NETWAYS® LConf Documentation

Tobias Redel

NETWAYS® LConf Documentation

Tobias Redel

Publication date 2010-03-01

Table of Contents

Chapter 1. Introduction

NETWAYS® LConf is a LDAP based configuration tool for Icinga® and Nagios®. All configuration elements are stored on a LDAP server and exported to text-based configuration files. Icinga® / Nagios® uses only these config files and will work independent from the LDAP during his runtime.

This software is still under development. In the case of bugs or errors, please do not hesitate inform our team at http://www.netways.org

Chapter 2. Installation

This chapter entails the installation and software requirements of NETWAYS® LConf.

Please note that package names or installation commands vary with the different Linux distributions e.g. Debian, SuSE, and RedHat.

Requirements

The following Packages are required for successfull installation of NETWAYS® LConf.

- OpenLDAP 2.3 or higher
- Perl with ldap libs (e.g. libnet-ldap-perl and perl-LDAP)
- Operationg system's ldap utils

Getting and installing LConf

1. Download NETWAYS® LConf.

   The installation package is available on http://www.netways.org.

2. Extract the downloaded archive.

   ```
   lconf-doku:/usr/local/src# tar -xvzf LConf-0.8.tar.gz
   ```

3. Execute the install wizard

   ```
   lconf-doku:/usr/local/src/LConf-0.8# ./install.sh
   ```

4. Now, follow the wizard's instructions

5. Add a custom dir to your Icinga® / Nagios® configuration (e.g. icinga.cfg)

   ```
   cfg_dir=/usr/local/icinga/etc/MyCustomConfig
   ```

Chapter 3. Basics

Base setup

Because of completeness and security, all hosts, services and contacts generated by LConf will refer to a template. So you have to configure these base templates once. After that templates can be overridden as usual in Icinga® / Nagios®.

The config file for default templtes can be found in $PREFIX/etc/default-templates.cfg. Customize it to your personal needs.

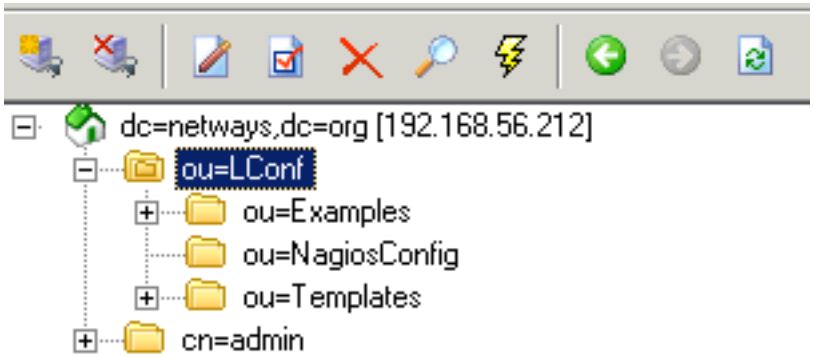
The LDAP Tree

After connecting with your LDAP browser to the LDAP Server, you could see that LConf has created different Items.

ou=LConf is LConf's base dir. Each LConf script will search below this ou.

ou=Examples contains some sample items for each ldap objectclass (see Chapter 3, 3.5 LDAP Objectclasses).

ou=Templates can contain your config templates. It's only a suggestion. Rename it as you want. Delete it. Never mind ;-)



LDAP Objectclasses

A LDAP Server is organized in attributes and object classes. An object class specifies a set of attributes used to describe an object. For example, the objectClass nagiosHost contains attributes associated with a Icinga® / Nagios® Host such as address, parent or contacts.

LConf object classes namend very visceral. For example the object class for timeperiods is called nagiosTimeperiod, the object class for commands is called nagiosCommand and so on...

The following object classes are available:

- nagiosTimeperiod
- nagiosCommand
- nagiosContact
- nagiosContactgroup
- nagiosHost
- nagiosHostgroup
- nagiosService
- nagiosStructuralObject

The special object class nagiosStructuralObject is used for attribute inheritance and will be described in Chapter 4, 2.10

Export a Icinga® / Nagios® configuration

Basicly, to generate / export a configuration vom LConf to Icinga® / Nagios® you have to run LConfExport.pl. The Script can be found in your LConf main dir.

Due to different problems with file permissions LConfExport.pl can only be executed by user you specified during installation. In this example LConf was installed with user nagios.

```
nagios@lconf-doku:/usr/local/LConf$ ./LConfExport.pl -o /usr/local/
nagios/etc/MyCustomConfig/
```

