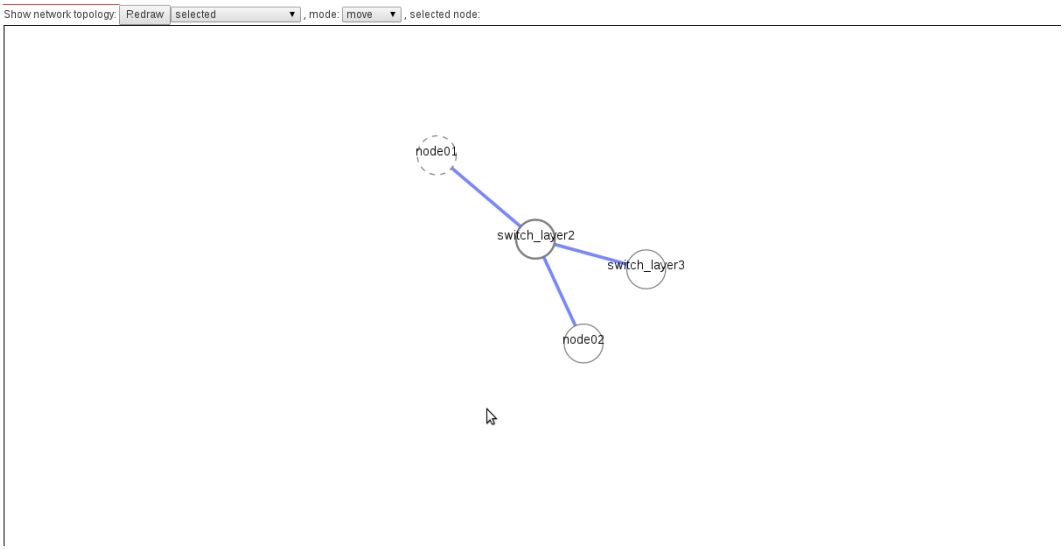


In arisen window define the penalty value and destination of the new created route. As penalty value you can use the „1“ value, if there are not redundant routes in the virtual network. If between two nodes many routes exist, with penalty value you can manage the priority of the route (the smaller value the higher priority).

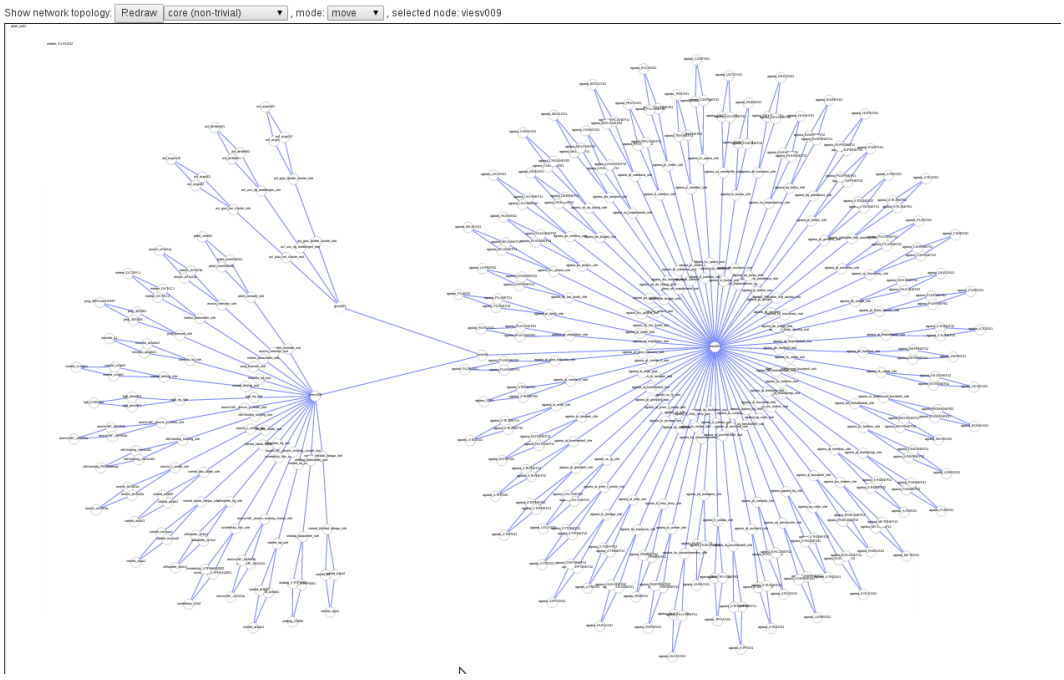
In destination drop-down menu all routable netdevices has been listed. Select the next required peer for the route and confirm with Create-button.



For example the virtual network in our case can be presented as shown on the following image:



The more complex virtual network can look like following example:



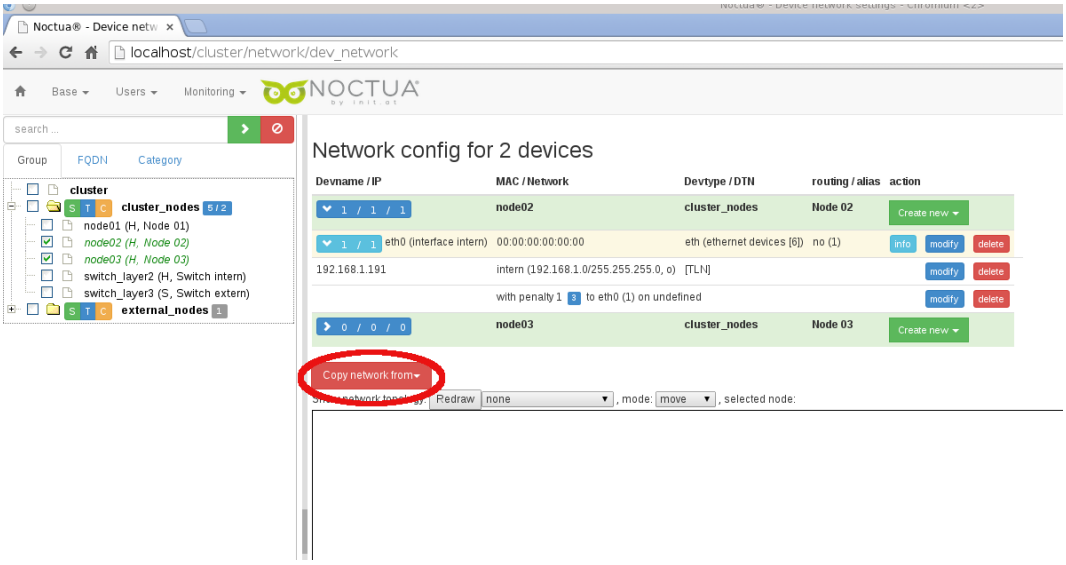
There are one more facility in the software, which can make the network configuration more easier: „Copy network from“- button on the main „Device Network“ view.

For example you have many nodes, which have the identical network configuration: the same network (network mask, broadcast, gateway, domain), the same next peer. For example, it can be the many nodes from one cluster, which connected to the one switch.

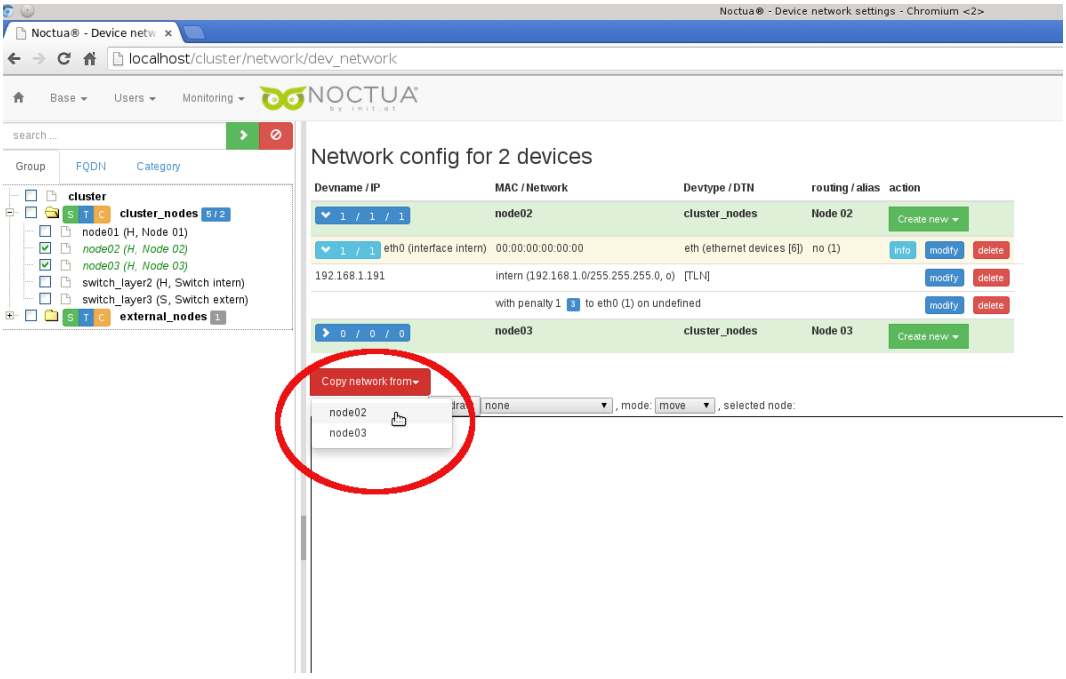
With this facility you can configure network and routing from one node and copy this settings to another 50, 100 or etc. nodes. The IP-address of the nodes would be incremented.

In our example: we have already configured the node02 and have to copy the network setting to blank node03.

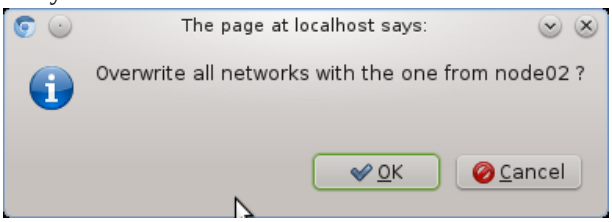
For using this feature select the nodes in the device tree: the fully defined node and blank nodes. Then click the red „Copy network from“ - button as shown on the image.



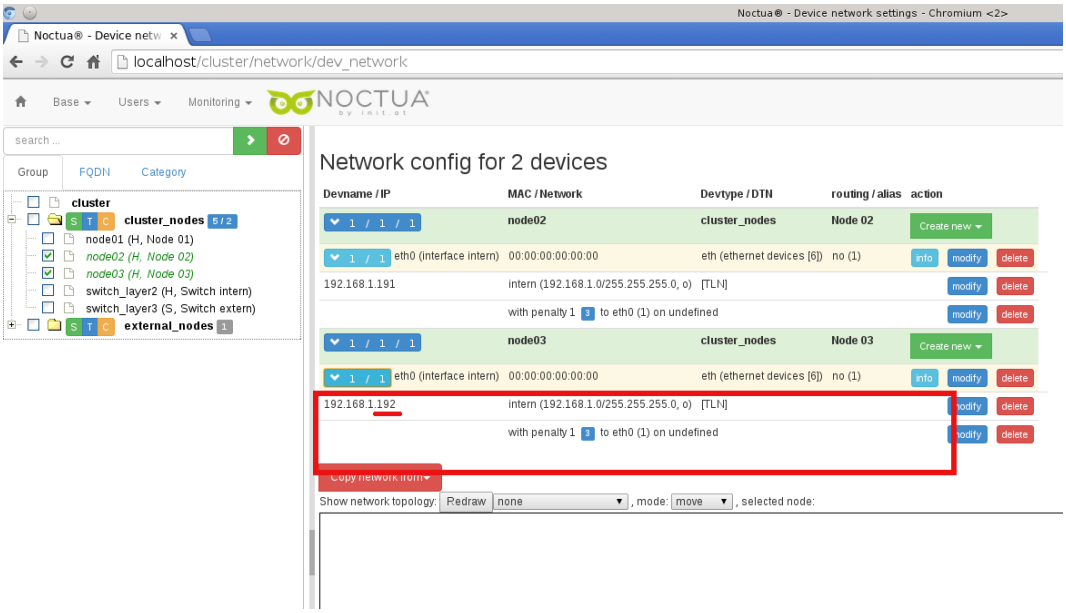
The system will ask you, which node has to be the initial point of the copy process. In our case it is the node02.



In arisen window confirm your decision.



The network settings have been copied one-to-one from node02 to node03. Thereby the IP-address of the node03 was incremented.



Once the virtual network has been completed, you can start to define monitoring checks and assign these to the devices.

Configuration setup

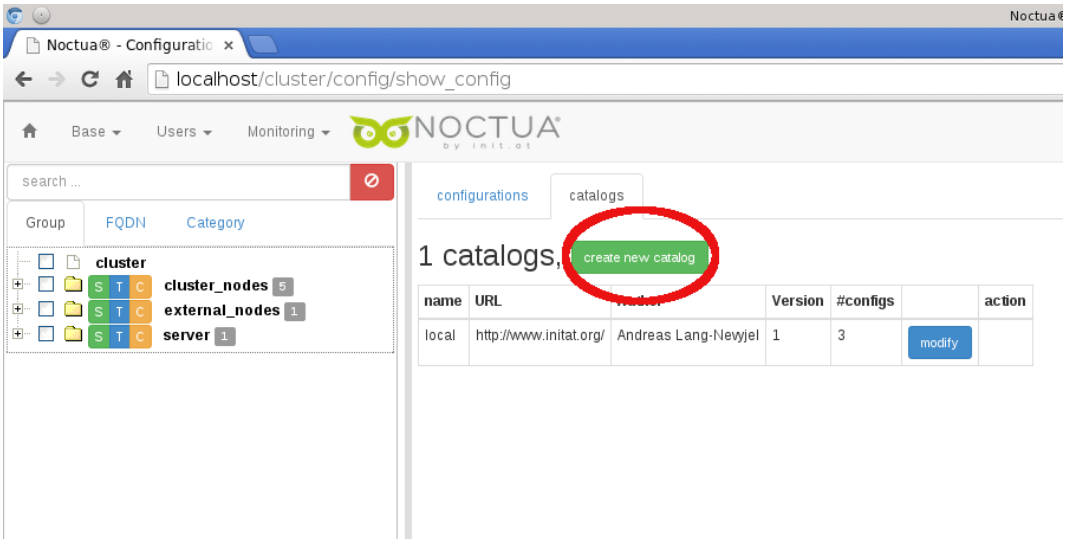
Configuration setup can be execute in two steps:

- adding the general checks to the monitoring software
- assigning the configuration checks to the devices

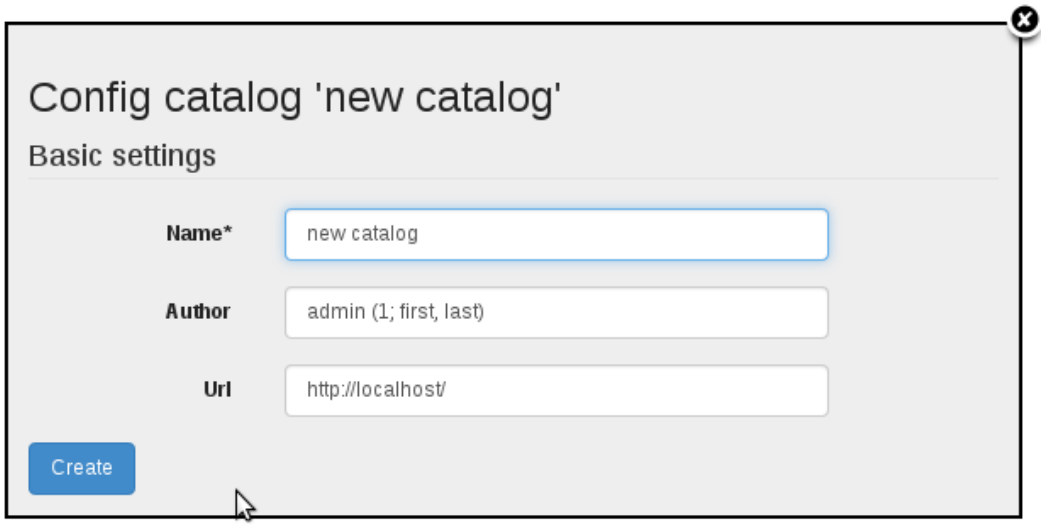
General checks configuration

For manipulation with general configuration checks select menu option Base -> Configurations. The main view contains to tabs: "configurations" and "catalogs". The first one - "configurations" - allows addition, modification, deletion of single configuration items. The second tab - "catalogs" - allows manipulation of the configuration catalogs. By default installation of monitoring/clustering software there is only one predefined catalog - "local". You can create your own catalog or upload to your server the catalogs, which will be published on www.initat.org. The consolidation of many configuration checks to one configuration catalog makes easier the handling with configuration.

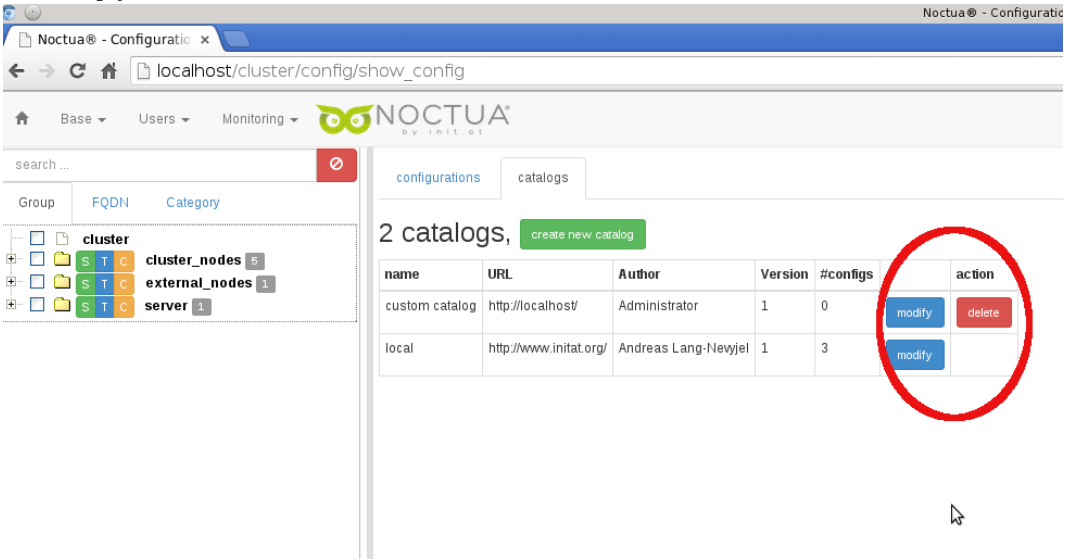
To create new configuration catalog switch to "catalog"-tab and enter "create new catalog" - button like shown on following image.



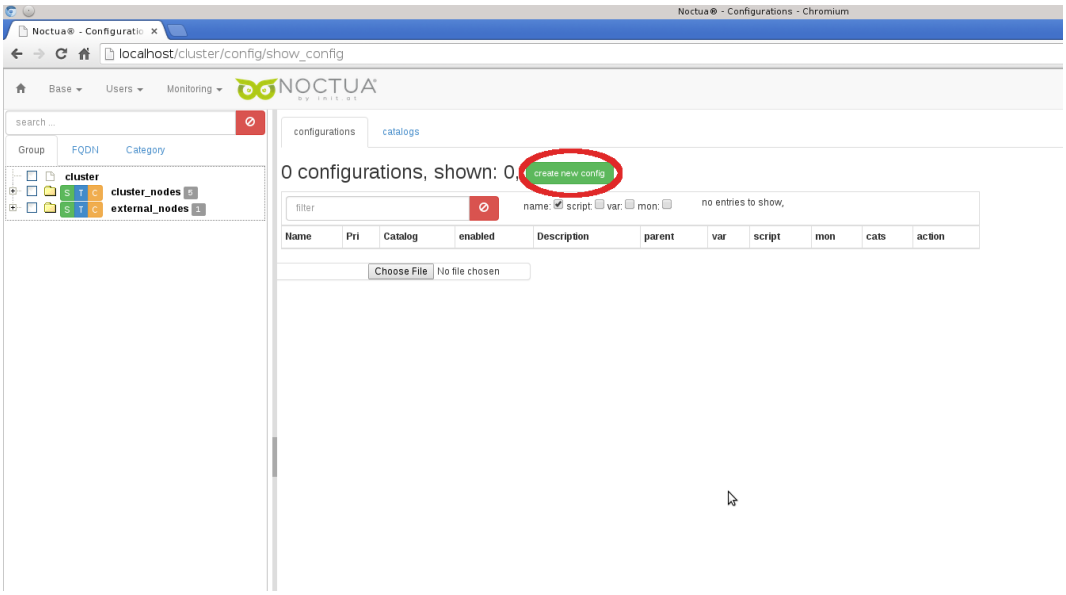
In appeared window define name, description (optional) and url (optional) of new created catalog. Once catalog was created, it can be filled with single configuration checks (see later in this chapter).



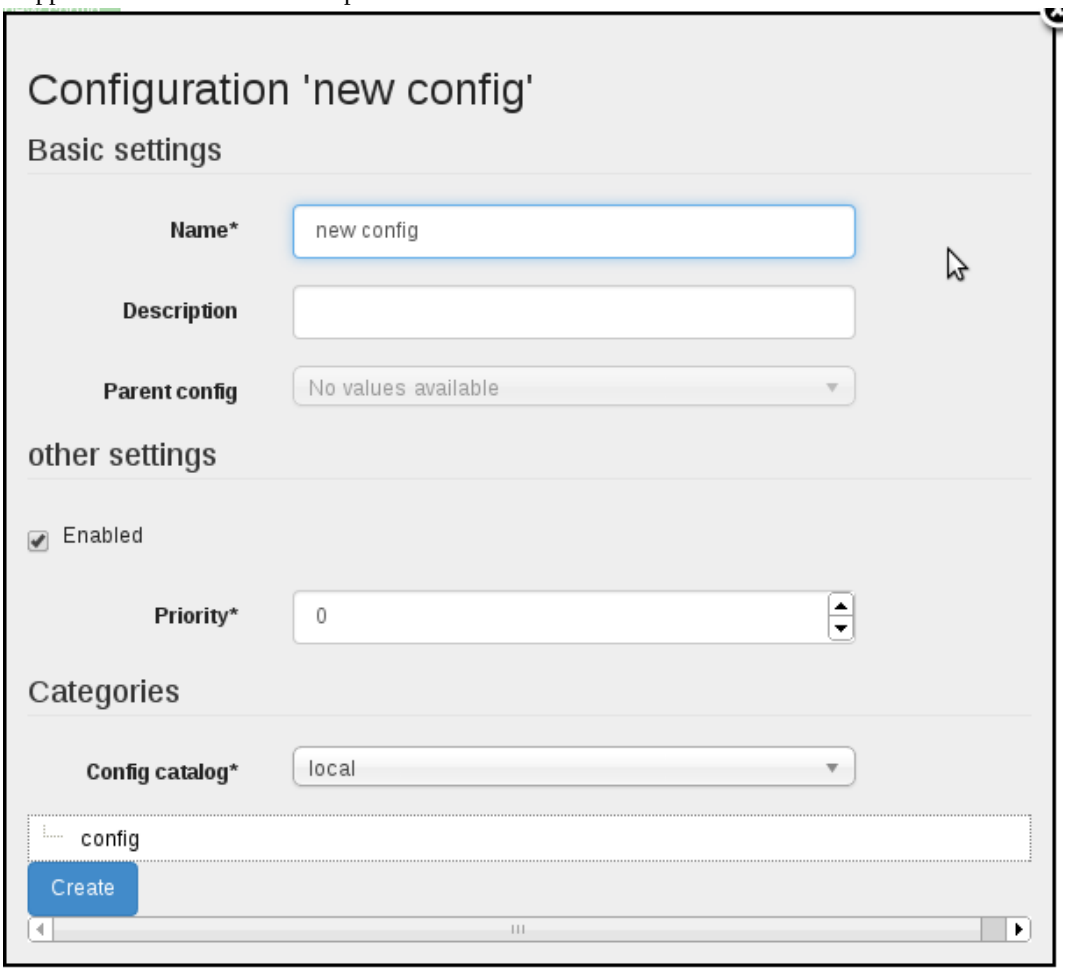
For new created catalog there are modify/delete options. If configuration catalog includes at least one configuration checks, then delete - button is blinded out. It is possible to delete any catalog, only if it is empty.