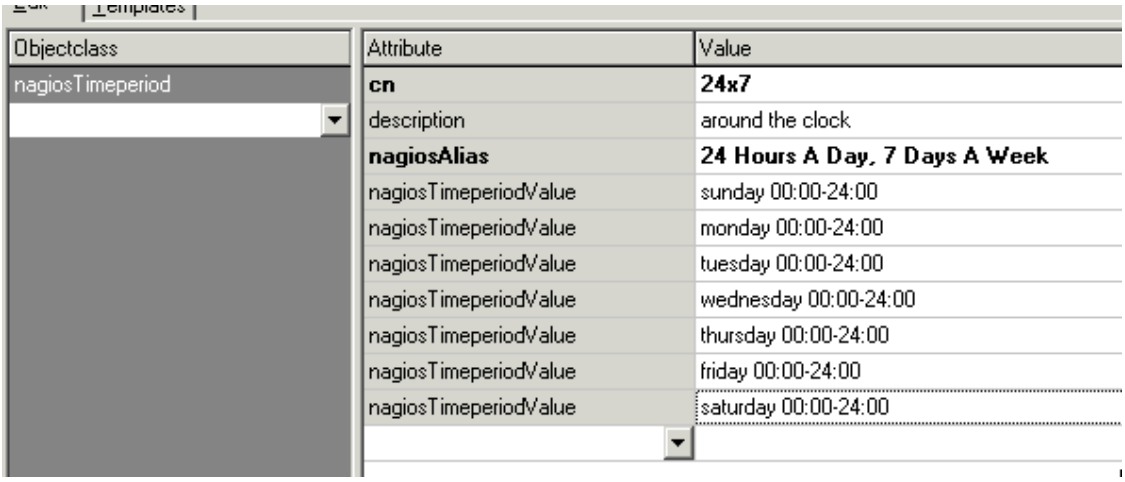
Chapter 4. The Power of LConf

Icinga® / Nagios® basic configuration

Timeperiods

For adding a new timeperiod to LConf, just create a new LDAP entry and choose object class nagiosTimeperiod. A bit special are the attributes cn and description. cn will be the title of the entry and description is an additional attribute for leaving comments. Everything else you can treat like a Icinga® / Nagios® config.

Of course, the attribute nagiosTimeperiodValue can be defined multiple times...



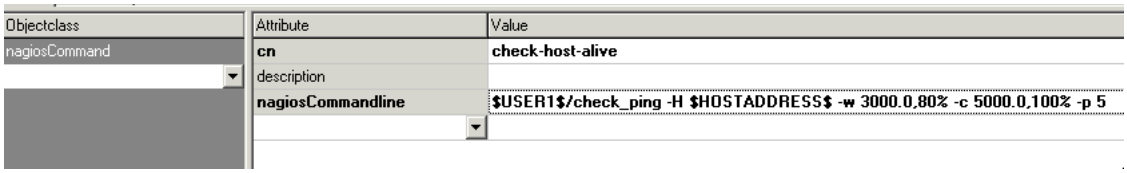
After running LConfExport.pl the config file will look like this:

```
define timeperiod {

sunday 00:00-24:00

monday 00:00-24:00

tuesday 00:00-24:00

wednesday 00:00-24:00

thursday 00:00-24:00
```

```
friday 00:00-24:00

saturday 00:00-24:00

alias 24 Hours A Day, 7 Days A Week timeperiod_name 24x7

}
```

Commands

For adding a new command to LConf, just create a new LDAP entry and choose object class nagiosCommand. A bit special are the attributes cn and description. cn will be the title of the entry and description is an additional attribute for leaving comments. Everything else you can treat like a Icinga® / Nagios® config.

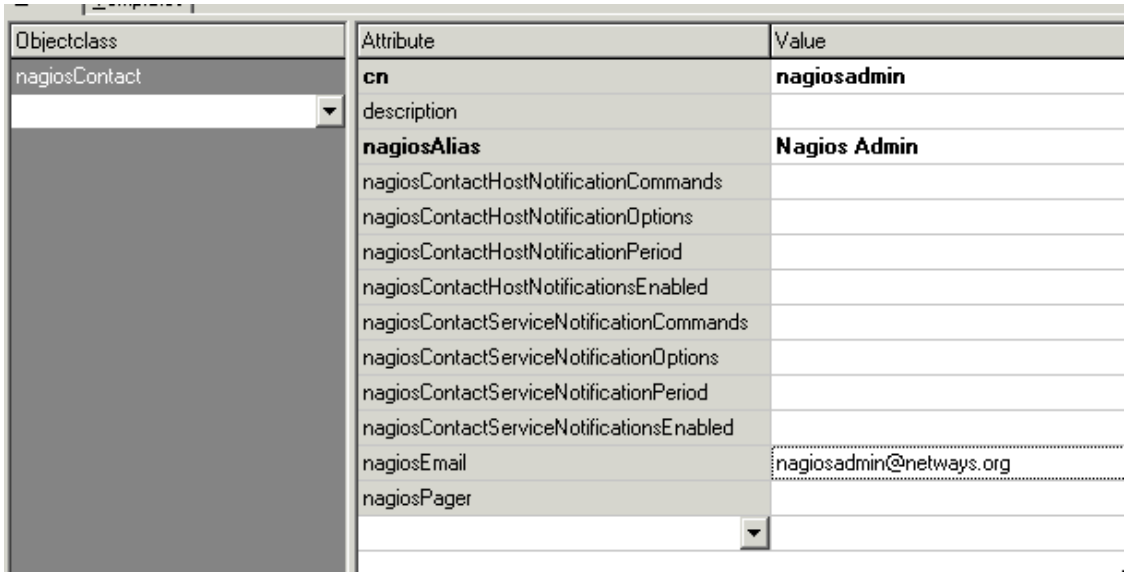


After running LConfExport.pl the config file will look like this:

```
define command {

command_line $USER1$/check_ping -H $HOSTADDRESS$ -w 3000.0,80% -c
5000.0,100% -p 5

command_name check-host-alive

}
```

Contacts

For adding a new contact to LConf, just create a new LDAP entry and choose object class nagiosContact. A bit special are the attributes cn and description. cn will be the title of the entry and description is an additional attribute for leaving comments. Everything else you can treat like a Icinga® / Nagios® config.