For manipulation with single checks switch to "configurations" tab. In our example the configuration of the monitoring task is empty. To create the new check, select the "create new config" - button on the main view.



In appeared window define the options of new check.

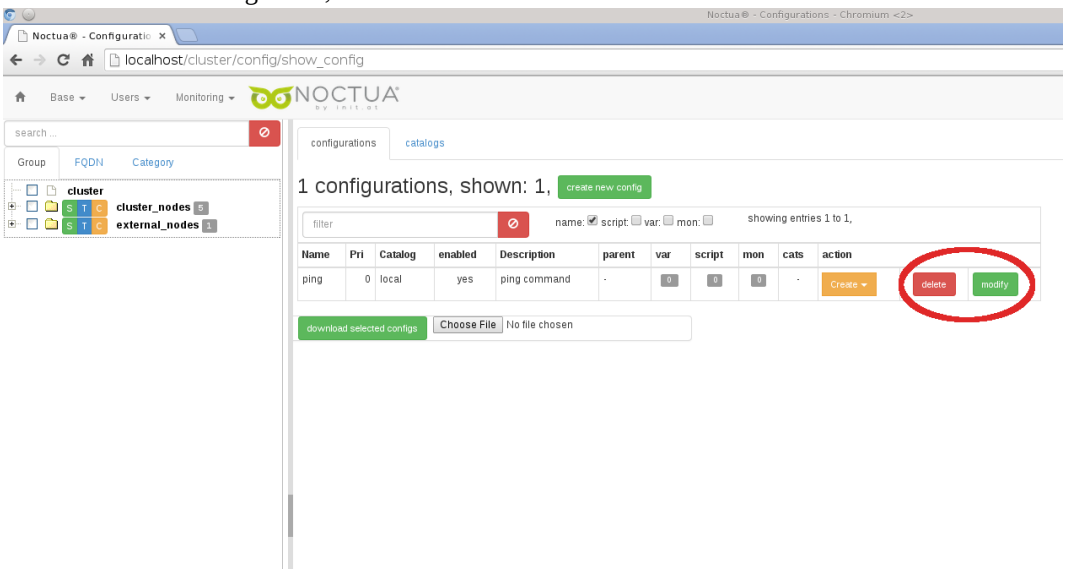


Available options of new configuration check:

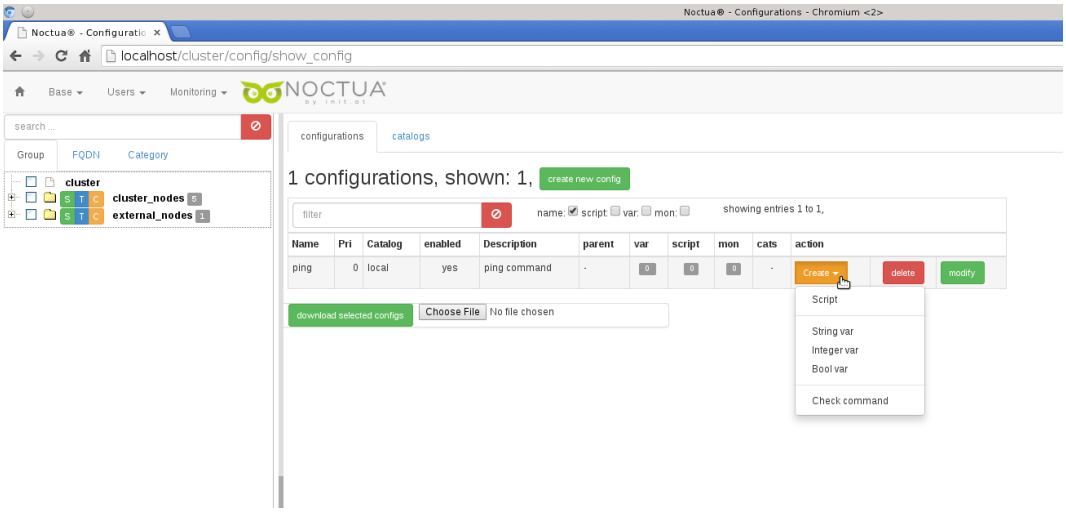
- Name: name of the new checks.

- Description: field for additional description for new check
- Parent config: gives the possibility to attach current config to the parent config. Practically it means if the child config will be attached to the device, the parent config is also applied to the same device.
- Enabled: checkbox enables/disables the configuration check.
- Priority: the useful setting for cluster installation. It has a big importance for the shell script-checks (explanation see later in the same chapter) in the clustersoftware. If you have many scripts, which have to be executed in precise order, assign the higher priority for the script, which has to be executed earlier.
- Config catalog: if there are many defined configuration catalogs, the configuration check can be ordered to determined one.

After the check is configurated, it can be modified and deleted.



There are some options to expand the config check. For listing all options click the Create pop-up-menu.

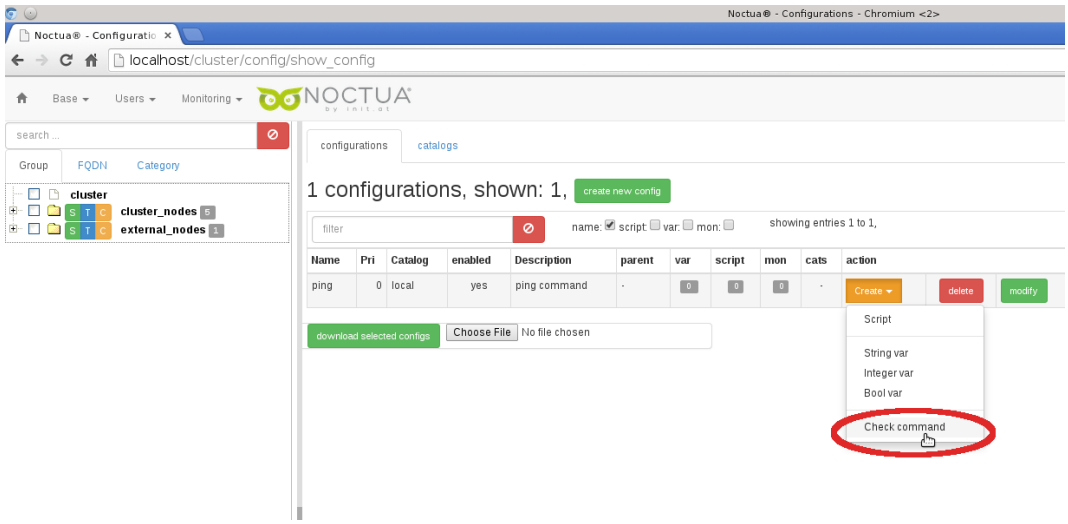


The expanded options are:

- Script: allows to attach the shell script to the check configuration

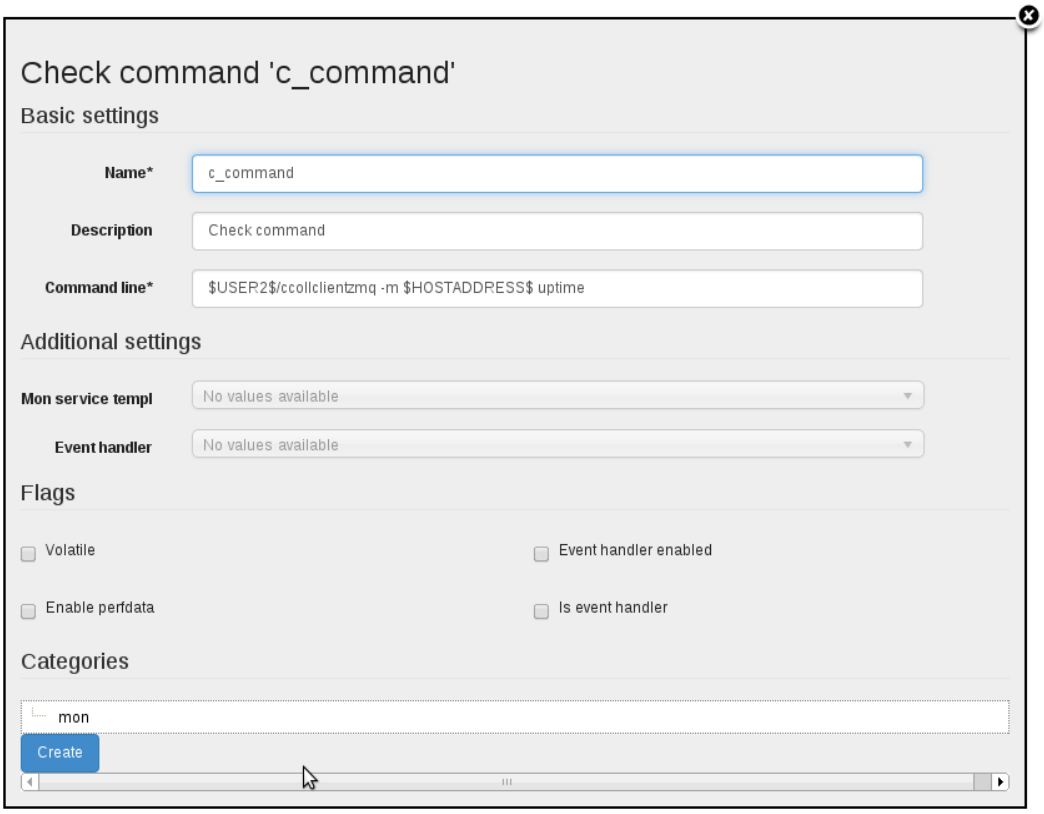
- String/Integer/Bool var: were be used in clustersoftware.
- Check command: allows to define the command, which will be executed on the destination host/device

Enter the "Check command" to arrive to the checks definition window.



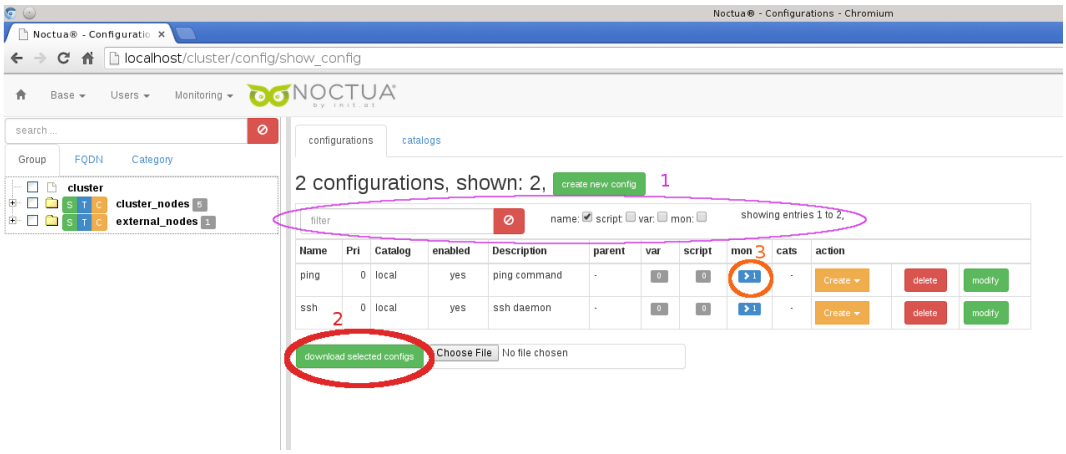
There are configuration options for the check command:

- Name: command name
- Description: description of the command
- Command line: the command, which will be executed on the device. How this command can be constructed, we will explain later.
- Additional settings:
  - Mon service templ: this option is available, it there are some defined monitoring service templates. For definition of monitoring server template, visit Monitoring-> Basic setup -> Monitoring service
  - Event handler: allows selection of configuration checks which have to be executed as event handler. The selected target check has to be marked as event handler earlier as it can be attached to the handled event.
- Flags:
  - Volatile: assigns the check as volatile process. For more information about volatile option see Icinga Documentation.
  - Enable perfdata: marks the command as one that deliver performance data to monitoring server back. The performance data can be used for draw performance graphs.
  - Event handler enabled: combines with next options "is event handler" the possibility to attach one event to the regular check.
  - Is event handler: marks the event as handler, which can be attached to another check as enabled handler. Look at icinga documentation for more information in which case the handler can be erased.

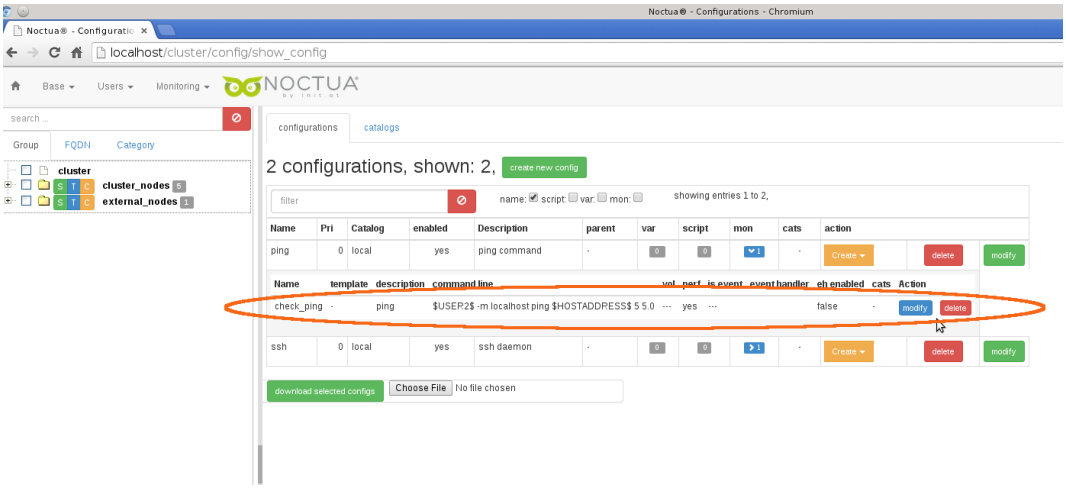


In configuration overview there are some features for facilitation of work:

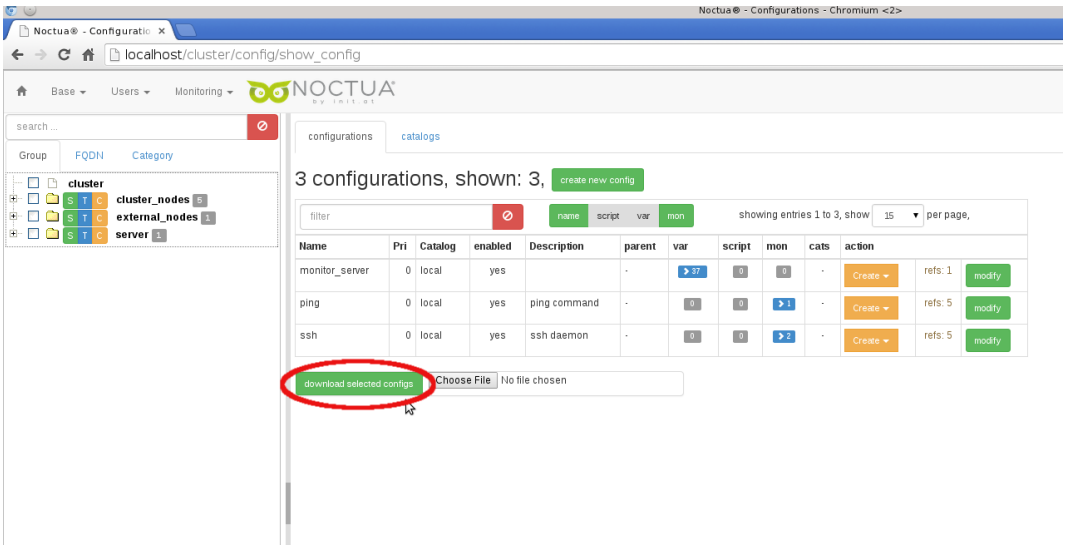
- filter panel (1): includes options for limiting of view to precise criteria.
- "download selected configs"-button(2): allows download of configuration to xml-file for following migration to another monitoring/clustering server.
- "blue"-arrow(3) for each configuration check: shows the detailed information of configuration check.



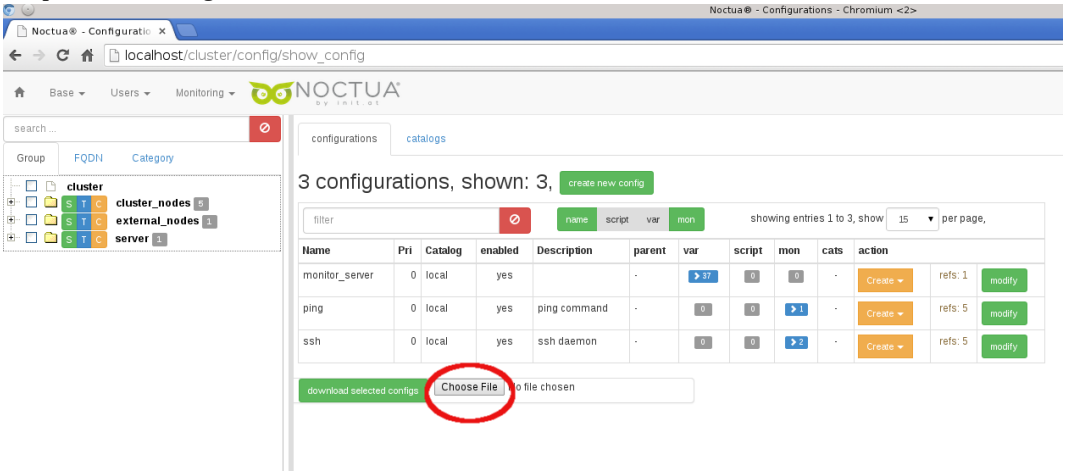
Actuation of "blue"-arrow for configuration check shows detailed information, also buttons for deleting and modification for this check.



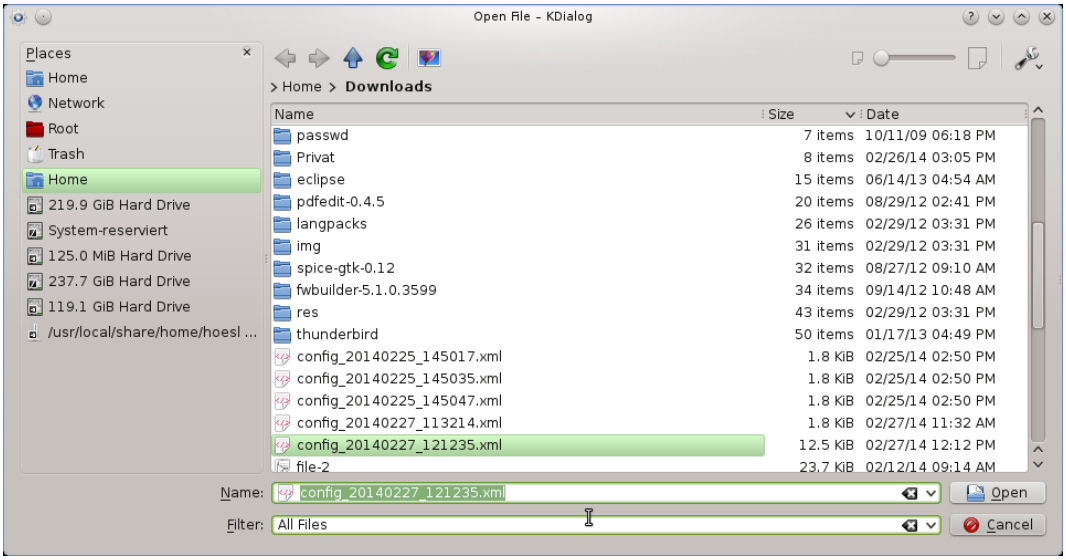
To migrate the configuration to another monitoring /clustering server, select the "Download selected configs" - button. The configuration will be downloaded in xml-format and can be transport to another server.



To upload the configuration file to the server, click the "Choose File" - button

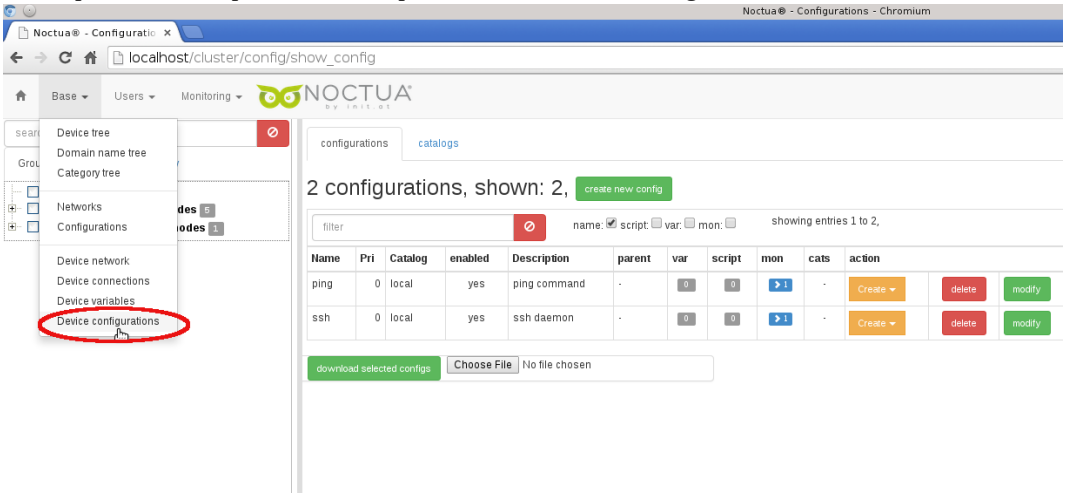


In appeared window select the necessary config-file and confirm your selection.



Device configuration

Once the checks were configurated, they can be assigned to the devices from monitoring/clustering landscape. For this step select menu option Base -> Device configurations.



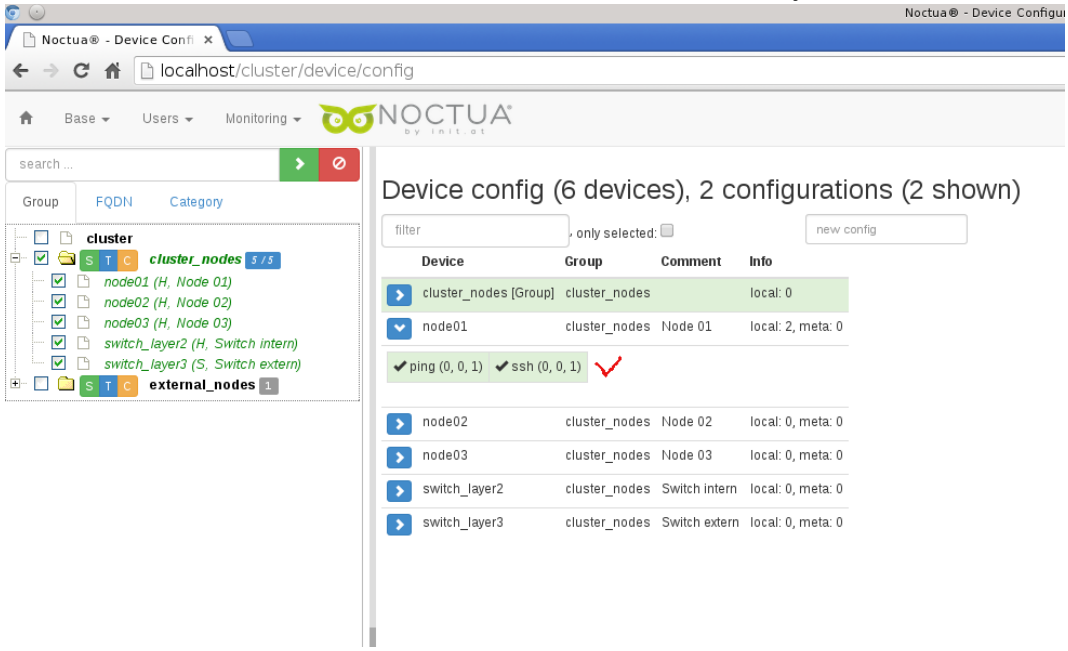
On the left site in device tree frame select the nodes or groups, which configuration you want to change. With click on "green"-arrow (like shown on the image) the device configuration view on the right site will be refreshed.



Expand the node's area with activation of blue arrow and select the checks, which have to be executed for this node.



After activation the field will be marked with different color and "check"-symbol on the left.



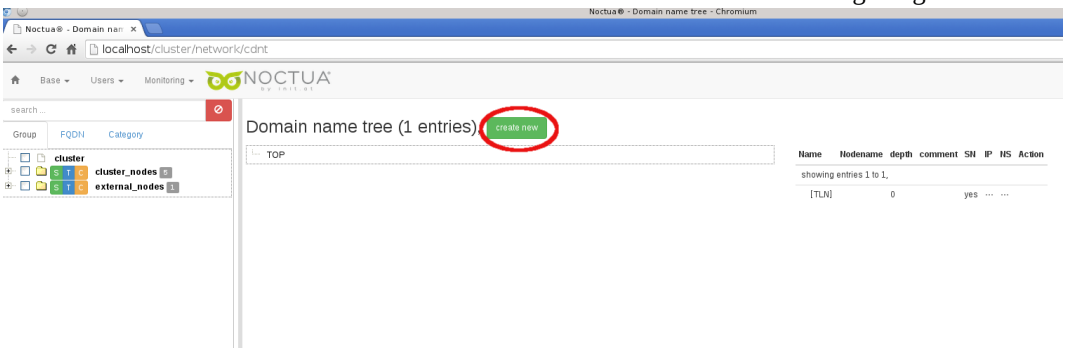
Domain name tree

"Domain name tree" menu option in "Noctua" webfrontend gives the possibility to create tree structure of the domains. This is an additional feature of the software, which helps to make some administration tasks easier. For example, the domains can be used for writing of /etc/hosts file or for discovery of the network settings with the help of domain name of host.

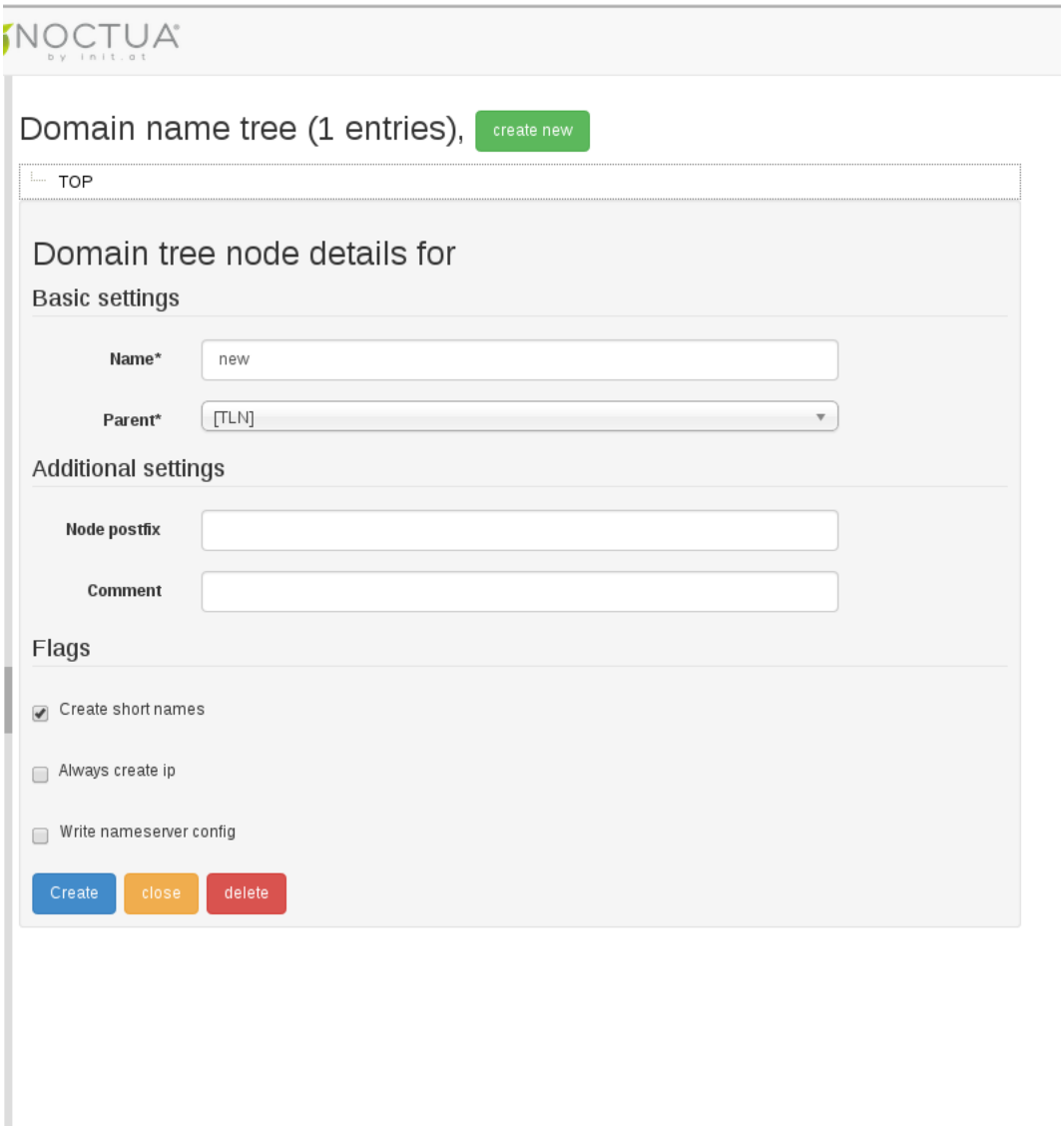
To reach the interface for domains view tree select menu point Base -> Domain name tree.



For creation new domain select "create new" - button like shown on the following image.



Define all necessary attributes of new created domain.



Here is the detailed explanation about details for new domain:

- Name: name of new domain. Give here the full name of domain. For example, *sales.init.at, main.server.company.com*. The software will split the name in level items and create different levels of domain tree.

- Parent: select the parent of the drop-down list. As default the main-parent is named as "TLN" ("Top level node"). Every next can be only descendant node of "TLN". Here is an examples for parent-node relation of domain server.company.com: "com" is the parent of "company", "company" is parent of "server" and so on.

- Additional settings:

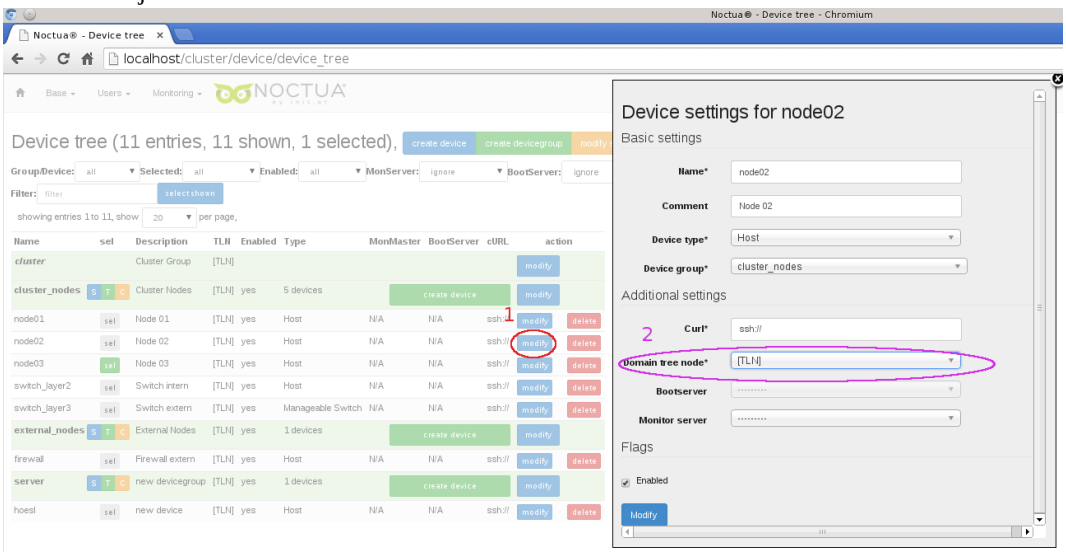
  - Node postfix: this field allows the possibility to add the prefix to the domain to distinguish two domains. For example, the "ipmi" can be added to "node.company.com" domain. Then device can be attached to node.company.com domain directly but ipmi-interface of this node can be attached to "ipmi.node.company.com" domain.

  - Comment: write any important information in this field.

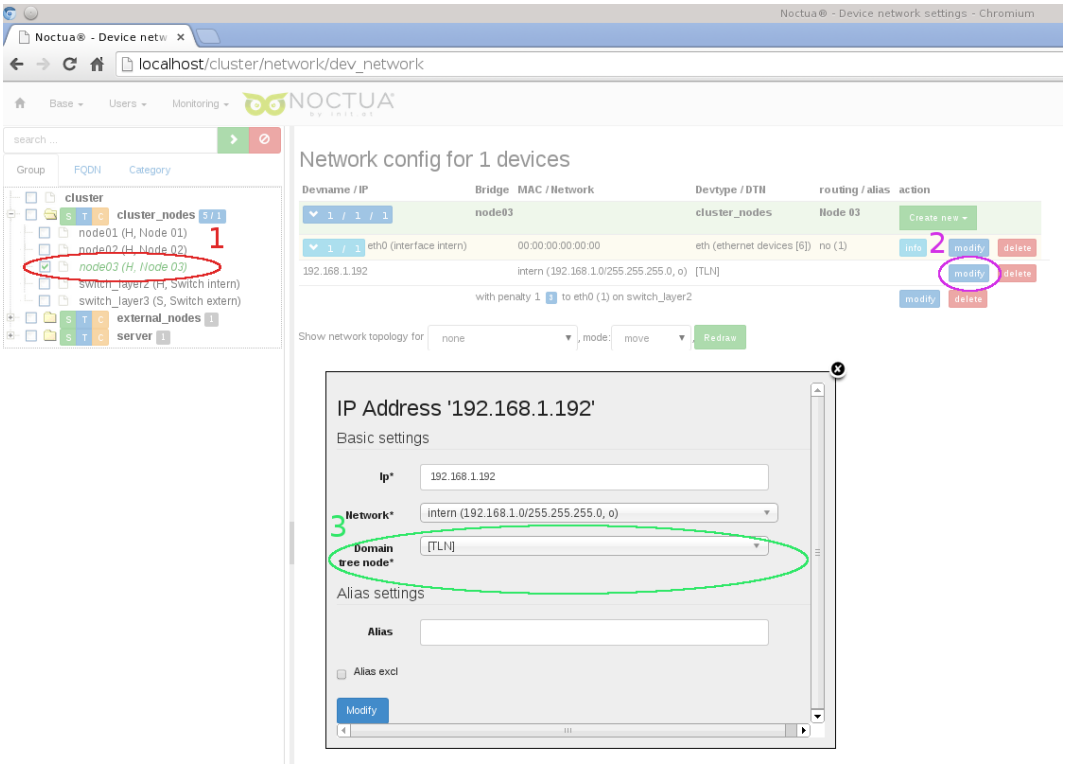
- Flags:

- Create short names: has primary importance by writing of /etc/hosts file. If this checkbox was activate, the short names will be used in /etc/hosts. For example, the row for node.company.com will be shown like following: "<ip-address> node.company.com node"
- Always create ip:
- Write nameserver config:

Let us see, how the domains can be assigned to the devices and ip-addresses. To apply the domain name to the device, select Base -> Device Tree menu option, then hit the "modify"-button(1) beside the target device. In "Device settings" - windows in "Domain tree node"-field(2) choose the domain name from drop-down list of domains. Finally actuate the "Modify"-button to save new attribute to the device-object.



You can also attached the domain name to the netdevice, strictly speaking to the ip-address of the netdevice. For this action at first select the device-object in the device tree frame (1). Then switch to Base -> Device Network, expand the netdevice information row and hit "modify"-button(2) beside the target ip-address of netdevice. In new appeared "Ip Address"- window give required domain name in the "Domain tree node" - field(3). Confirm the changes with click on "Modify" - button.



Category tree

Category tree feature of the software allows to collect some components of the software to one or more groups (here means categories). Categories in cluster/monitoring software have a importance to facilitate of handling with monitored/clustered devices and configurations.

Category tree can be created under "Base" -> "Category tree" menu option.



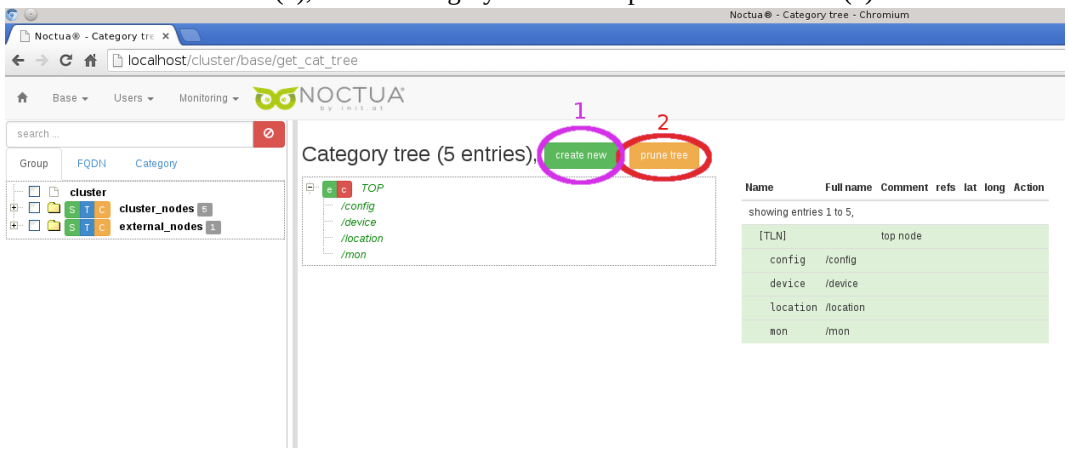
Default installation contains four main category groups: config, device, location and mon (from monitoring). These four can be expanded with custom categories. Here are some examples for possible categories fragmentation:

- "config"-category can include "node", "master" sub-categories, which accord with the configurations for nodes and master.

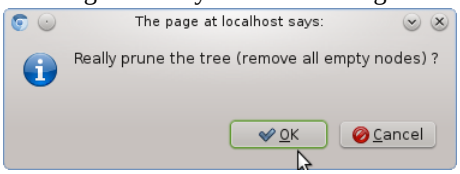
- "device"-category can contain "server", "switch", "node" sub-categories.

- "location"-category can include "Europe", "Asia" or "room01", "room02" segmentations.
- "mon"-category can be split to "disk", "ram", "cpu" and so on sub-categories.

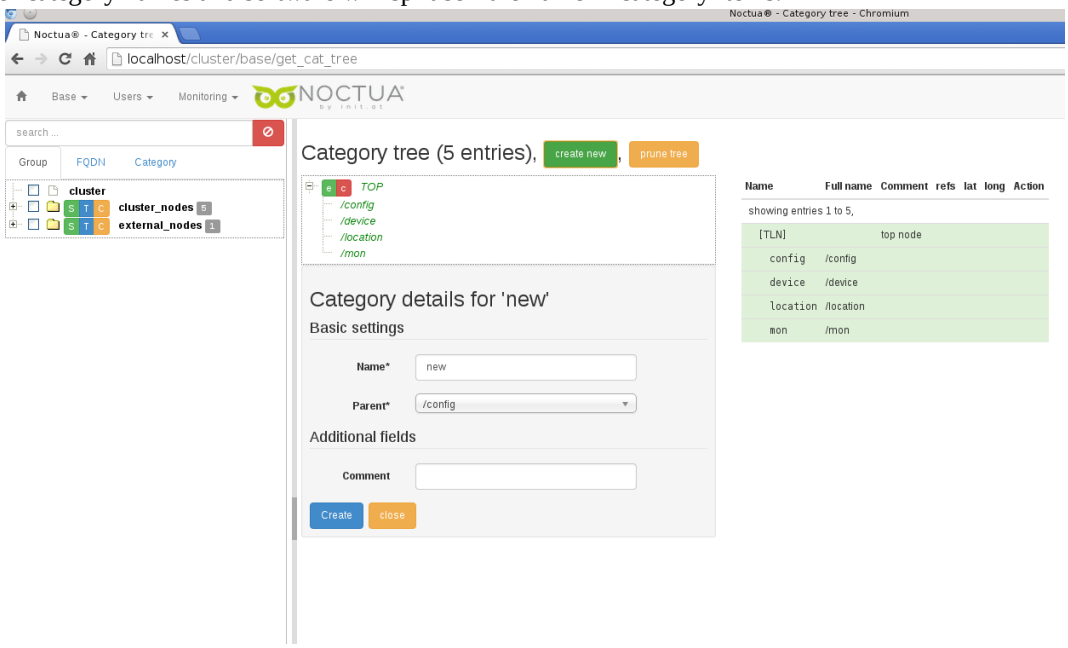
There are some tool kits, which can help in manipulation of categories: to create new category, select "create new" - button (1), to clear category tree select "prune tree" - button (2).



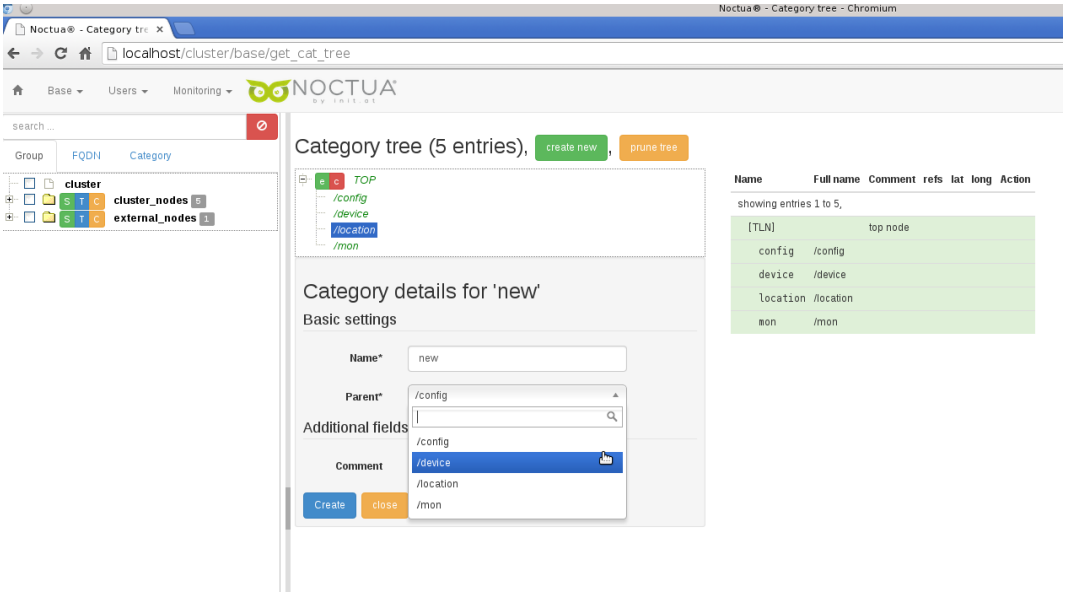
In event of tree's clearance confirm your decision in appeared window. So only categories, which are not assigned to any device or configuration, will be deleted.