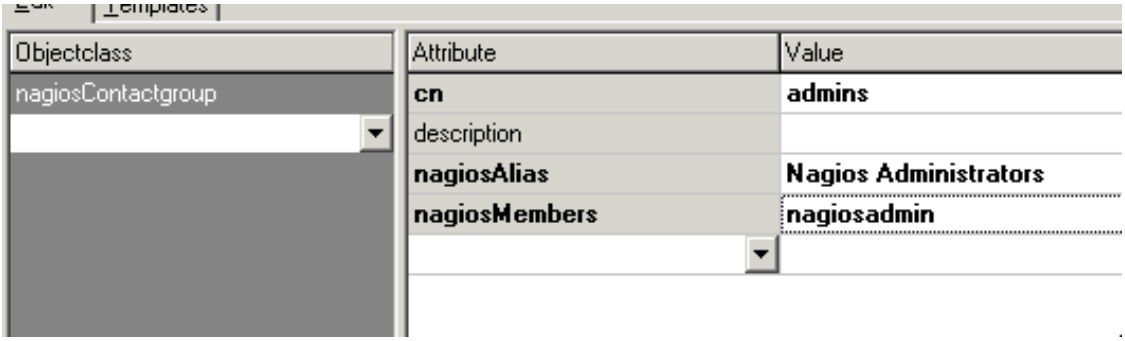


After running LConfExport.pl the config file will look like this:

```
define contact {

use generic-contact

contact_name nagiosadmin

email nagiosadmin@netways.org

alias Nagios Admin

}
```

Contactgroups

For adding a new contactgroup to LConf, just create a new LDAP entry and choose object class nagiosContactgroup. A bit special are the attributes cn and description. cn will be the title of the entry and description is an additional attribute for leaving comments. Everything else you can treat like a Icinga® / Nagios® config.

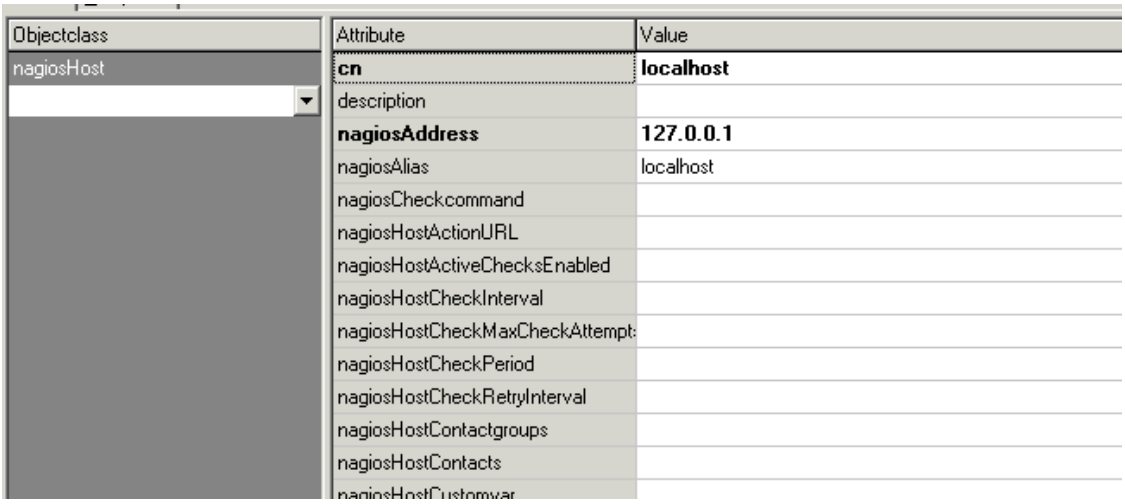


After running LConfExport.pl the config file will look like this:

```
define contactgroup {

contactgroup_name admins

alias Nagios Administrators

members nagiosadmin

}
```

Hosts

For adding a new host to LConf, just create a new LDAP entry and choose object class nagiosHost. A bit special are the attributes cn and description. cn will be the title of the entry and description is an additional attribute for leaving comments. Everything else you can treat like a Icinga® / Nagios® config.

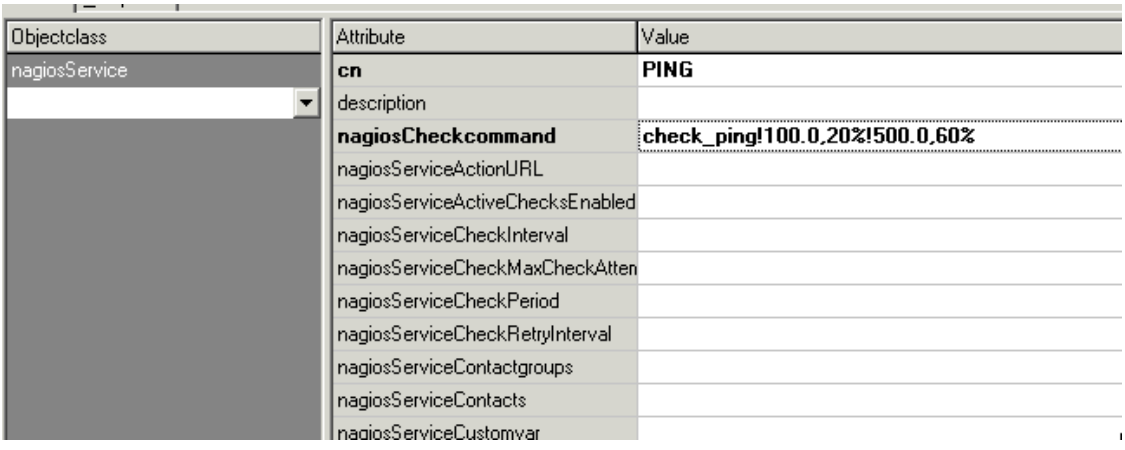


After running LConfExport.pl the config file will look like this:

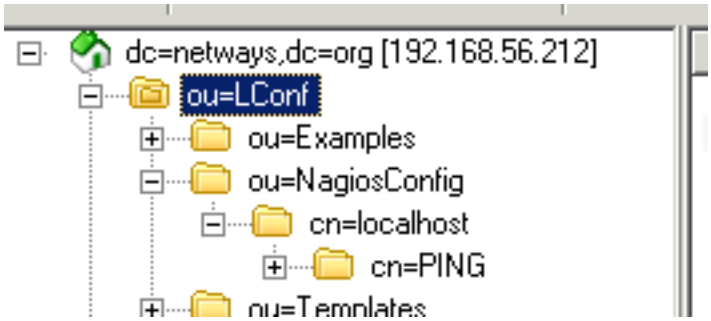
```
define host {

use generic-host

host_name localhost

alias localhost

address 127.0.0.1

}
```

Services

For adding a new service to LConf, just create a new LDAP entry and choose object class nagiosService. A bit special are the attributes cn and description. cn will be the title of the entry and description is an additional attribute for leaving comments. Everything else you can treat like a Icinga® / Nagios® config.