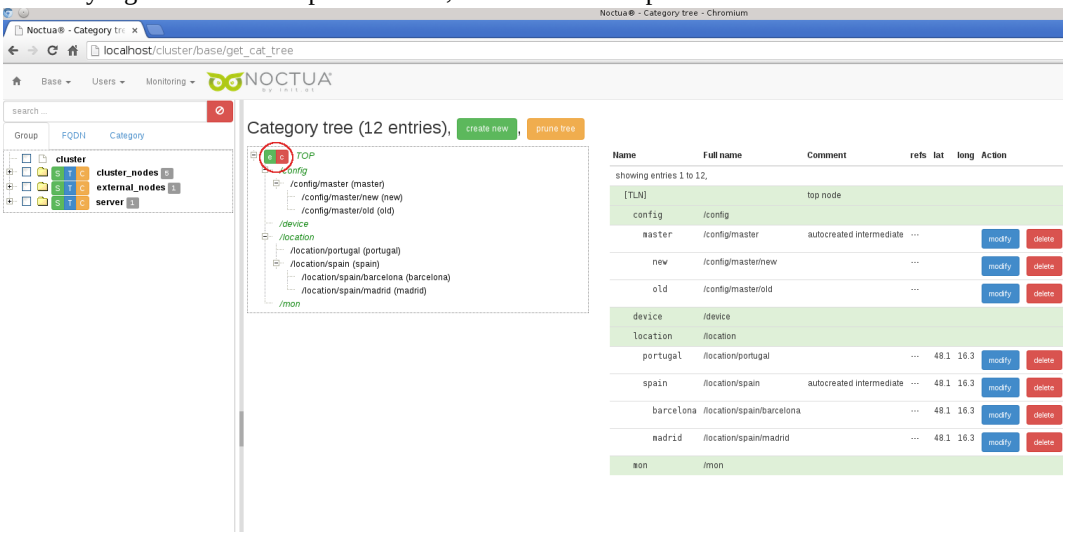
For creation there are two important attributes: Name and Parent. For Name use expression in following format: *name[/name1[/name2]]*. It means, for name value you can use any desired interlacing of category names and software will split self the name in category items.



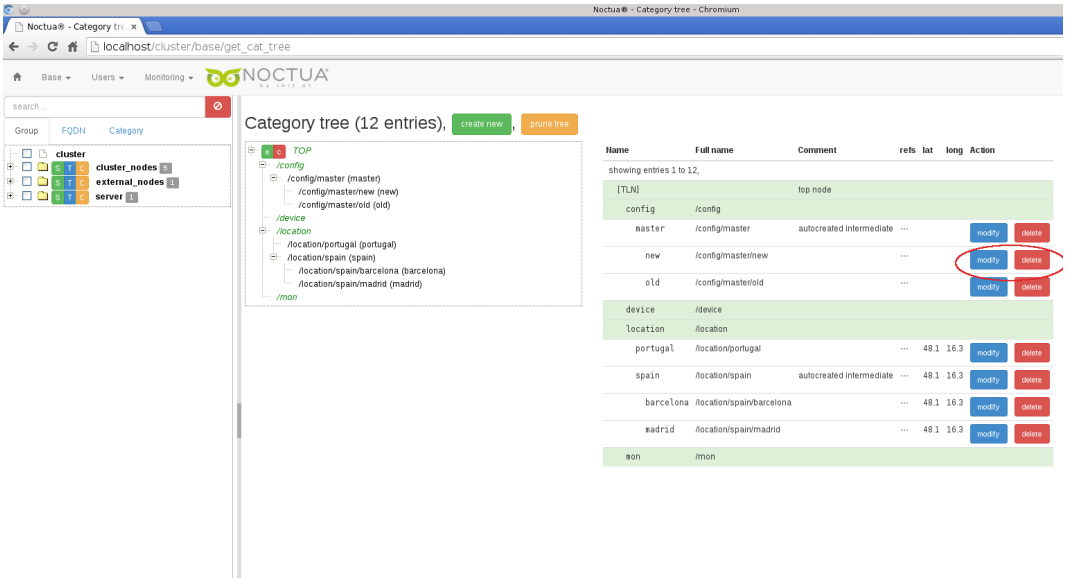
In "Parent" drop-down menu you can decide, which category has to be the parent of new created one.



The partially filled category tree can be presented like shown in the image. On the top of tree there are to keys: green "e" - for expand the tree, red "c" - for collapse the tree.

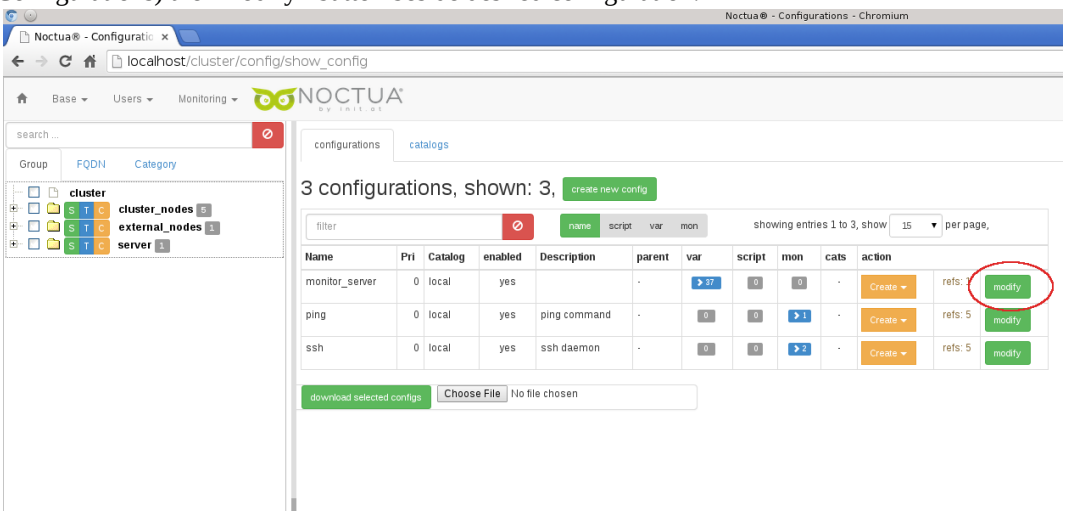


On the right site of category tree you can see detailed information of single categories. There are also buttons for quick modification or deletion for each category.

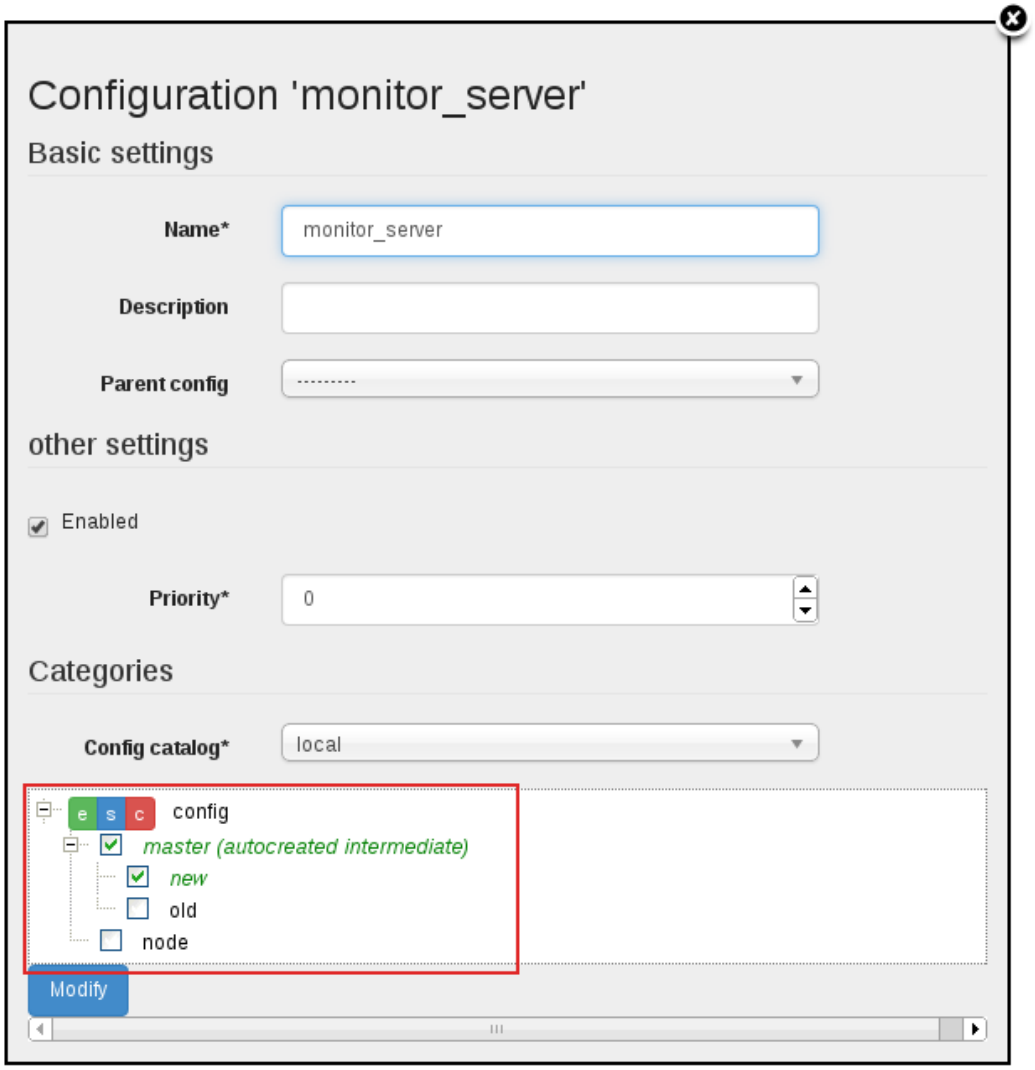


Now we would explain, how to use this categories in accord with another components of software, principally with devices and configurations.

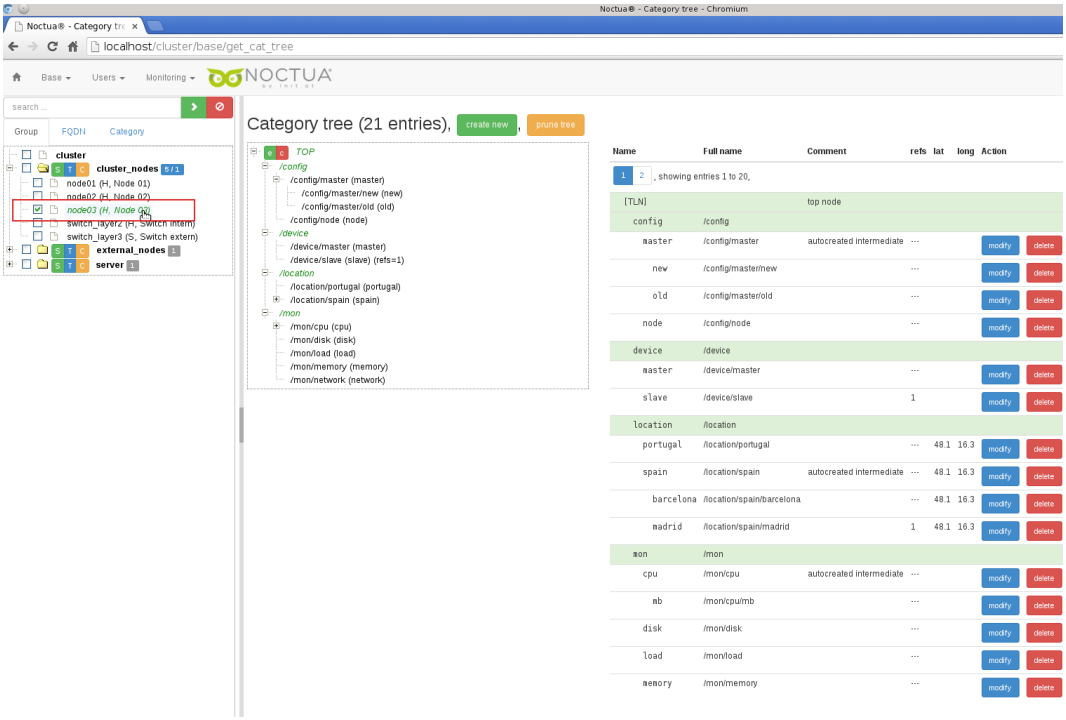
The "config" - category can be only used for configurations (Base -> Configurations). To attach one configuration to any "config" category, select under main configuration view (arrive by Base -> Configurations) the "modify"-button beside desired configuration.



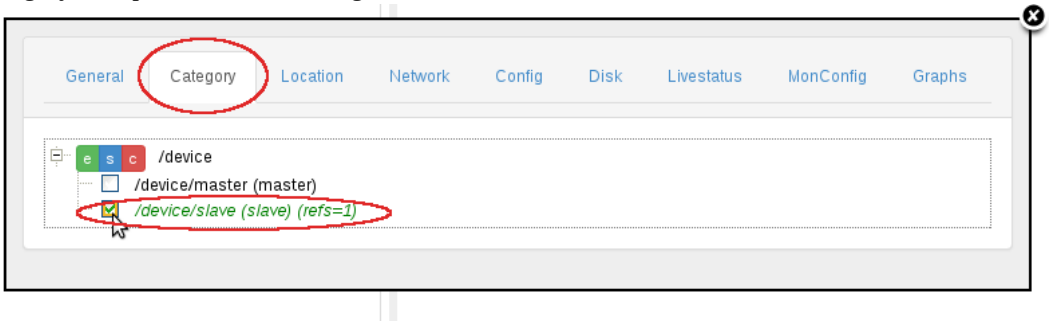
On the bottom of appeared window elect the category for this configuration and confirm your changes with click on "Modify" - button.



The sub-categories of "location" and "device" groups can be applied to single devices. To apply any device to "device"/"location"-subcategory, select the necessary device from device tree on the left frame. With double-click you get the properties-window of selected device.



Switch to "Category" - tab on property window. Select green "e" - button to expand the presented category tree, pick the desired categories.



To regulate the membership to "location"-category, switch to "Location" tab on property window of device. Operate the checkboxes for desired "location" - subcategories.

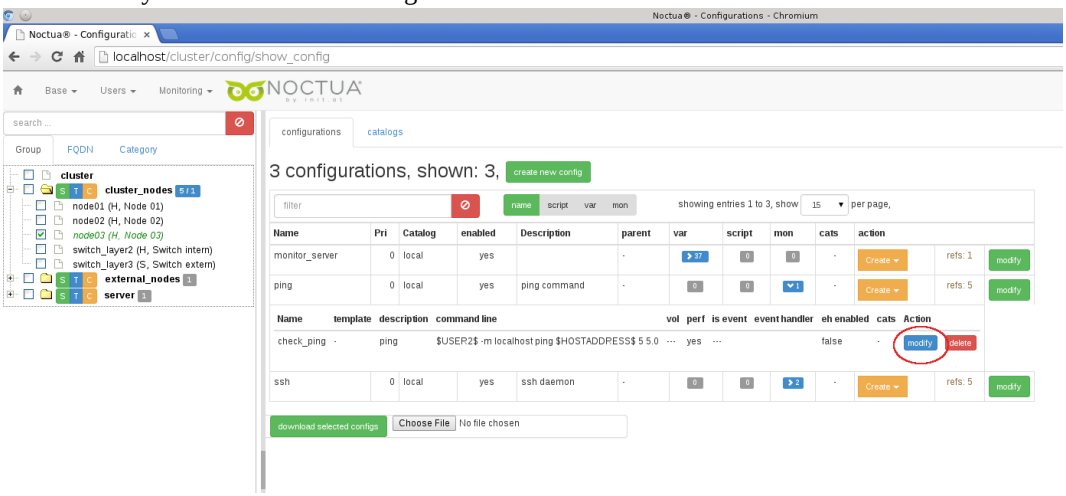


The last group or categories is the "mon" category. It covers the sub-categories for configuration checks and can be used only for these.

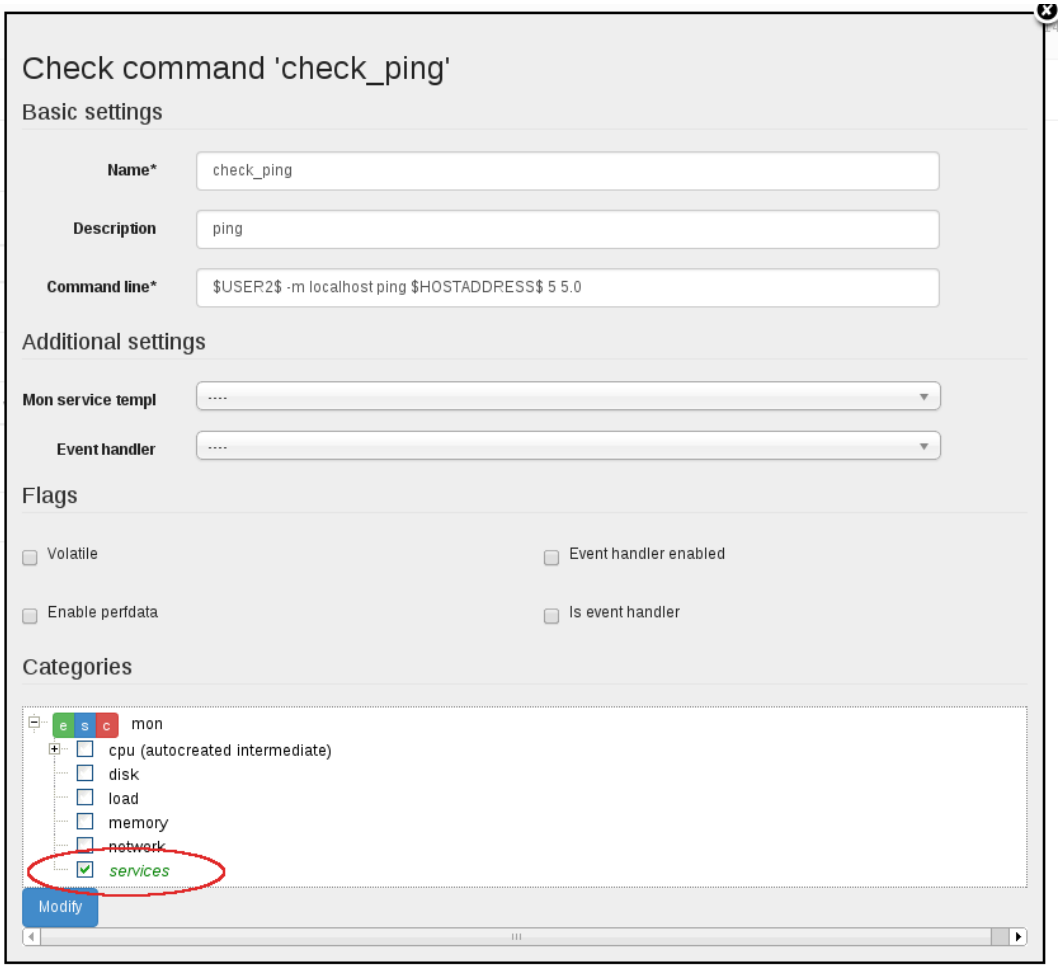
To assign a configuration check to mon-category, switch to Base->Configurations view, then expand the view of single configuration (click the blue-arrow in "mon" column).



Select "modify" - button beside the target check.



On the bottom of checks properties window you can see the "mon" subtree. Choose the necessary sub-category and confirm the changes with "Modify" - button.



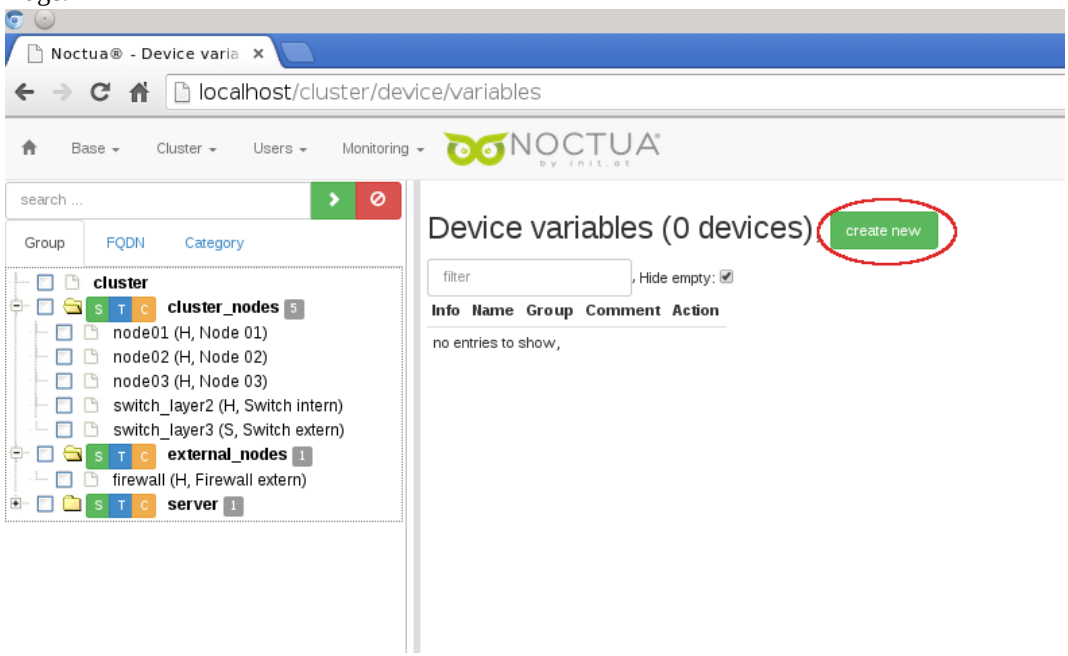
The membership to "mon" categories can be only applied to command checks. Scripting checks are excluded from this category by default.

Device variables

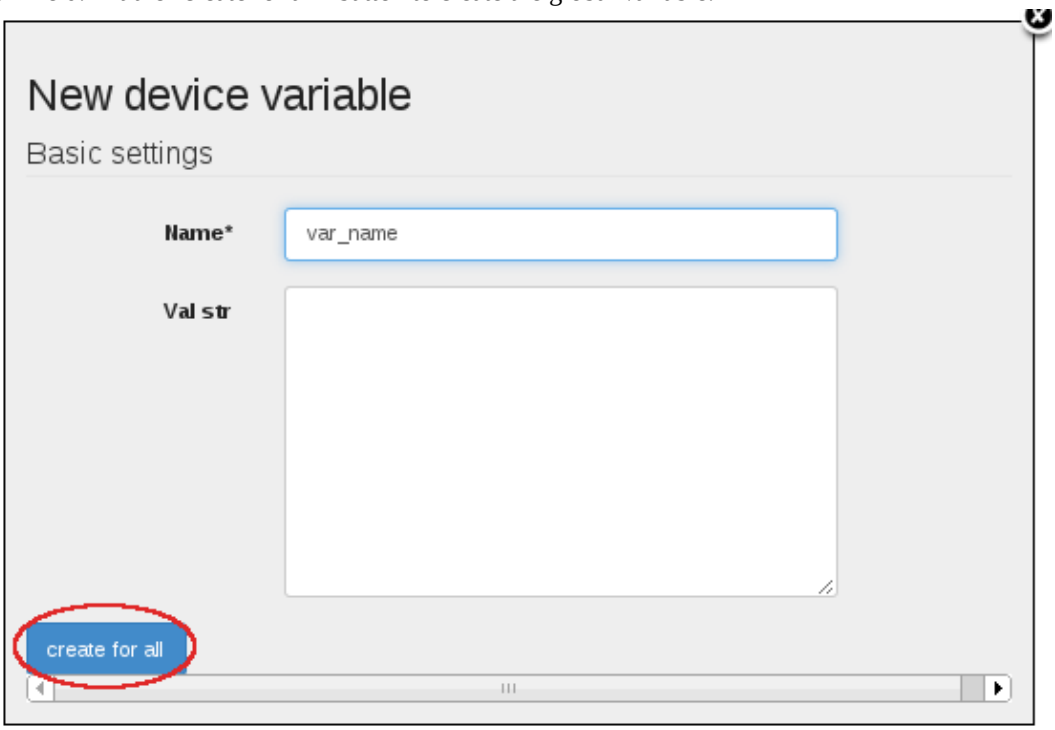
"Device variables" part of cluster/monitoring software gives the possibility to create global and local variables, which can be used as variables in configurations checks and scripts. To reach the interface of variables overview, select Base -> Device variables.



To define new global variable, click the "create new" - button on main overview, like shown on the image.



In the new window define the name of the new created variable. This name can be used in another areas of the cluster/monitoring software, especially in configurations. Define also the value in "Val str"-field. Hit the "create for all"-button to create the global variable.



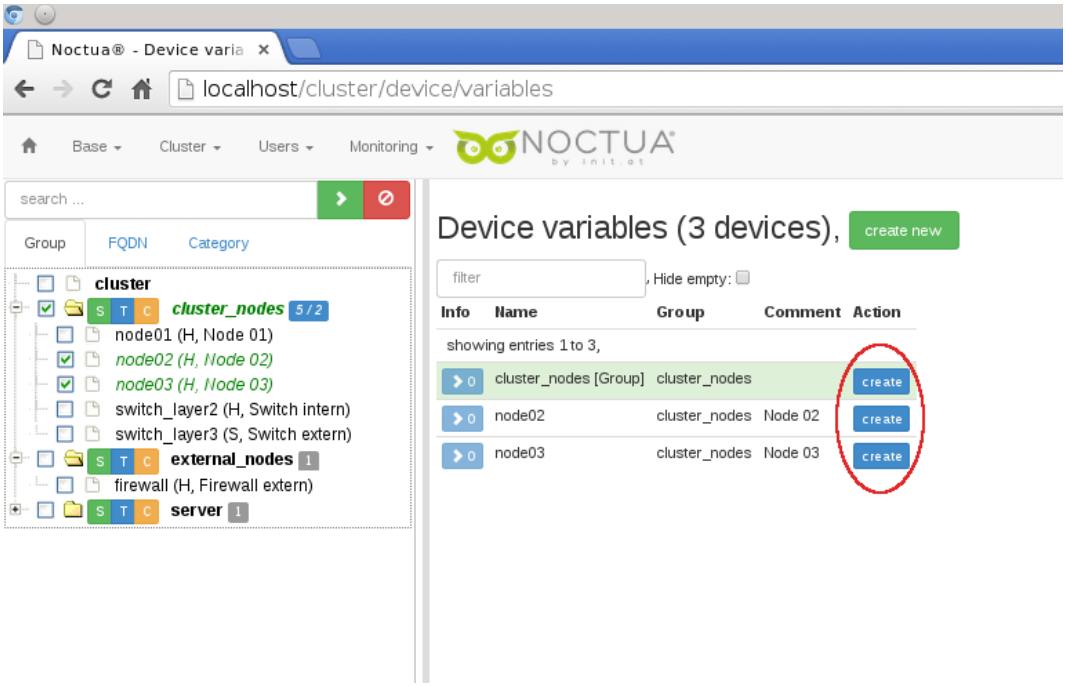
To create a local variable for the certain group or device, select this objects on the left device tree(1), then refresh the variables view with click of green arrow (2) like demonstrated on the picture.



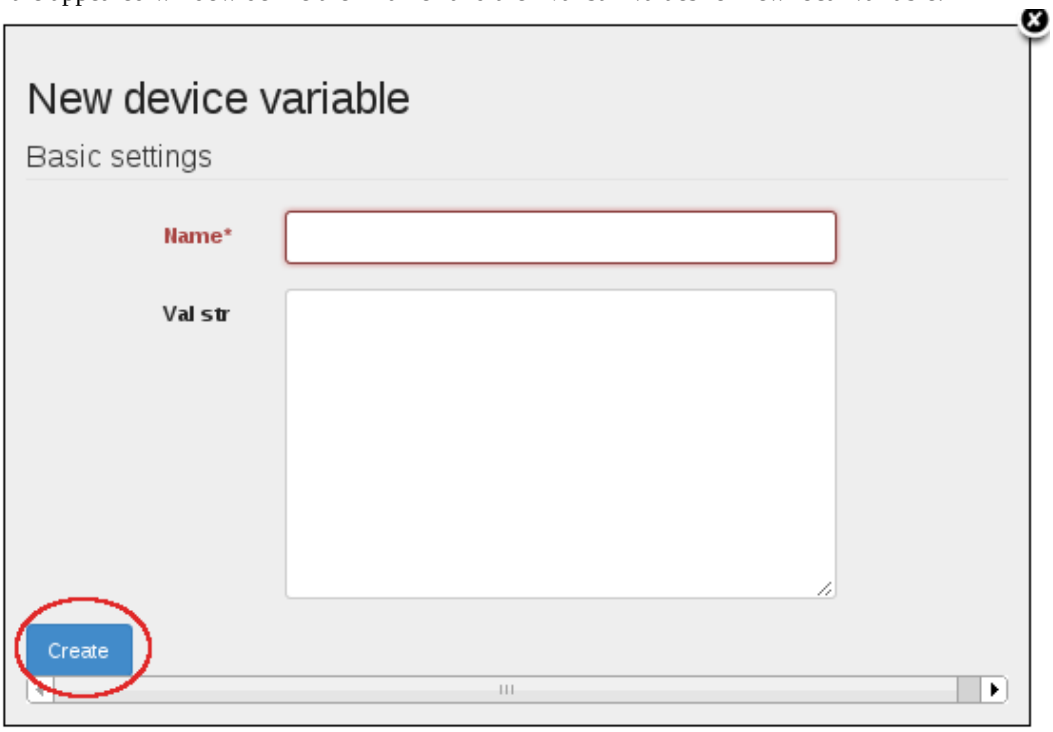
In our configuration there are no one variable presented. For that reason the message "no entries to show" was appeared. Click out the "Hide empty"-checkbox to illustrate the empty nodes.



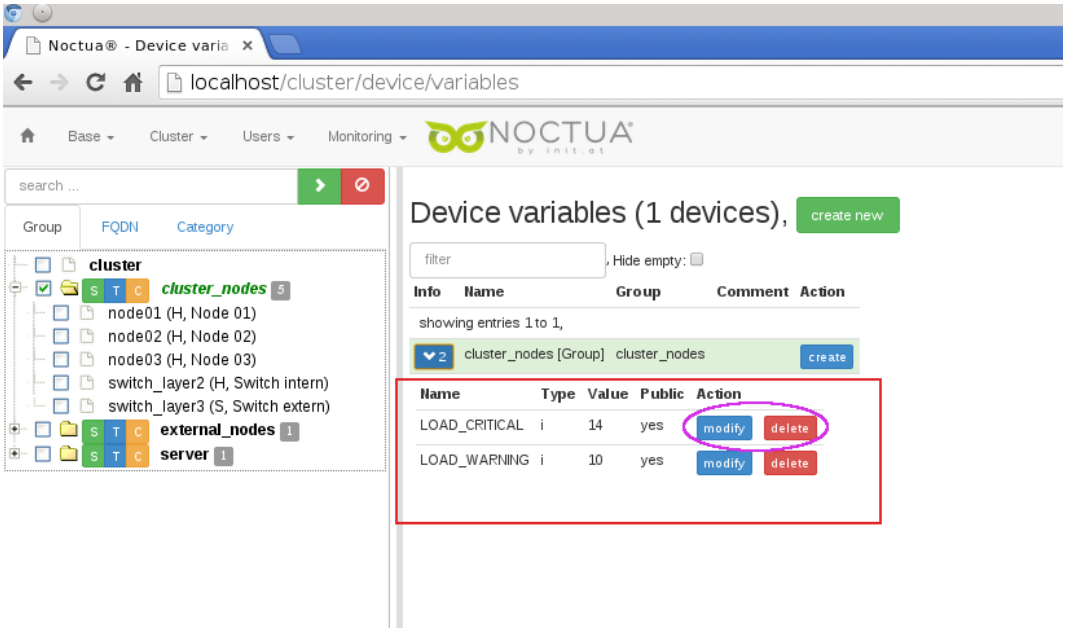
To define the local variable for certain objects, click the "create"-button beside the desired objects.



In the appeared window define the "Name" and the "Val str" values for new local variable.

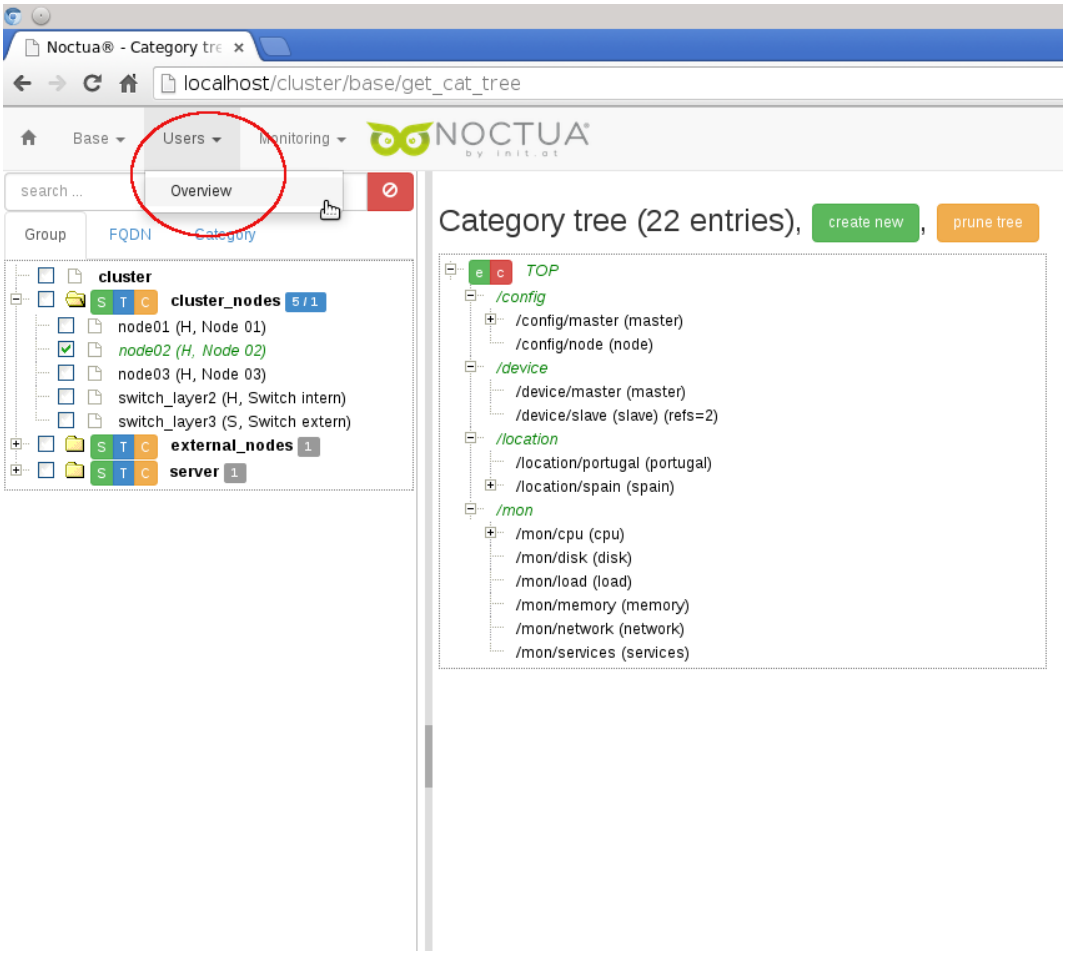


If there are some variables in the software were created, they can be listed on the variables overview. For each variable there is main information on the overview, also the "modify"-button - for modification, and "delete"-button - for deletion of the variable.

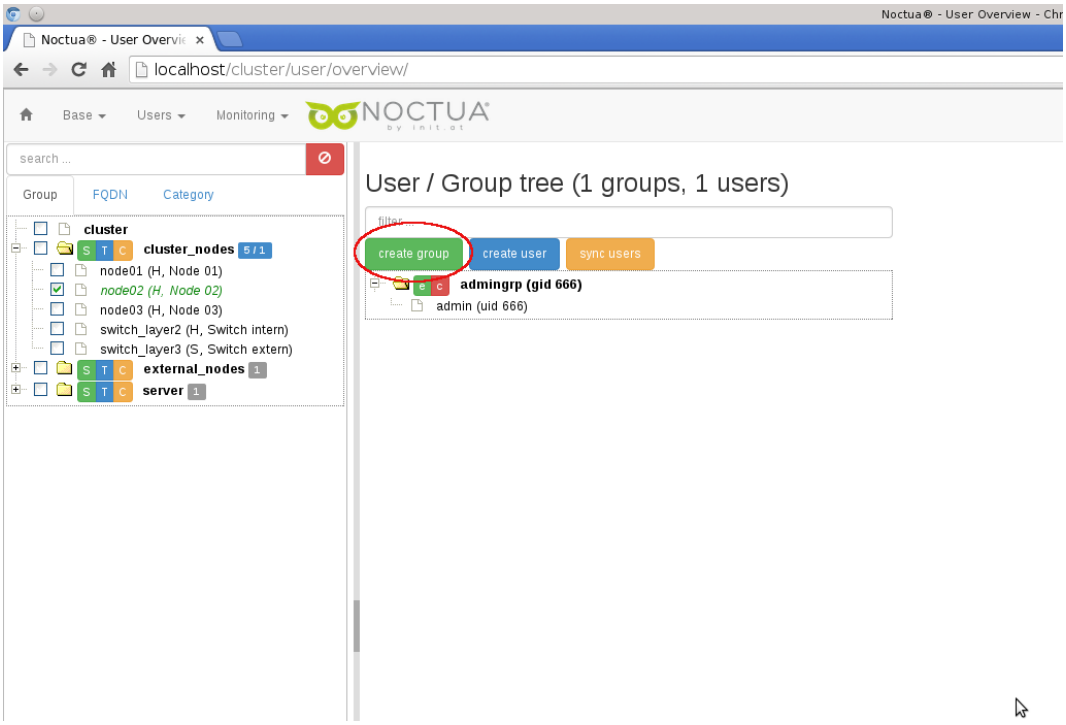


User definition

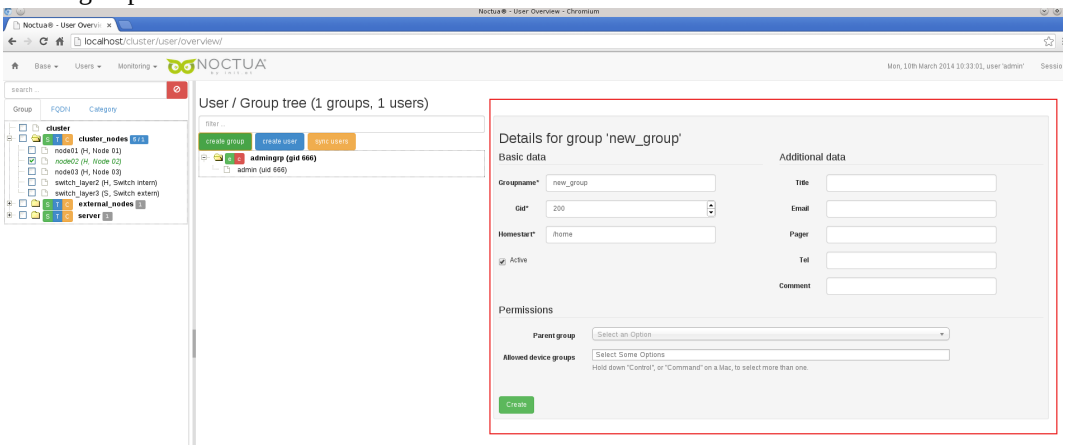
To make a change in user's structure of cluster/monitoring software, switch to menu option "Users"->"Overview".



In the tree of main view there is one during installation created group - "admingrp" and one user - "admin". To create new group, select "create group" button as demonstrated on the image.



After that new frame appears on the right site of the tree, where you can insert the attributes of new created group.



Here is detailed overview of usergroup attributes ('*'- required options):

- Basic data:
  - Groupname*: identifier for new created group
  - Gid*: is the group id, which will be used by creation group in ldap.
  - Homestart*: the home-directory of new created group
  - Email: email address of new created group
  - Mobile: give here the mobile-phone number of the group (in earlier version was named as "Pager")
  - Tel: telephone number of new created group
  - Comment: commentary to new created group

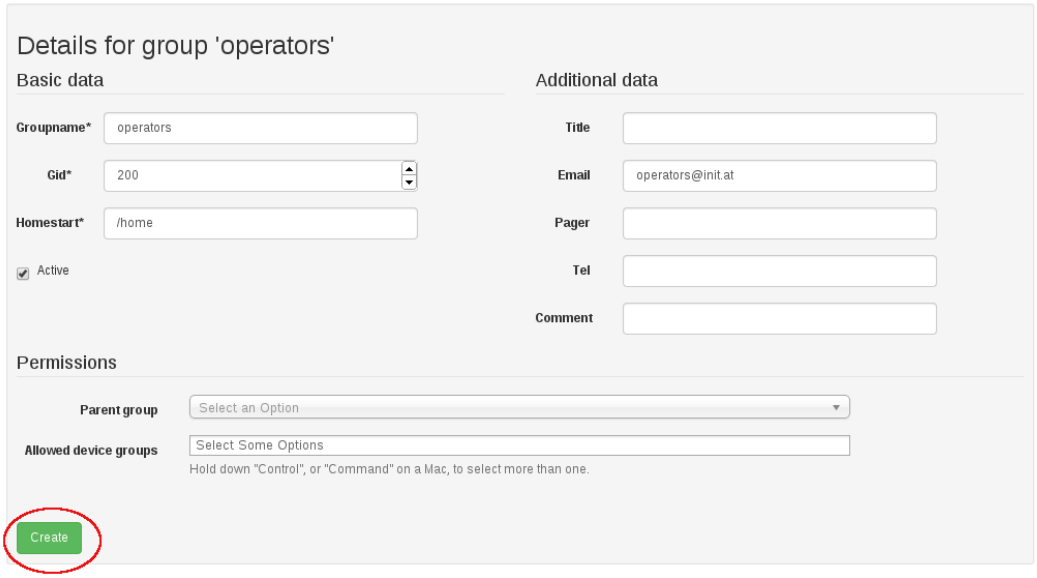
- Active: pick this checkbox, if you want to enable/disable group

- Permissions:

  - Parent group: this field presents the list of all existing groups in software. Select one of them to assign it as parent of new created group.

  - Allowed device groups: here you can choose one or many groups, which will be shown in device tree frame.

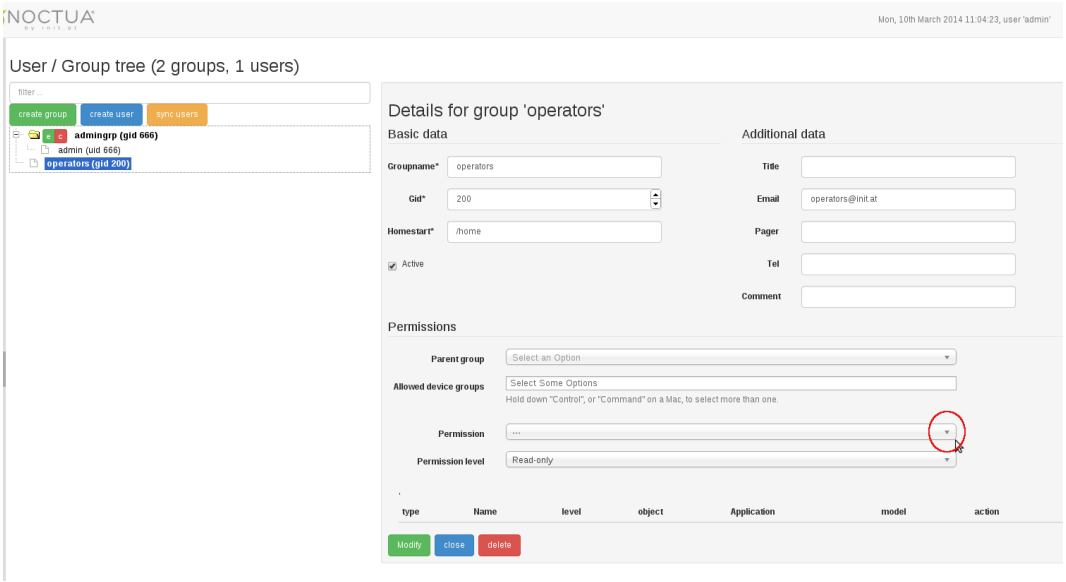
Confirm creation process with click on green "Create" - button.



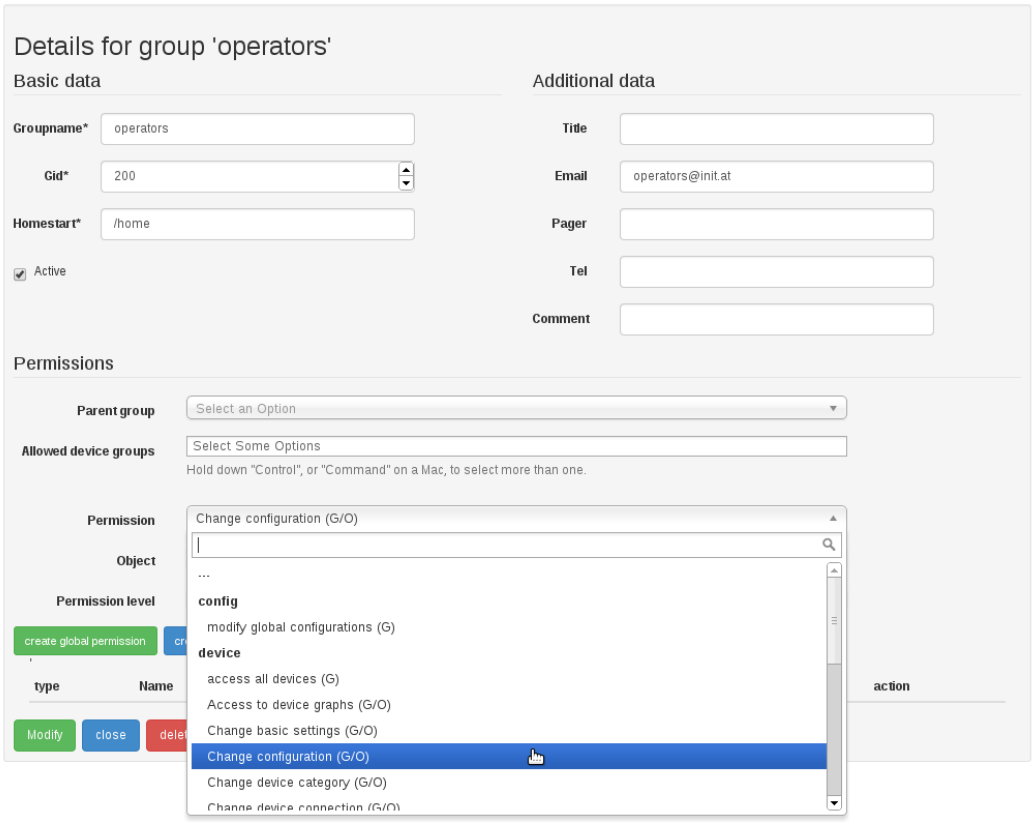
Once the group was created, you can decide, what permissions this group will get. With double click on the group identifier on user/group tree you get the group options in the frame on the right. In "Permissions" part there are two new fields:

- Permission: this field presents the lists of common permissions in cluster /monitoring software.