To assign a service to a host, simply create or move the service below a host object class.

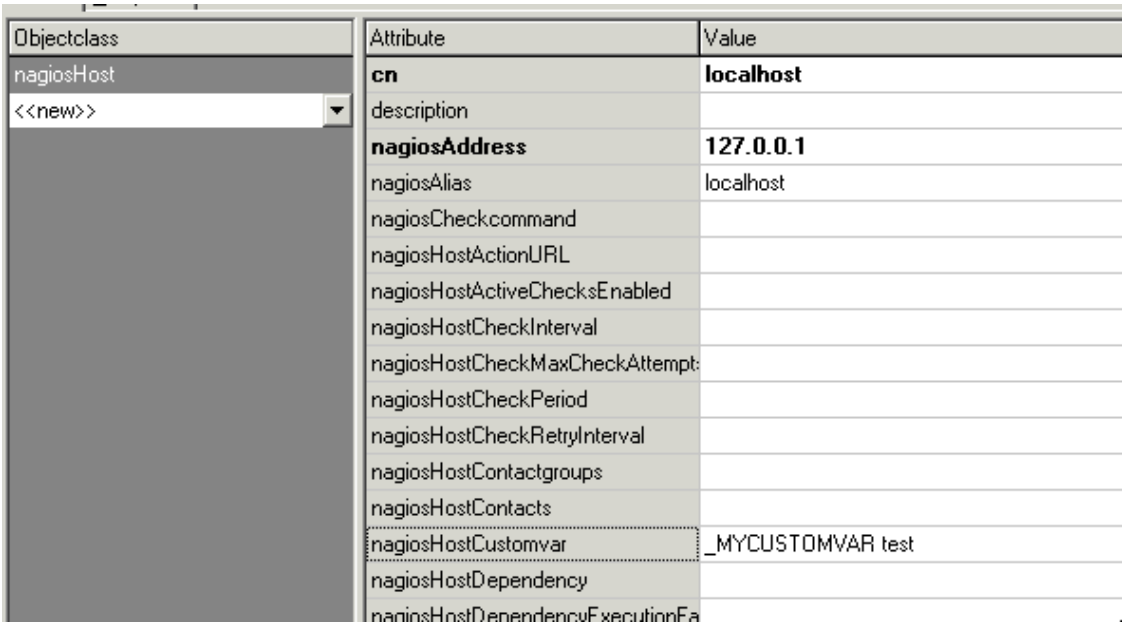


After running LConfExport.pl hosts config file will look like this:

```
define host {

use generic-host

host_name localhost

alias localhost

address 127.0.0.1

}


define service {

use generic-service

service_description PING

host_name localhost

check_command check_ping!100.0,20%!500.0,60%

}
```

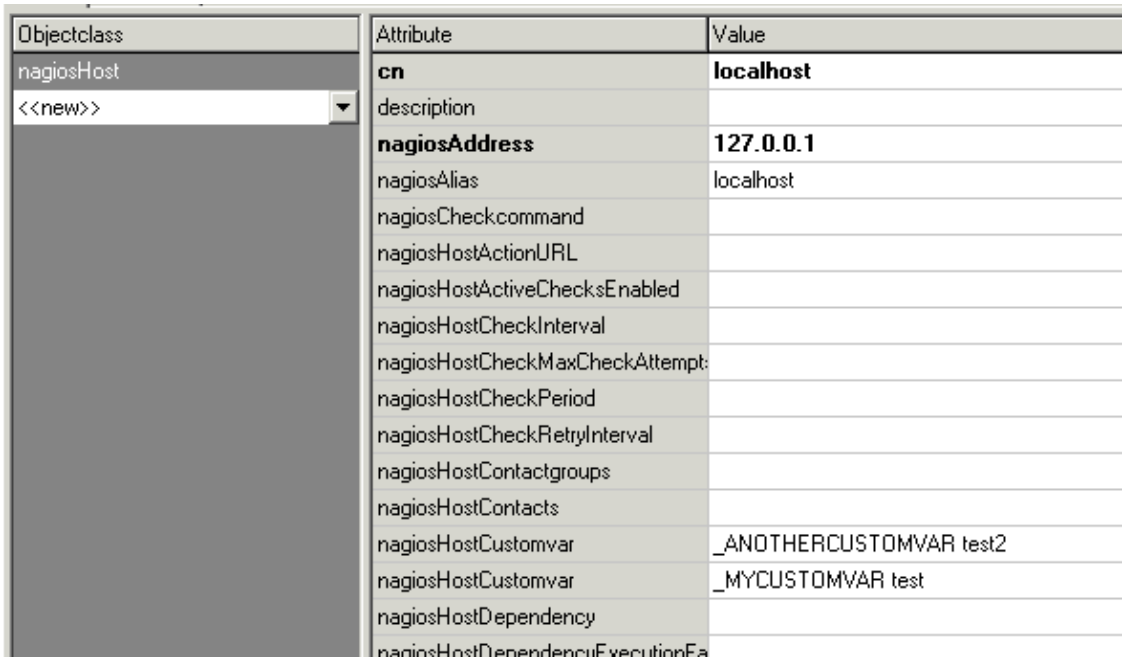
Customvars

Adding customvars to hosts or services is just as easy. For hosts, fill the nagiosHostCustomvar attribute, for services the nagiosServiceCustomvar attribute. The fist part of the line, e.g. _MYCUSTOMVAR (till a space) represents the name of the customvar, the second part (from space) will be customvars value.



Objectclass	Attribute	Value
nagiosHost	**cn**	**localhost**
<<new>>	description	
	nagiosAddress	**127.0.0.1**
	nagiosAlias	localhost
	nagiosCheckcommand	
	nagiosHostActionURL	
	nagiosHostActiveChecksEnabled	
	nagiosHostCheckInterval	
	nagiosHostCheckMaxCheckAttempt:	
	nagiosHostCheckPeriod	
	nagiosHostCheckRetryInterval	
	nagiosHostContactgroups	
	nagiosHostContacts	
	nagiosHostCustomvar	_MYCUSTOMVAR test
	nagiosHostDependency	

Of course, the attributes nagiosHostCustomvar and nagiosServiceCustomvar can be defined multiple times...



Objectclass	Attribute	Value
nagiosHost	**cn**	**localhost**
<<new>>	description	
	nagiosAddress	**127.0.0.1**
	nagiosAlias	localhost
	nagiosCheckcommand	
	nagiosHostActionURL	
	nagiosHostActiveChecksEnabled	
	nagiosHostCheckInterval	
	nagiosHostCheckMaxCheckAttempt:	
	nagiosHostCheckPeriod	
	nagiosHostCheckRetryInterval	
	nagiosHostContactgroups	
	nagiosHostContacts	
	nagiosHostCustomvar	_ANOTHERCUSTOMVAR test2
	nagiosHostCustomvar	_MYCUSTOMVAR test
	nagiosHostDependency	

After running LConfExport.pl the config file will look like this:

```
define host {

use generic-host

host_name localhost

alias localhost

address 127.0.0.1
```

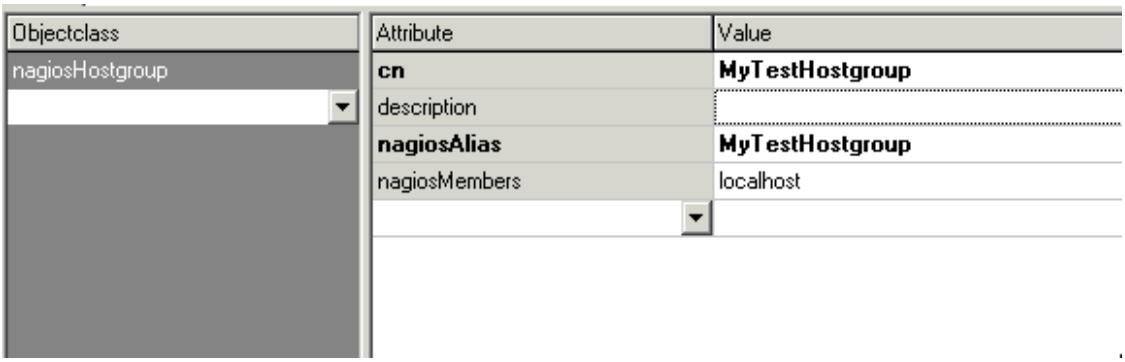
```
_ANOTHERCUSTOMVAR test2

_MYCUSTOMVAR test

}
```

Hostgroups

You can define hostgroups in two ways. To define by object class nagiosHostgroups is just as possible as define by object class nagiosHost. Do it in the way you want.

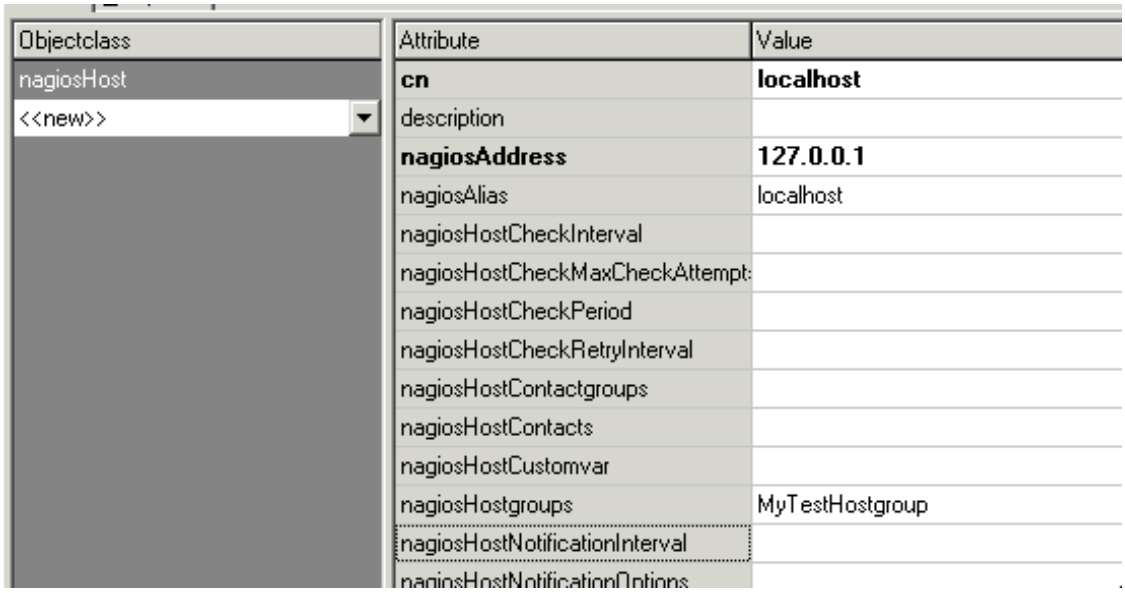
Object class nagiosHostgroup

For adding a new Hostgroup to LConf, just create a new LDAP entry and choose object class nagiosHostgroup. A bit special are the attributes cn and description. cn will be the title of the entry and description is an additional attribute for leaving comments. Everything else you can treat like a Icinga® / Nagios® config.