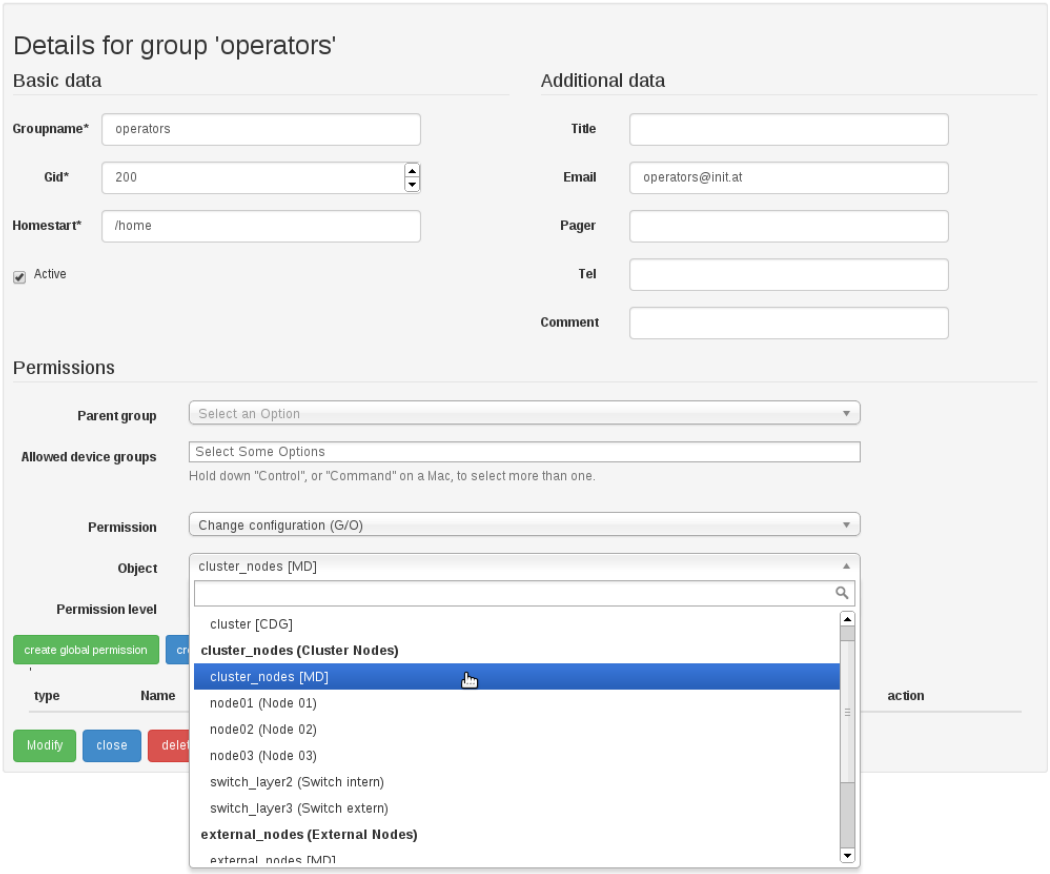
- Permission level: includes "Read-only", "Modify", "Modify, Create" and "Modify, Create, Delete" options in drop-down menu.



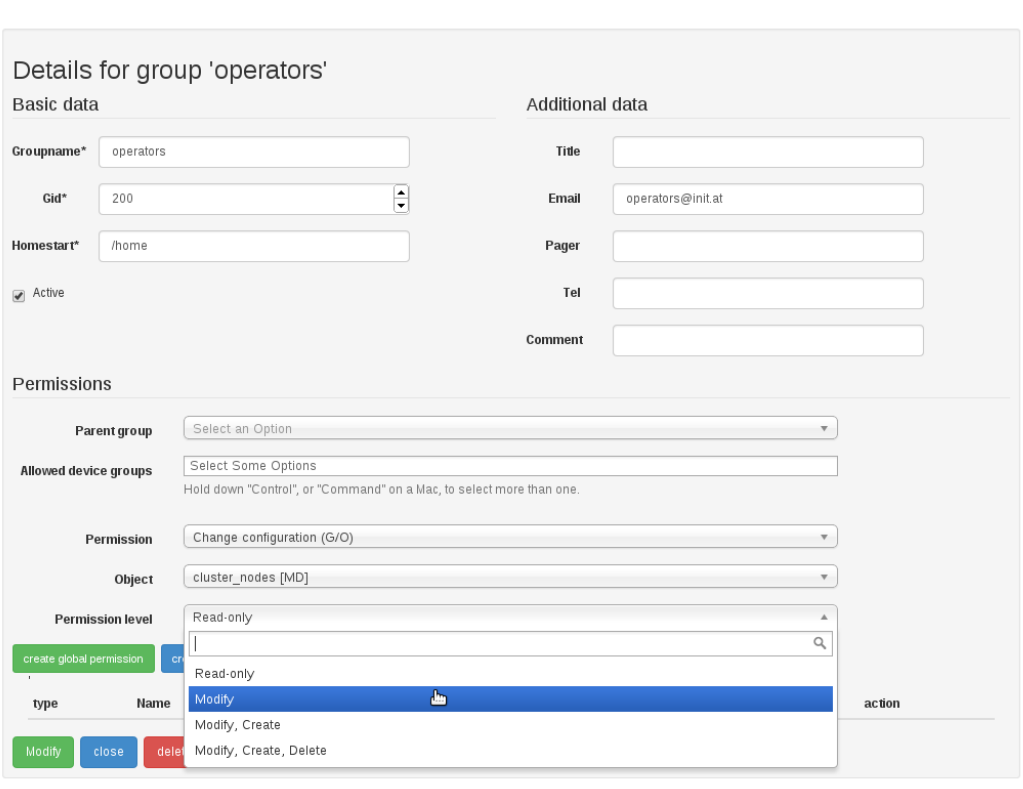
To allocate the permission to the group at first select necessary permission from drop-down list like shown on the image.



If the permission can be exercised on any device or group, the additional field "Object" appears below the "Permission" field. In the drop-down list there are single devices and groups (identified with "[MD]" symbol), which can be selected as manageable object.



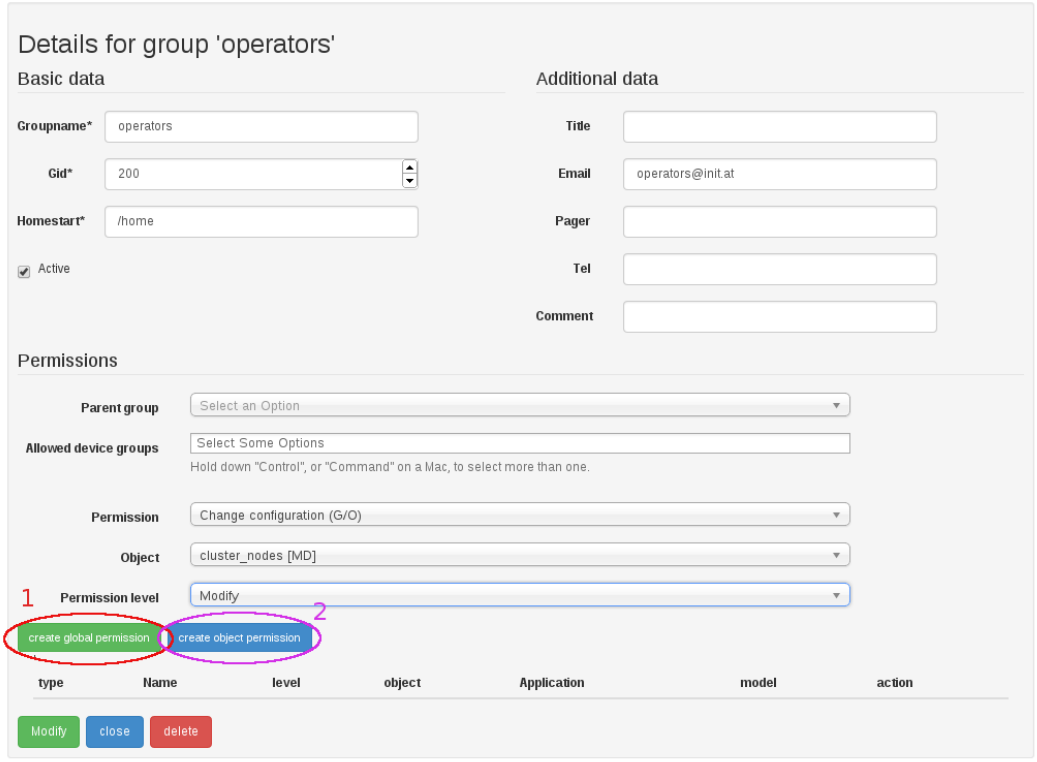
In the "Permission level" field select the level of the permission.



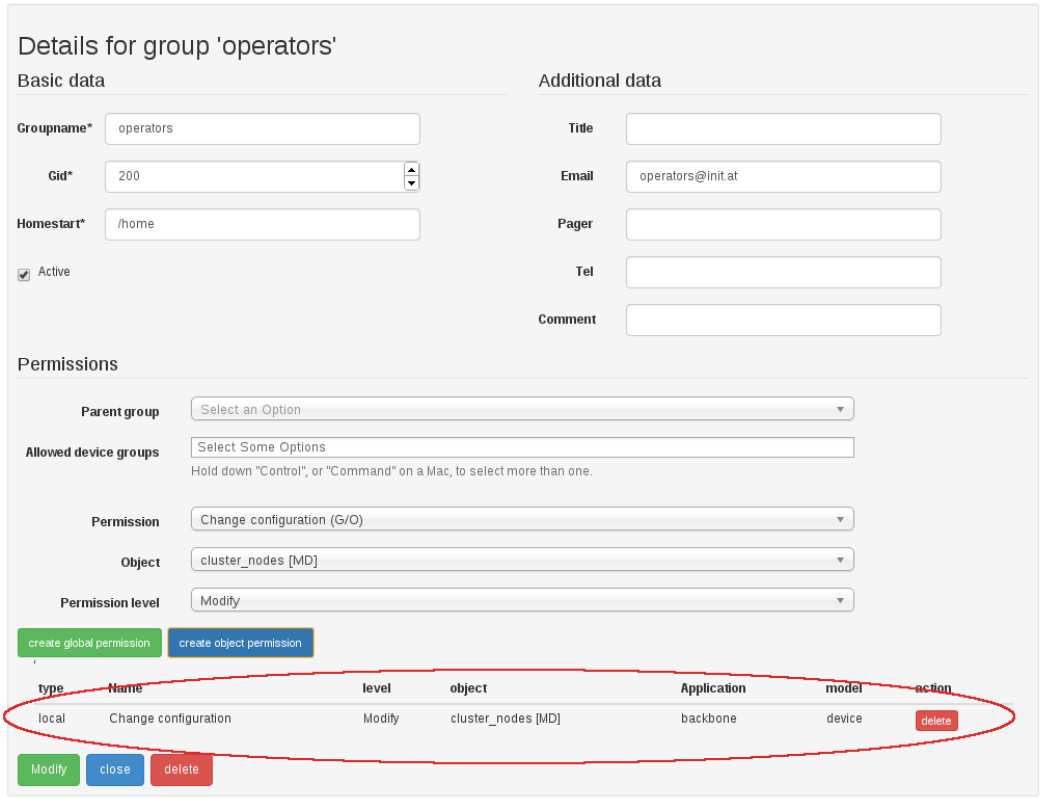
If three options "Permission", "Object" and "Permission level" are defined, two new buttons appear on the bottom of "Details" frame: "create global permission"(1) and "create object permission"(2).

Select first one - "create global permission" - button (1), if you want to assign this permission as global permission applying to all devices from allowed groups. In this case the value of "Object" - field is unimportant. This feature was developed for possibility of fast permissions definition.

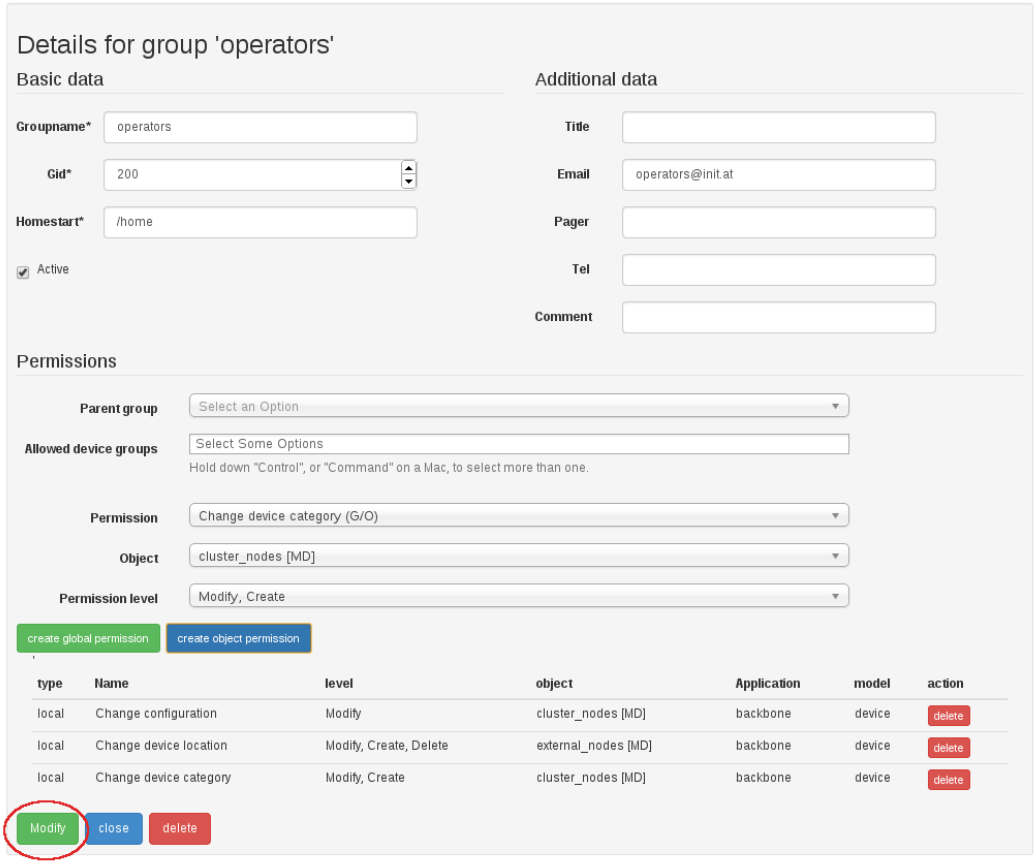
Choose second button "create object permission" (2), if you have to allocate this permission only for object, which select in "Object"-field. It can be a group or single device.



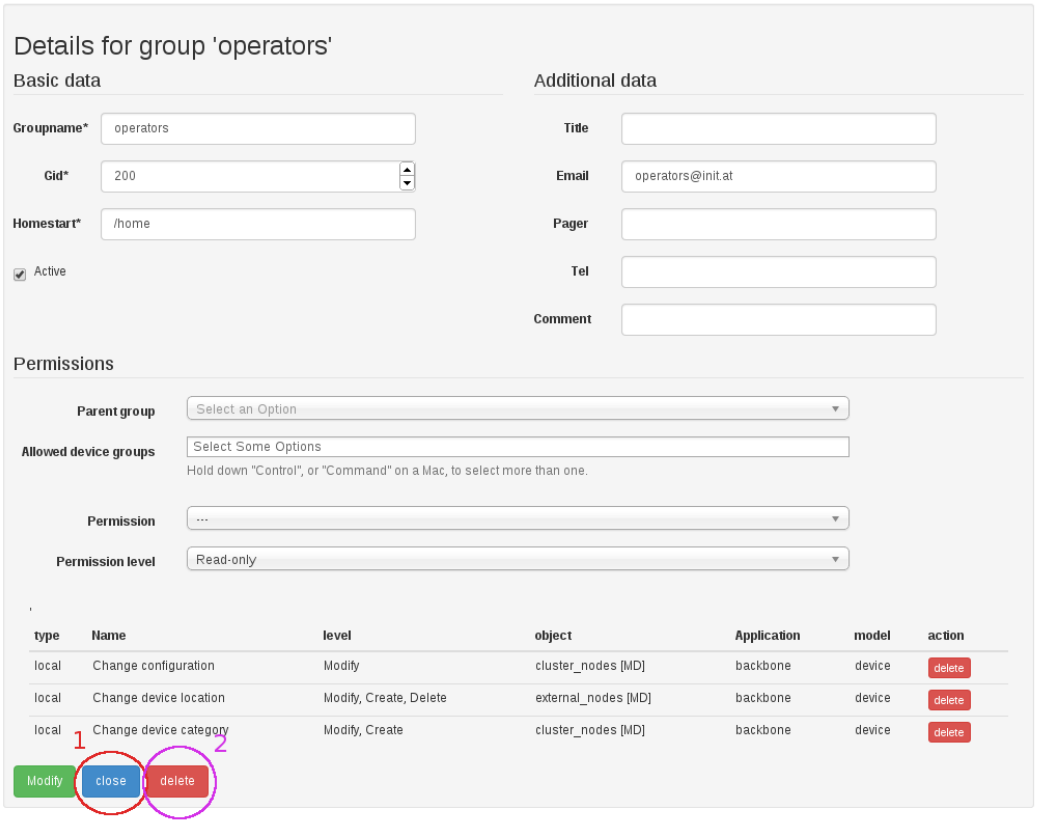
After activation of "create..." button the permission will be added to permission list on the bottom of the frame. Each row from this list includes main information of the allocated permission. On the end of each permissions row there is "delete" - button. Activate it, if you want to delete current permission.



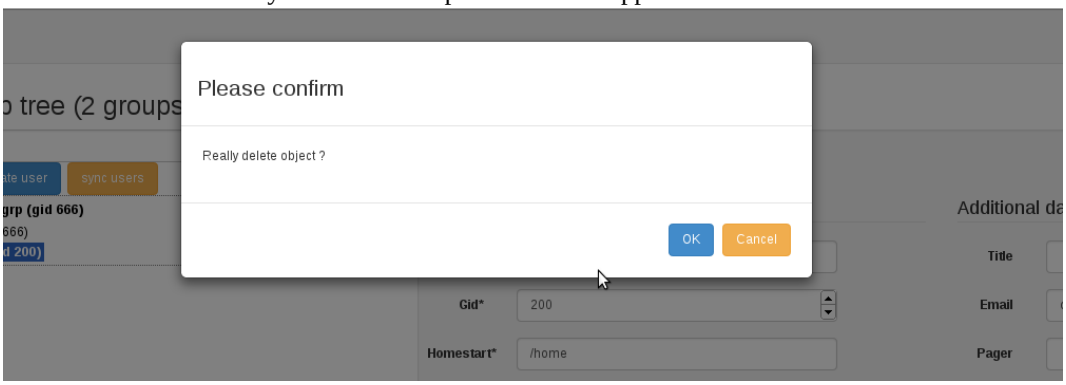
To apply the permissions to the usergroup hit "Modify" - button like demonstrated on the image.



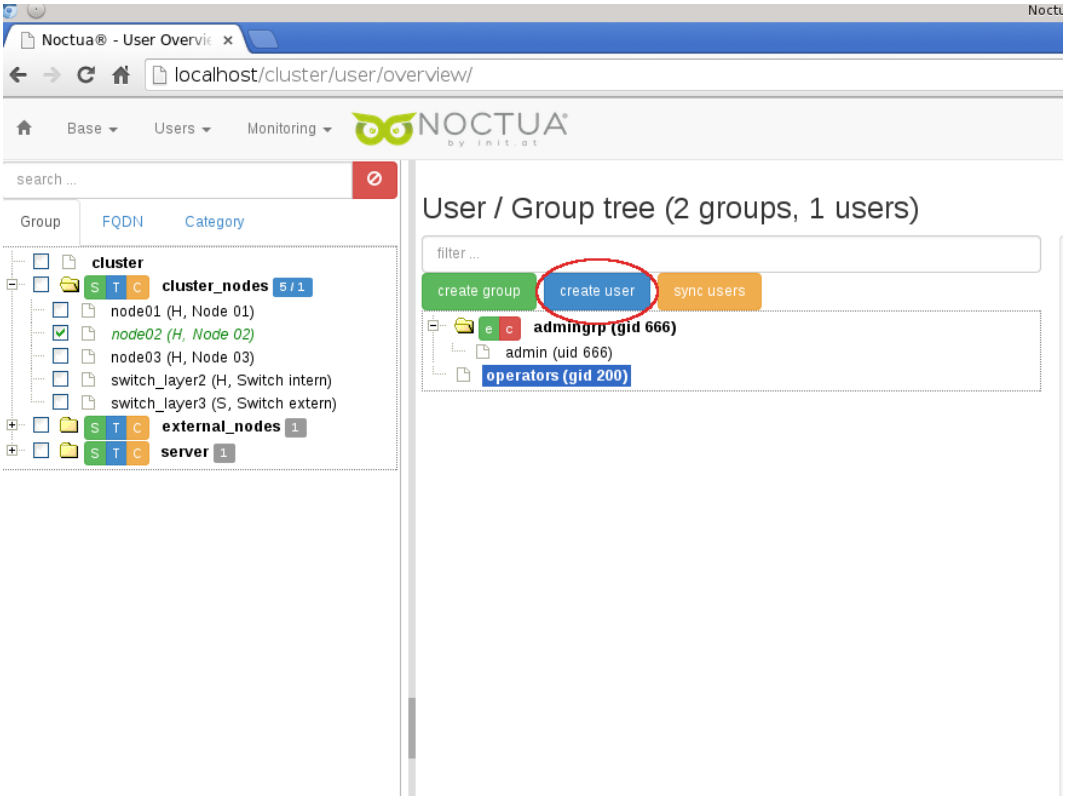
There are also two buttons: "close"(1) for closing of details view without object's modification and "delete"(2) - for deletion of group.



For deletion it is necessary to confirm the process in new appeared window.