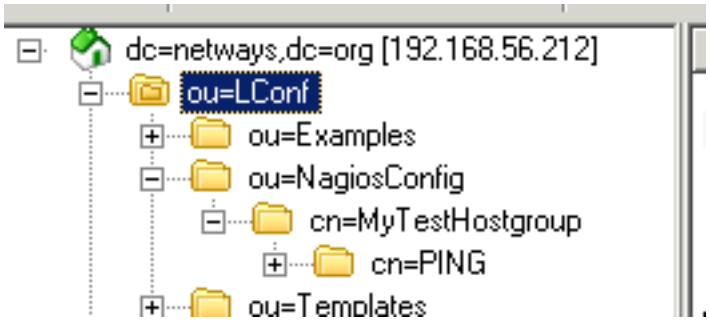
Object class nagiosHost

For adding a host to a hostgroup, just edit the existing LDAP host entry at the attribute nagiosHostgroups. You can treat the attribute like a Icinga® / Nagios® config.



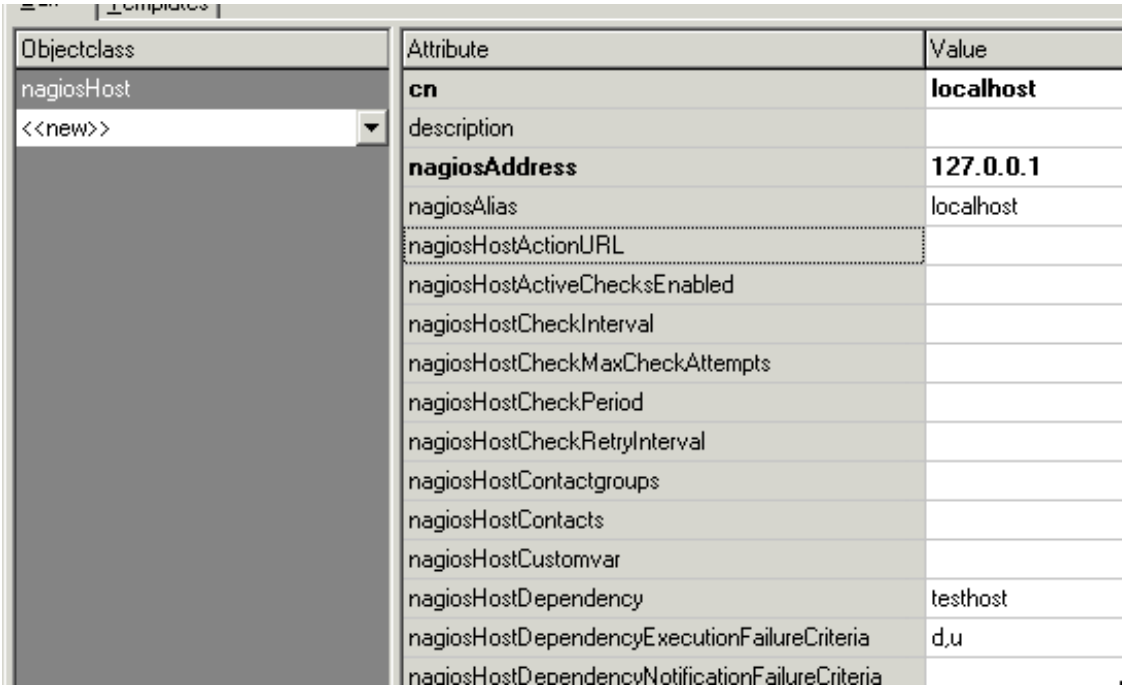
Assign a services to a hostgroup

To assign a service (see Chapter 4, 4 Services) to a hostgroup, simply create or move the service below a hostgroup object class. All hosts in this hostgroup will get this service after running LConfExport.pl.



Host Dependencies

For adding a new host dependency to LConf, just edit an existing host entry and fill the attributes nagiosHostDependency and nagiosHostDependencyExecutionFailureCriteria. The attribute nagiosHostDependencyNotificationFailureCriteria is optional.



After running LConfExport.pl hosts config file will look like this:

```
define host {

use generic-host

host_name localhost

alias localhost

address 127.0.0.1
```

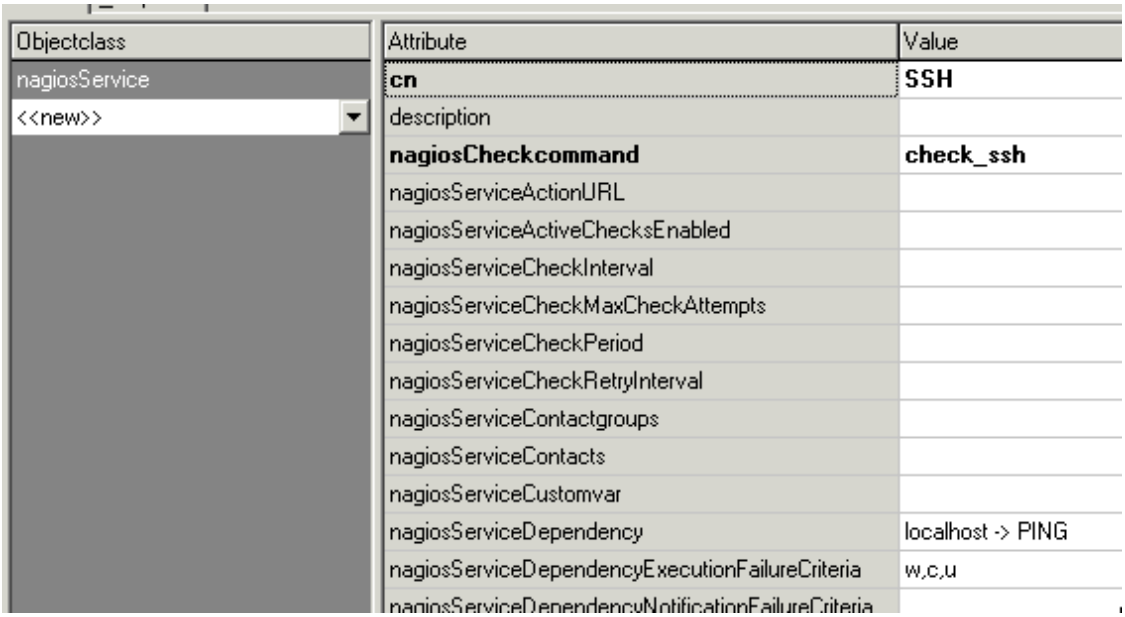
```
}

define hostdependency {

host_name testhost

dependent_host_name localhost

execution_failure_criteria d,u

}
```