For user creation actuate the "create user"- button like shown on the image.

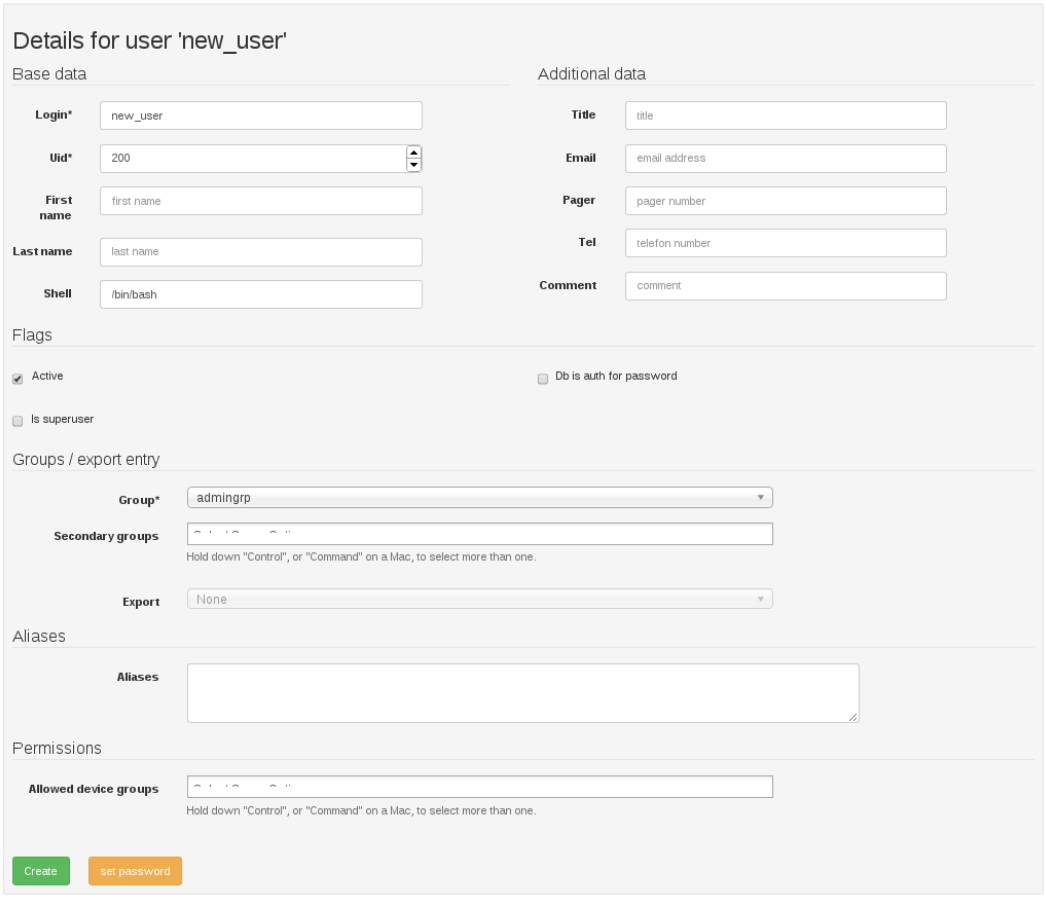


Here is detailed overview about user's attribute:

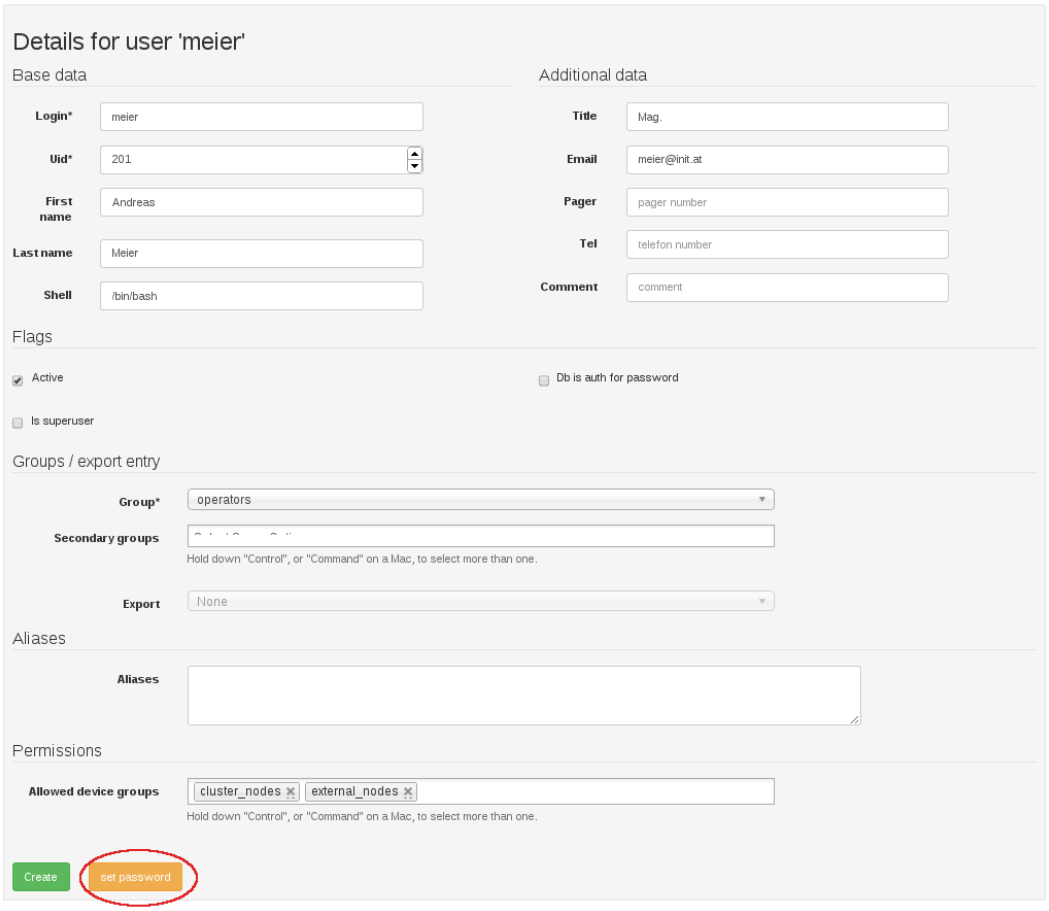
- Base data:
  - Login*: login name of new user
  - Uid*: user id
  - First name: first name of the user
  - Last name: last name of the user
  - Shell: here you can decide about active shell for the user
- Additional data:
  - Title: academic title of the user
  - Email: email address of the user
  - Mobile: optional field for mobile-phone number (in earlier version of software was named as "Pager")
  - Tel: phone number of the user
  - Comment: commentary for the new created user
- Flags:
  - Active: hit this checkbox if the user should be activated immediately.
  - Is superuser: activate this checkbox if you want to give the user the administrator's permission.
  - Db is auth for password: default value for this field is true, that means the password for the user will be inherited from "Noctua" database to the ldap-server. The fact of deactivation of this

checkbox makes safe the ldap password for this user from overwriting with value from "Noctua" database.

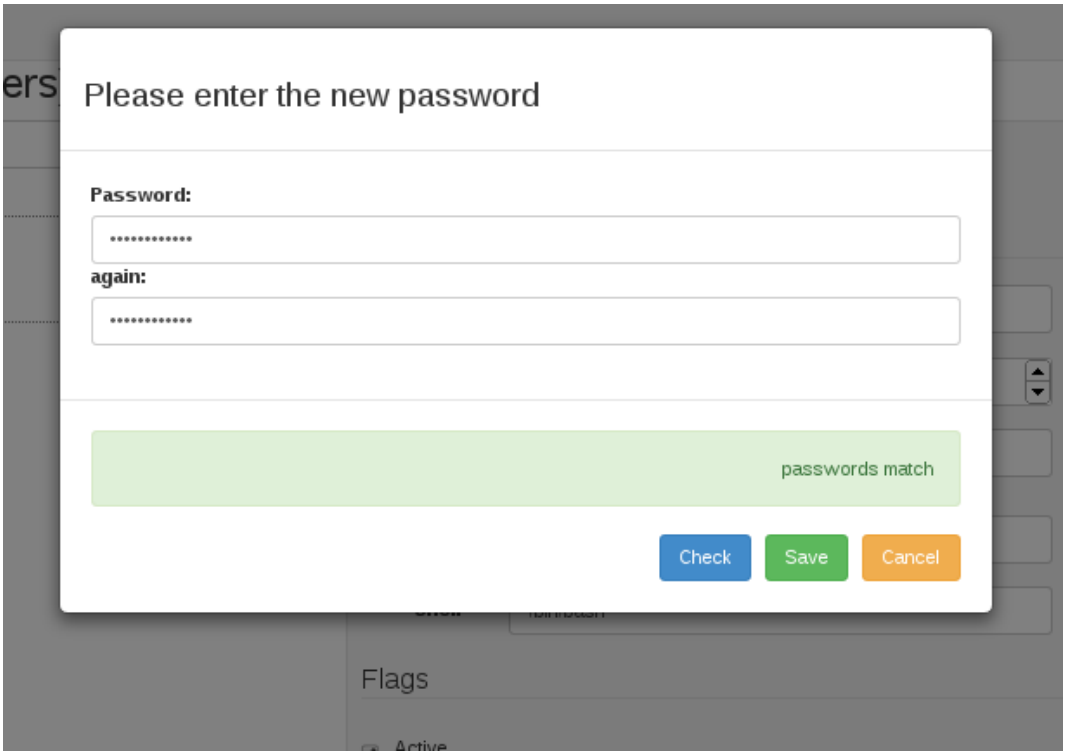
- Group / export entry:
  - Group*: select the main group for new created user from drop-down group's list.
  - Secondary groups: gives the possibility to attach the user to many different groups.
  - Export: lists a target location for home-directory of user. But previously the locations have to be declared under home_<dir> configuration under Base -> Configurations menu option.
- Aliases: give here the aliases, which can be used from user as login name. Use spaces, tabs, newlines between singular values.
- Permissions:
  - Allowed device groups: define device groups, which should be shown on the device tree in user-interface for new created user.



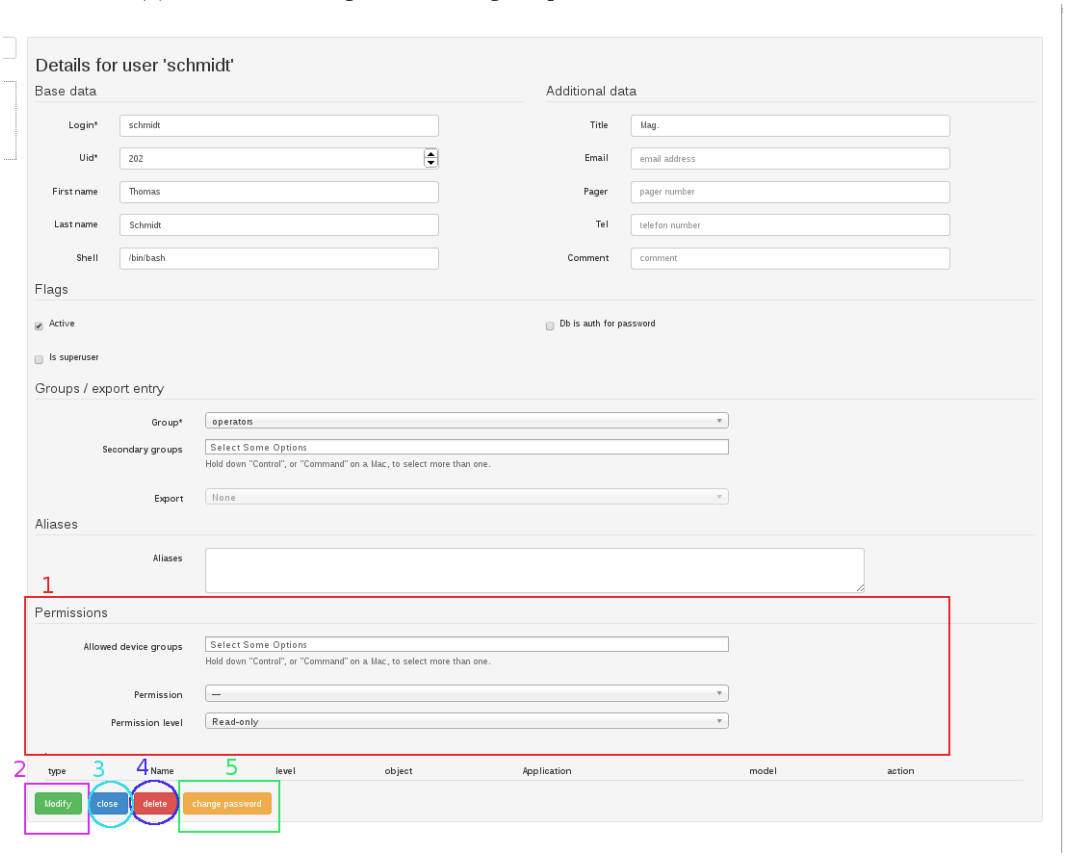
On the bottom of the user creation window there is the "set password" button. Hit it to define the password for the user.



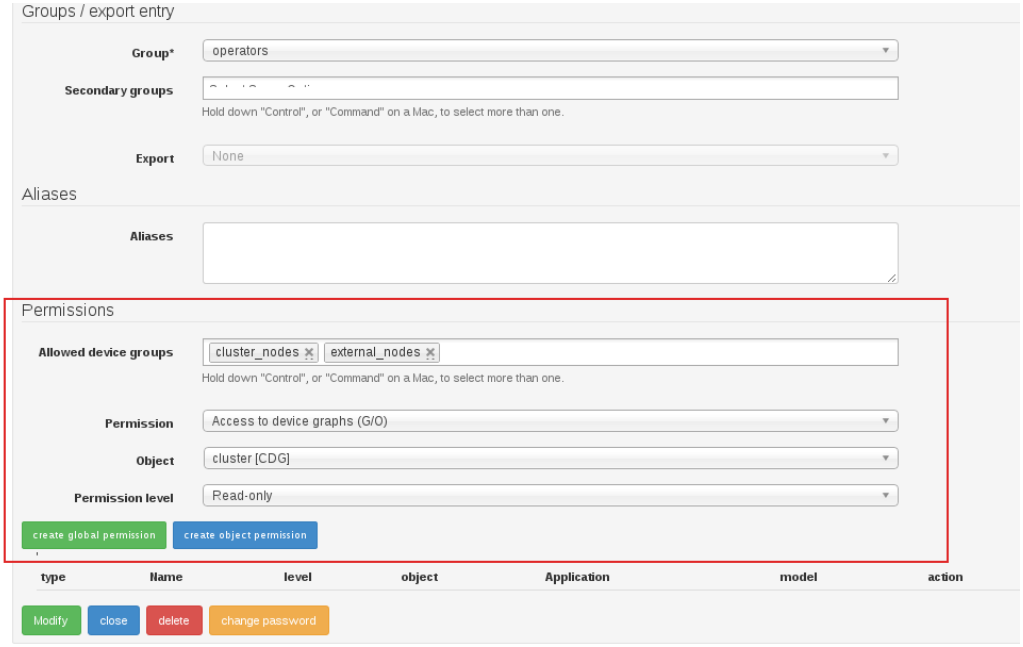
In new appeared window give the password two times. Control the messages from password policy, which will be shown on the line over the buttons line. If there are some mistakes in the password field, message line will be colored in red and error message will be shown. Hit "Check" - button - for password check, "Save" - to save the password and leave the password window, "Cancel" - to leave the window without saving.



Once the user was created, you can add the permissions (1) for this user. "Modify"-button(2) will apply the changes to user-object. "Close"-button(3) closes the window without saving any attributes from user. "Delete"-button(4) manages the deletion process of the manageable user. "Change password"-button(5) allows the dialog for re-setting the password.



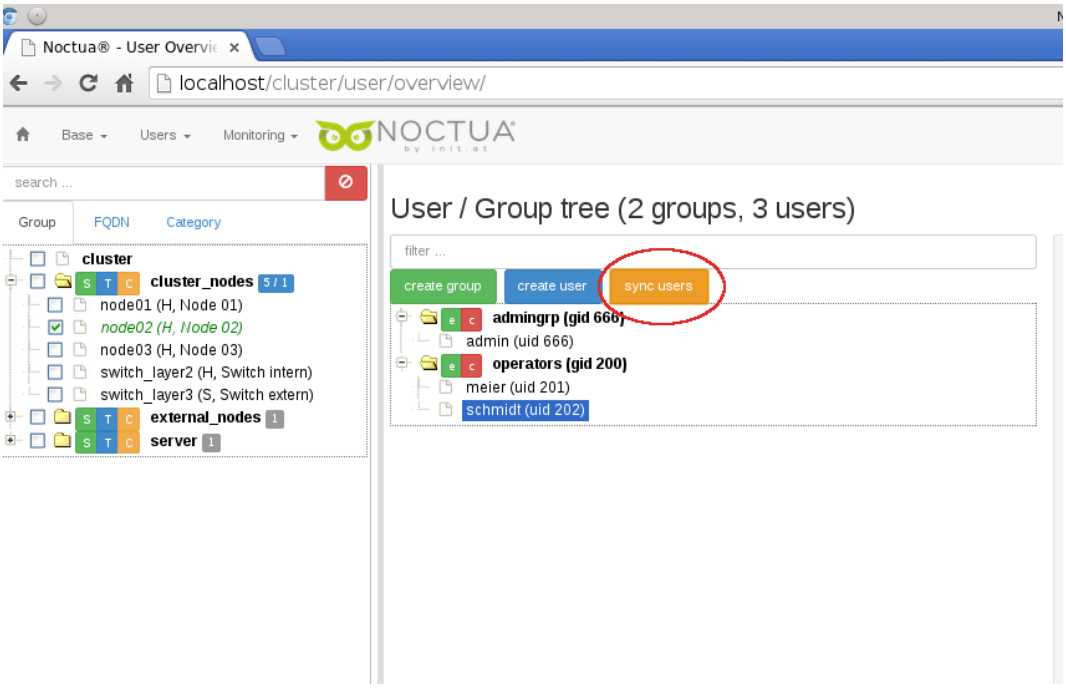
The fields of permissions area are very similar to the same fields on the group details window, which has been introduced earlier in this documentation. To understand the behavior of the permission rules, see the explanation in part "User group creation".



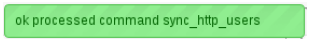
On the user/group tree there are two keys for switching between expanded and contracted view of the tree.



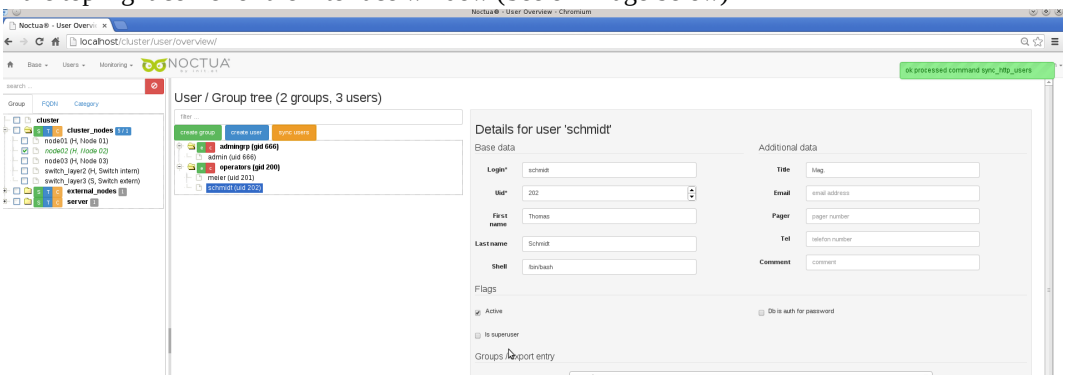
"Sync users"-button will synchronize the users/groups objects with the ldap-server, which is running in the background. During synchronization process the home-directories will be also created for the users, if they are not still exist.



If synchronization was running successfully, you see following message



in the top-right corner of the interface window (See on image below)



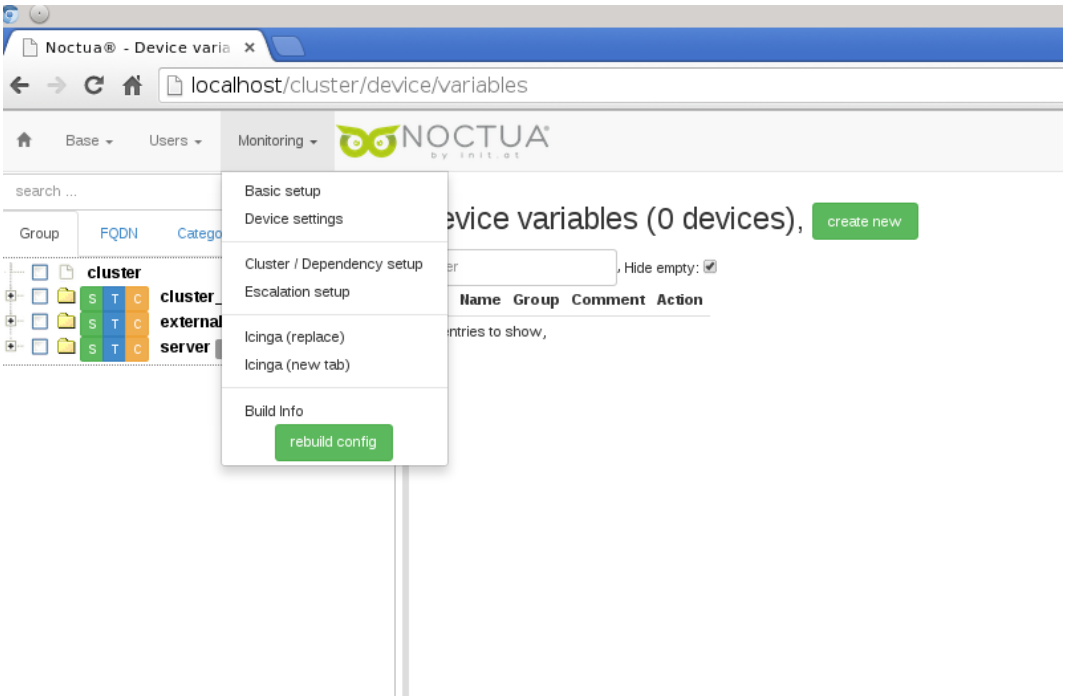
Monitoring setup

After basically configuration of Noctua software, you can switch to the "Icinga"- webfrontend to control and monitor the processes from "Noctua" - landscape.