Service Dependencies

For adding a new service dependency to LConf, just edit an existing service entry and fill the attributes nagiosServiceDependency and nagiosServiceDependencyExecutionFailureCriteria. The attribute nagiosServiceDependencyNotificationFailureCriteria is optional.

The attribute nagiosServiceDependency is a bit special. You have to specify it in the way HOSTNAME -> SERVICE

The sign -> will act as separator between host and service. In the example below it means: SSH depends on service PING of localhost.



Objectclass	Attribute	Value
nagiosService	**cn**	**SSH**
<<new>>	description	
	nagiosCheckcommand	**check_ssh**
	nagiosServiceActionURL	
	nagiosServiceActiveChecksEnabled	
	nagiosServiceCheckInterval	
	nagiosServiceCheckMaxCheckAttempts	
	nagiosServiceCheckPeriod	
	nagiosServiceCheckRetryInterval	
	nagiosServiceContactgroups	
	nagiosServiceContacts	
	nagiosServiceCustomvar	
	nagiosServiceDependency	localhost -> PING
	nagiosServiceDependencyExecutionFailureCriteria	w,c,u
	nagiosServiceDependencyNotificationFailureCriteria	

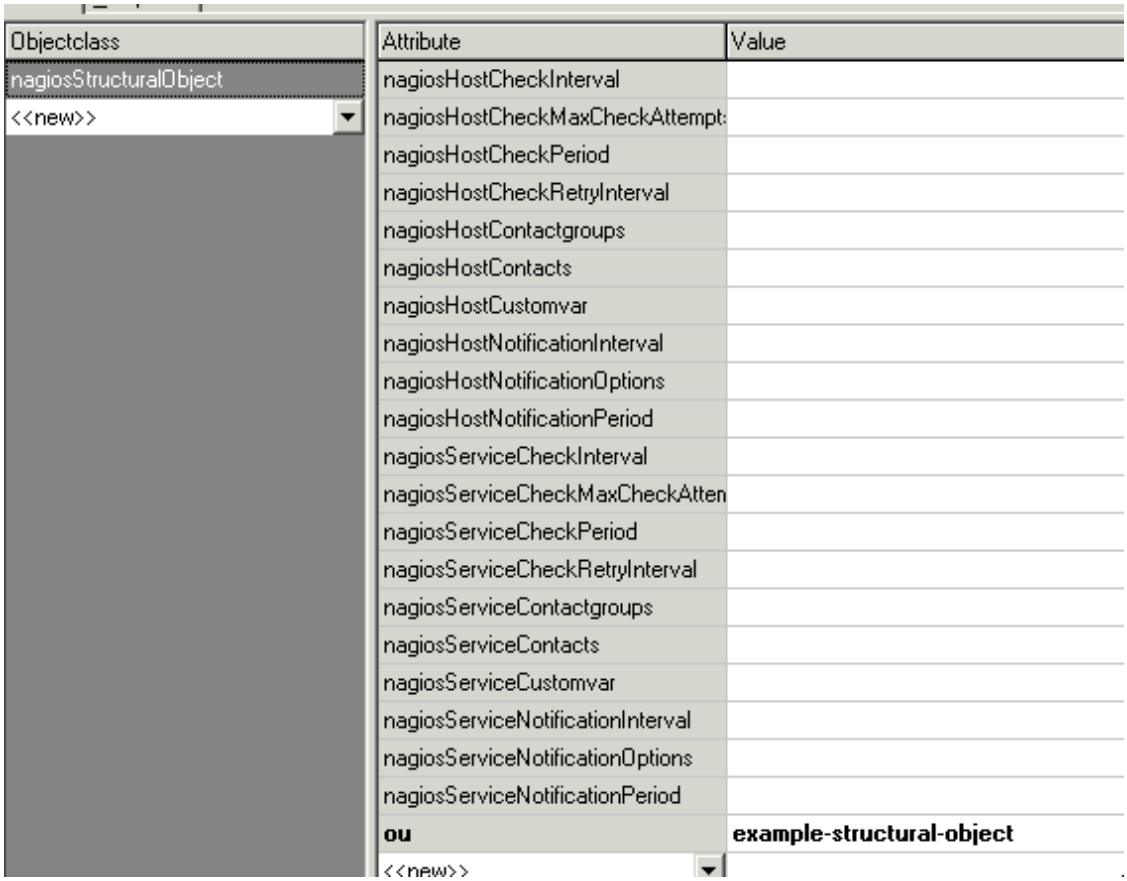
After running LConfExport.pl hosts config file will look like this:

```
define host {

use generic-host

host_name localhost

alias localhost
```

```
address 127.0.0.1
}

define service {
use generic-service
service_description PING
host_name localhost
check_command check_ping!100.0,20%!500.0,60%
}

define service {
use generic-service
service_description SSH
host_name localhost
check_command check_ssh
}

define servicedependency {
host_name localhost
service_description PING
dependent_host_name localhost
dependent_service_description SSH
execution_failure_criteria w,c,u
}
```