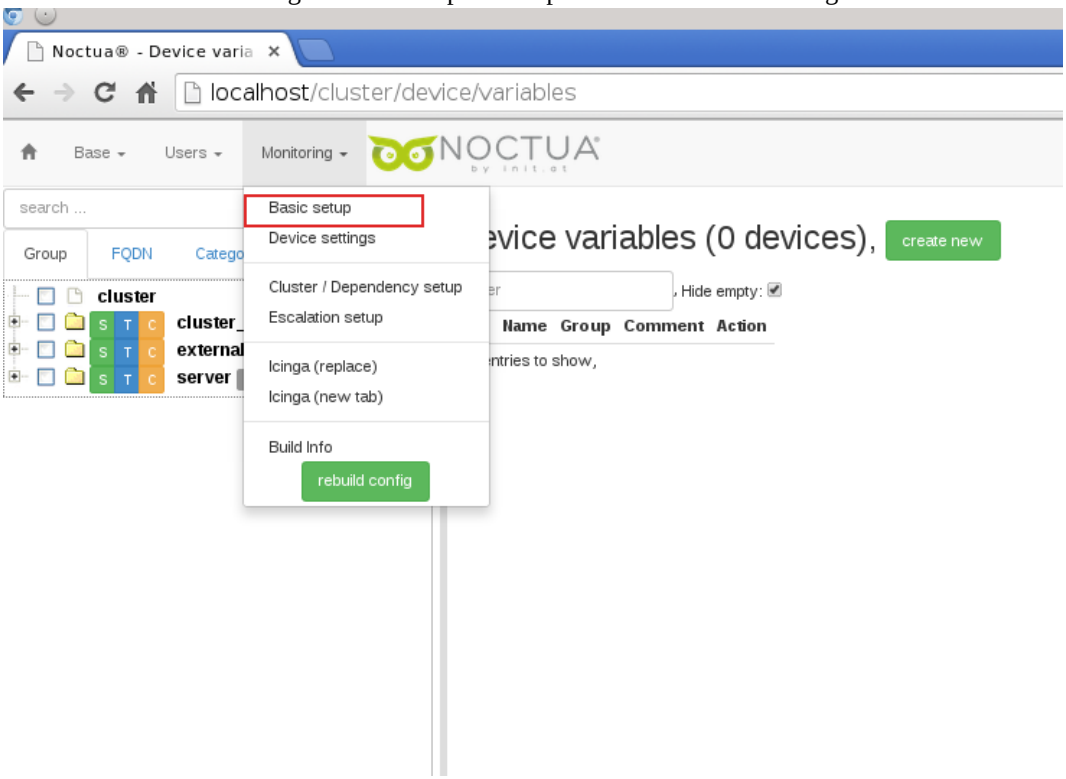
There are the options of monitoring menu options:

- Basic setup:
- Device settings:
- Cluster / Dependency setup:
- Escalation setup:
- Icinga(replace): switch to Icinga-webfrontend in the same web-browser window
- Icinga(new tab): switch to Icinga-webfrontend in the new tab of the web-browser

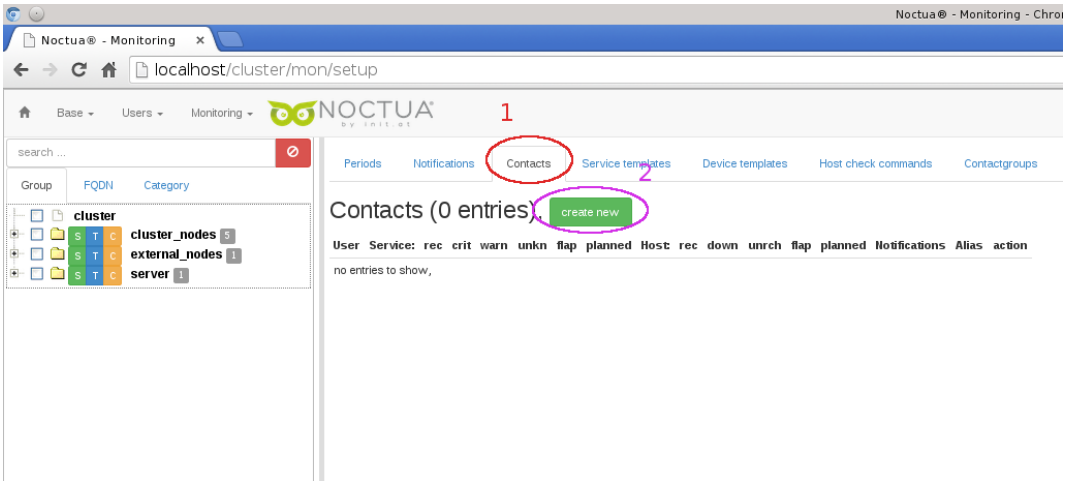
- Build info:
- rebuild config: selection of this button erases the rewriting of file based configuration of Icinga.



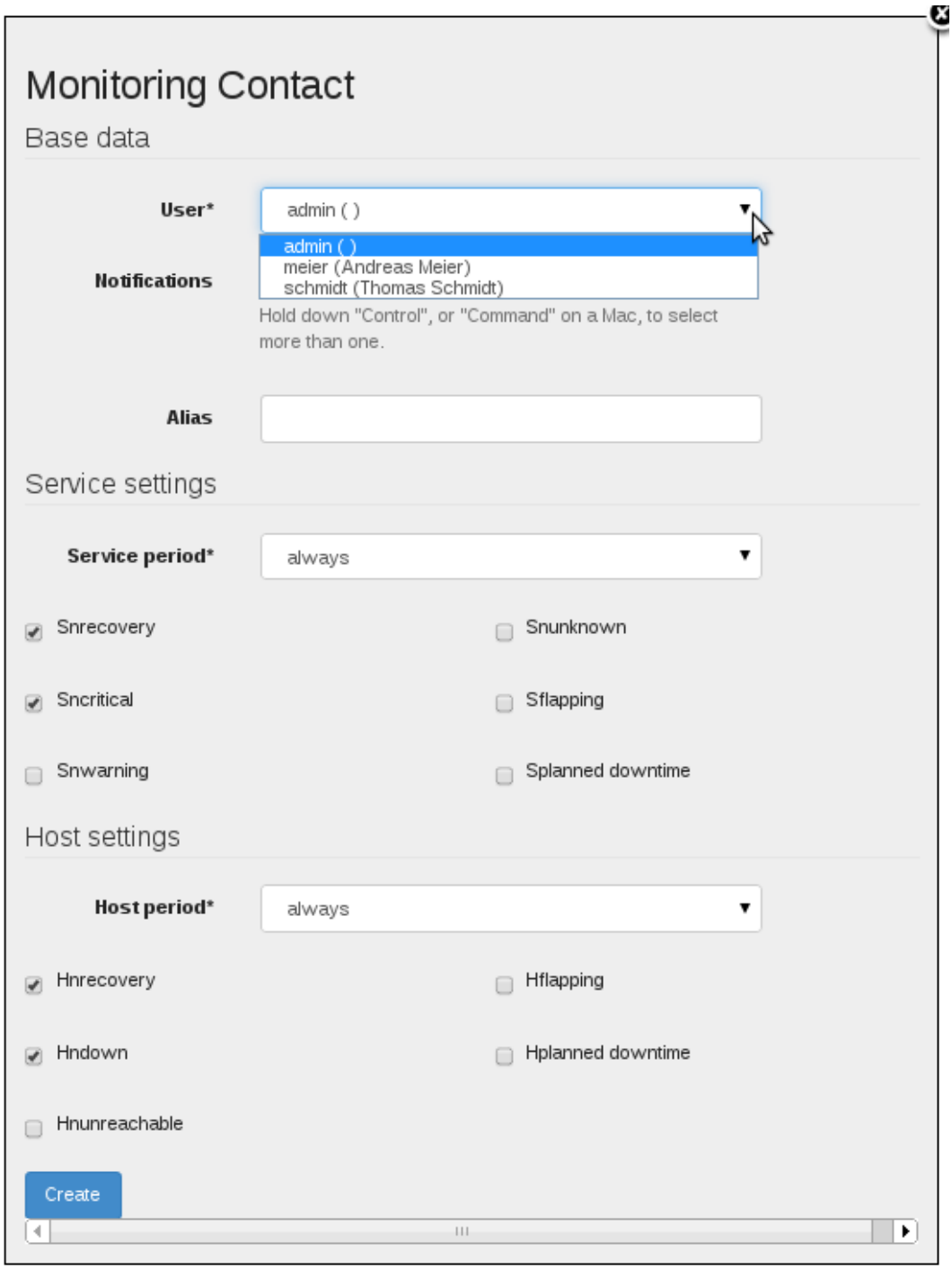
Before you can use the Icinga webfrontend you have to expand the basic setup of Icinga. For this feature switch to Monitoring -> Basic setup menu option as shown on the image.



For first use of Icinga webfrontend it is necessary to create the first contact under Base settings - > "Contact"-tab(1). To create new contact select the "create new" - button(2).



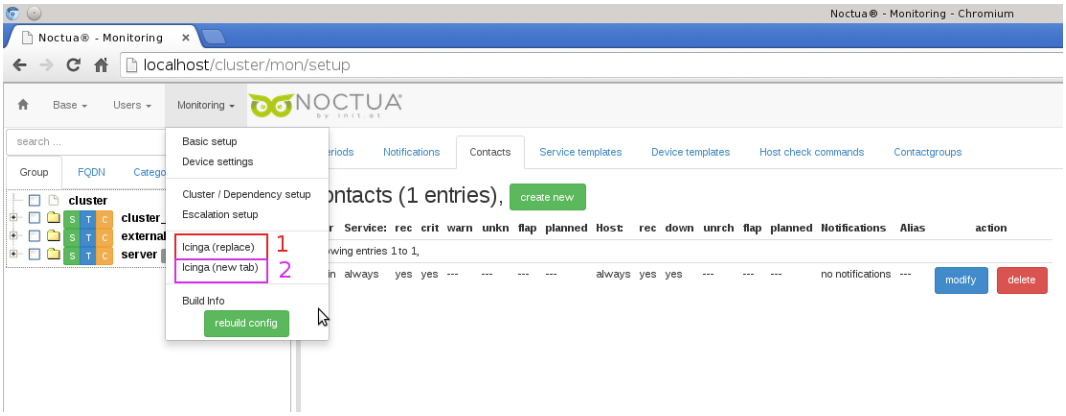
In "Monitoring Contact" window select the User, which has been already created under User->Overview menu option. Selection of the "Create"-button will create new Icinga-user.



Select "rebuild config" - button to recreate the "Icinga" configuration file in the background.



Now you can switch to Icinga-webfrontend. Select the "Icinga(replace)" - option(1) to replace the "Noctua" view with Icinga-interface. Choose the "Icinga(new tab)" - option(2) to activate the Icinga in new tab.



Adding nodes

Add a new node by visiting Setup → Trees → Device Tree. Add an new device Group for your Nodes. Add the first node. If your node has some kind of IPMI Interface add it a controlling device with the right ip address. The *IPMI* password and username can be set by creating configuration variables named `IPMI_PASSWORD` and `IPMI_USERNAME` for the META-Device for all ipmi devices. Associated the controlling devices with the node by creating a connection under Setup → Device settings → Connections . Create a Group for your Server Add the device clusterserver

Create the first network Under Setup → Networks You have to create a boot and a installation network. This separation is required by mother and generally considered a Good Thing™.

Connect your clusterserver and the first node. **CORVUS®** manages and internal peering table. This structure should in a way represent the logical connection of devices. In most setup this will mimick the physical network connection - but sometimes a more complex setup is required. (Some nodes should use another LDAP server than others...) Nodes and clusterserver are connected by adding the clusterserver as a peer of the node. The clusterserver itself has to be marked as "routing capable". Connections can be visualized under Setup → Device settings → Network

To install the nodes you have to create three things. A partition, a kernel and a node image. We are going to start with the Partition. Create a new partition under Operation > Partition overview. Make sure to set the nodeboot flag of the partition.

Creating a kernel Create a configuration named kernel_server. Assign it to the clusterserver. Create a configuration named mother_server. Assign it to the clusterserver. Choose a directory that contains your kernels. We will be using /usr/local/share/cluster/tftpboot/kernels/3.7.10 in this tutorial. The default target directory is `/tftpboot/kernels`. Copy the compiled kernel to the newly created folder. Rename it to bzImage. Copy the System.map to the folder. Rename it to System.map. Copy

the config file of the kernel to the newly created folder . Rename it to .config. If you are running the same kernel you can do zcat /proc/config.gz > .config Copy /lib/modules of the desired kernel to the newly created folder + "/lib". Copy /lib/firmware to the new folder "/lib" + firmware + "kernel-name". Create a bz2 tar of the ./lib/modules tar -cj lib > modules.tar.bz2 Copy the firmware to lib/firmware/(kernel-version)

Tip

If you don't want to compile a kernel by hand you can use **copy_local_kernel.sh**. This script prepares a given kernel for use with the cluster software.

`copy_local_kernel.sh` [*KERNEL_NAME*] [*KERNEL_DIR*]

To create the initial ramdisk the script **populate_ramdisk.py** is used.

`populate_ramdisk.py` [-m *LIST_OF_MODULES*] [*KERNEL_DIRECTORY*]

Use `populate_ramdisk.py -L -L` and `populate_ramdisk.py -L` to show all the generated ramdisks and the included modules.

Example 3.1. Using populate_ramdisk.py to create a ramdisk

```
clusterserver:~ # populate_ramdisk.py /usr/local/share/cluster/
kernels/3.7.10 -m list_of_modules_you_need

...
```

The list_of_modules_you_need will most likely be the modules used for your node networkcard. Watch for file missing errors and install the missing binaries to your system. killall for example would be found in the package psmisc. Restart mother. The mother scans its kernel directory and adds the new kernel into the database. The populate_ramdisk.py script copies the stages scripts /opt/cluster/lcs and some binaries into an initrd + kernel usable for pxe booting. Now you can associated your first node with the new kernel via operation > nodeboot.

Creating a new image Create the configuration image_server and associate it with the clusterserver. Create "/usr/local/share/cluster/images/openSuSE-12.3" Create a new architecture object in the DB Create a new image object in the DB. Set source to the newly created directory. Copy a full / structure from an existing installation or install into this directory. This is the pattern for the image that all nodes are going to be provisioned with. Make sure you include the /run directory in your image Remove any network configuration form the new image so no conflicts occur when the node is first brought up. If you are doing an installation into a directory don't configure the network interfaces. Install libxml2-tools (xmllint) to create the new image. call **make_image.py -v --build-image** watch for packages missing errors. install the missing packages into the template. Either chroot into the dir and install or your package management software support chroot installation. Check /tftpboot/images: if xz files of your toplevel directories exist hte image was built correctly. Hit scan for image -> Then use -> Check if the right vendor was detected!

Create the config master_server and syslog_server. Associated it with the clusterserver. First provisioning install dhcp-server (ISC DHCP Server, needed for it's OMAPI Interface) install tftpserver install xinetd install rsyslog install atd - make sure it is running Change tfpd configuration in xinetd to /tfptboot. Set the motherserver of node01 to "clusterserver" under setup > device tree > "". Start mother make sure mother writes /etc/rsyslog.d/mother.conf create /var/lib/dhcp/db/dhcpd.leases Create the dhcp config **cluster-server.py -c write_dhcpd_config -D authoritative:True**

DHCP OMAPI

CORVUS® talks OMAPI with the DHCP server.

Add the first IP to the node. Set the clusterserver routing capable under setup > device settings > network. Add clusterserver as peer to node01. Set DHCP under operation > node boot > network. Make sure that a NFS server is up and running on your clusterserver. The Clustersoftware just writes `/etc/exports` - the server has to be started by hand. A!!! Ask if config has to have the name base" Create a config named "base" with a variable named INIT_MODS = "nfsv3 nfsv4 nfsv2" There you should specify all the necessary modules to access a remote NFS server and your local node disk.

Example 3.2. foo

nfsv3 nfsv4 nfsv2 ata_piix

start the node and watch for errors on booting. You should have a first node up an running after this step. Check that all basic software packages that you need are included in the image. Special packages can be installed on the node later on. To add software to the image - add the software - rebuild the image and reboot the node. If you are satisfied set the target state of the node to "boot into production". under Operation > nodeboot. Create the config auto_etc_hosts and associated it with clusterserver. Execute **cluster-server.py -c write_etc_hosts** to get a /etc/hosts file with all the cluster names.it.

Node configuration Under Setup>Configurations add a script to the config base with the following content: `do_nets(config) do_routes(config) do_fstab(config)`

Chapter 4. Configuring Nodes

Overview

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*]

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.

Chapter 5. Installing and distributing packages

Installing and distributing packages

package-client, package-server

RMS - Resource Management Systems

SGE

- rms-tools

Additional rpm output:

```
Creating sge_root /opt/sge62 ...
Creating sge_spool /var/spool/sge62 ...

most of the files described below can be found in /opt/cluster/sge
Script to copy the tools and create the necessary links is located in /opt/sge6
Script to modify SGE-config is located in /opt/sge62/bin/noarch/modify_sge_confi
Script to modify logging-server.d is located in /opt/sge62/bin/noarch/add_logta
Don't forget to set the starter_method of all queues to /opt/sge62/3rd_party/sg
Don't forget to copy sge_request to /opt/sge62//common
init-Scripts for SGE can be found in /opt/cluster/sge/init.d
```

Chapter 6. Debugging and Error hunting

General information

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

Example 6.1. Using check_scripts.py to view running services

```
clusterserver:~ # check_scripts.py -t --mode show --server ALL
No handlers could be found for logger "initat.cluster.license_tools"
Name                  type   Thread info           status
------------------------------------------------------------
logcheck-server       server all 5 threads running running
package-server        server all 9 threads running running
mother                server 3 threads too much    dead
rrd-grapher           server not installed         skipped
rms-server            server no threads            dead
cluster-server        server 3 threads too much    dead
cluster-config-server server all 9 threads running running
host-relay            server not installed         skipped
snmp-relay            server not installed         skipped
md-config-server      server not installed         skipped
```

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 6.2. Using lse to display the last error

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

```
Found error_file /var/log/cluster/logging-server/err_py with 1879 lines
Found 50 error records
Error 50 occured today, 10:07:57, pid 1425, uid/gid is (30/8 [wwwrun/www]), sou
  0 (err) :    IOS_type             : error
  1 (err) :    args                 : None
  2 (err) :    created              : 1383901677.46
  3 (err) :    exc_info             : None
  4 (err) :    exc_text             : None
  5 (err) :    filename             : rms_views.py
  6 (err) :    funcName             : log
  7 (err) :    gid                  : 8
  8 (err) :    levelname            : err
  9 (err) :    levelno              : 40
 10 (err) :    lineno               : 44
 11 (err) :    message              : [sge] //opt/sge62/util/arch does not exis
 12 (err) :    module               : rms_views
```

```
13 (err) :    msecs                : 456.768989563
14 (err) :    msg                  : [sge] //opt/sge62/util/arch does not exis
15 (err) :    name                 : init.at.cluster.rms
16 (err) :    pathname             : /opt/python-init/lib/python2.7/site-packa
17 (err) :    pid                  : 1425
18 (err) :    ppid                 : 1360
19 (err) :    process              : 1425
20 (err) :    processName          : MainProcess
21 (err) :    relativeCreated      : 427287.201881
22 (err) :    thread               : 139873177450304
23 (err) :    threadName           : MainThread
24 (err) :    uid                  : 30
```

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 6.3. Retrieving information from nodes

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


                    alkdfldfkj
```

Important logfiles

There are two main directories **CORVUS®** writes logfiles to. `/va/log/cluster` and `/var/log/hosts`. All the logging output from the nodes is collected under /hosts/{node-name}/...

Chapter 7. 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 7.1. Monitoring configuration to check diskspace

```
$USER1$ df –e 90 –w 80
```

Chapter 8. Troubleshooting

In case something goes wrong there are a lot of logfiles you can examine in order to see what exactly was the problem. Beside the standard system logs **CORVUS®** brings its own logging infrastructure which resides in `/var/log/cluster/`.

Problems accessing the webfrontend

The webfrontend uses the following system components:

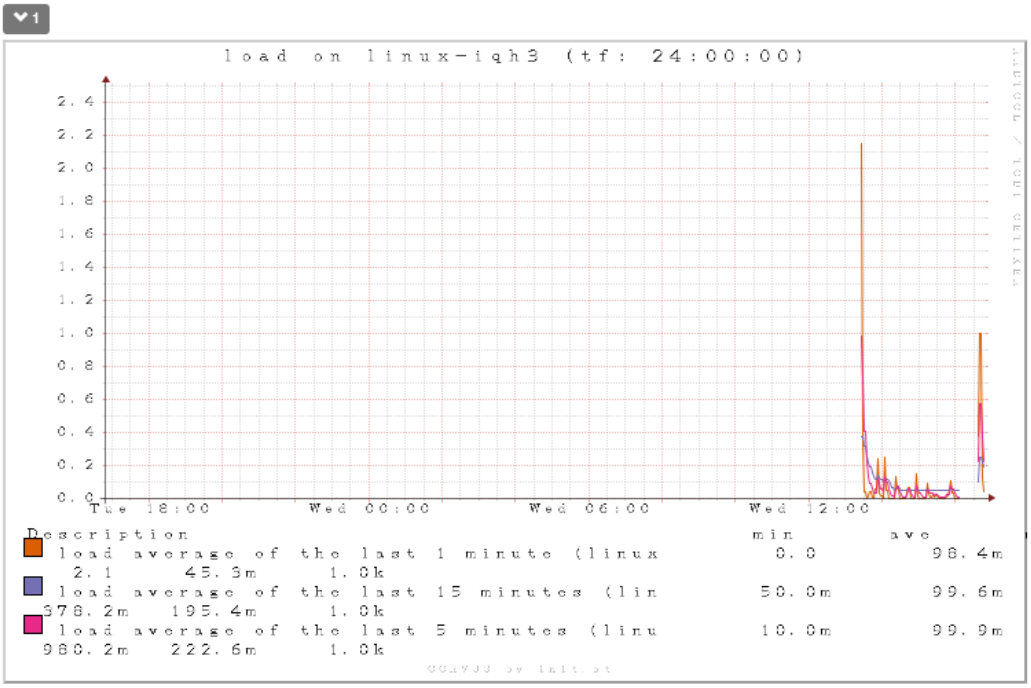
- uwsgi, logfile `/var/log/uwsgi-emperor.log` and `/var/log/cluster/django-webfrontend.log`
- nginx, logfile `/var/log/nginx/access.log` and `/var/log/nginx/error.log`
- memcached, logs via syslog (in most cases `/var/log/messages`)

The application uses to **CORVUS®** infrastructure. The logs of the various parts of the application are in `/var/log/cluster/logging-server/SERVERNAME/cluster.d`, everything unspecified goes to `/var/log/cluster/logging-server/SERVERNAME/unified`.

Chapter 9. Frequently Asked Questions

Collection of repeated Questions about **CORVUS®**

Q1: Why does my font in RRD Graphics is not readable?



A: You have to install *fetchmsttfonts* (OpenSUSE) and ... (debian) package.

Glossary