Inheritance

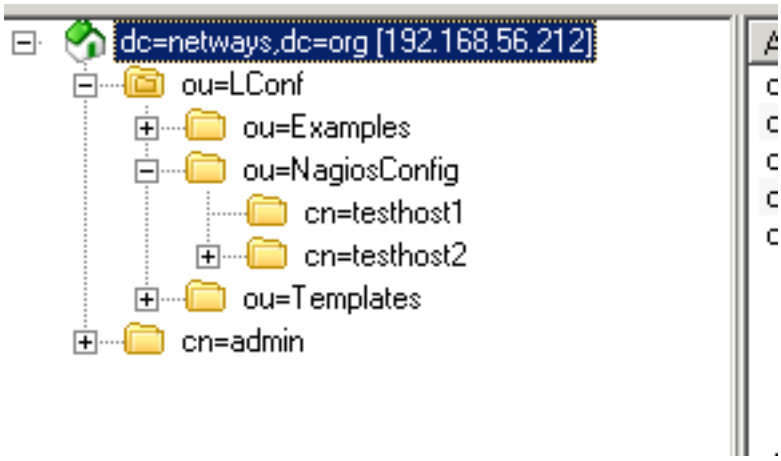
With LConf it is possible to inherit host- and service attributes or complete service objects down the LDAP tree. Each object class nagiosStructuralObject can be used for inheritance. An object class nagiosStructuralObject looks like this:



As you can imagine, *Host* attributes are used for host inheritance and *Service* attributes for service inheritance. For more detailed information about host- and service inheritance, read below...

Host attributes

Starting point are the two hosts testhost1 and testhost2.

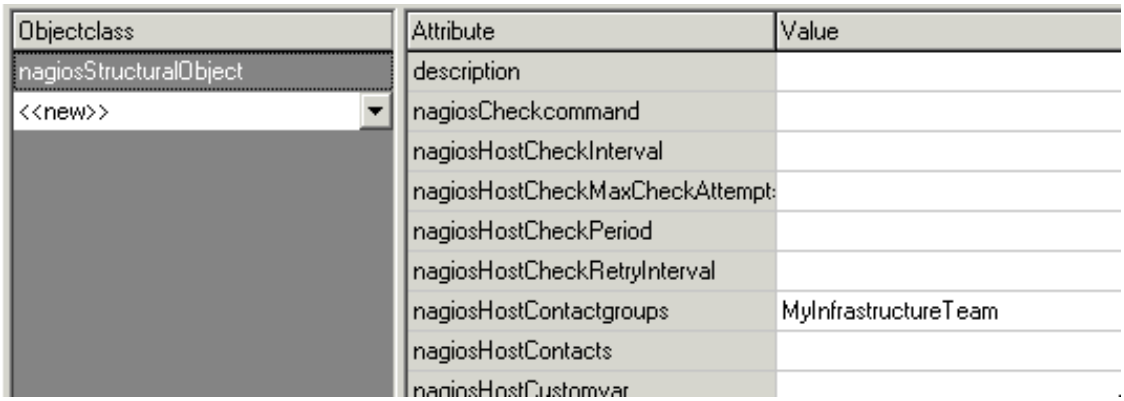


For this scenario, the generated config file will look like this:

```
define host {

use generic-host
```

```
host_name testhost1

alias testhost1

address 127.0.0.1

}


define host {

use generic-host

host_name testhost2

alias testhost2

address 127.0.0.1

}
```

To assign a host attribute to both hosts you have to edit the nagiosStructuralObject which is the lowest common denominator. In our simple example it's ou=NagiosConfig. For example, we want to assign the contactgroup MyInfrastructureTeam to both hosts, it will look like this in the LDAP editor:



After running LConfExport.pl the config will look like this:

```
define host {

use generic-host

host_name testhost1

alias testhost1

# from: ou=NagiosConfig,ou=LConf,dc=netways,dc=org

contact_groups MyInfrastructureTeam

address 127.0.0.1

}
```

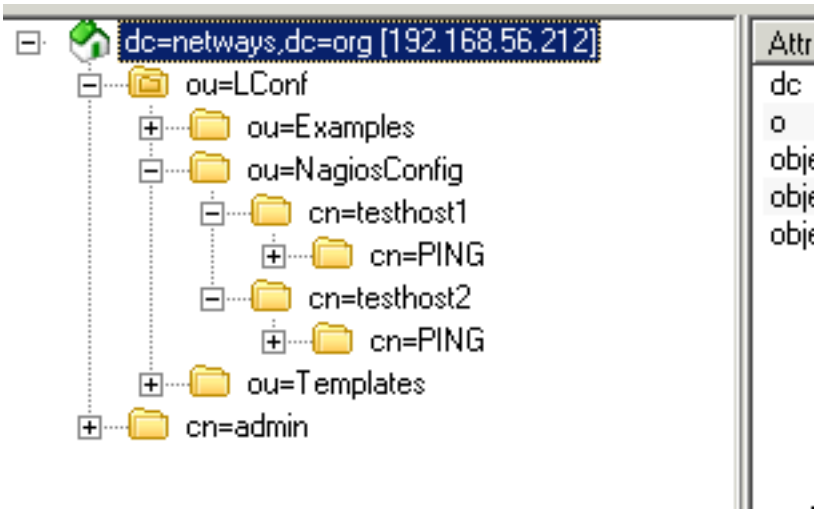
```
define host {

use generic-host

host_name testhost2

alias testhost2

# from: ou=NagiosConfig,ou=LConf,dc=netways,dc=org

contact_groups MyInfrastructureTeam

address 127.0.0.1

}
```

Of course, you can treat the attributes like a Icinga® / Nagios® config. Comma seperated lists like MyInfrastructureTeam,SecondLevel,Management are also possible.

As you have probably noticed, LConf adds a comment line above all inherited attributes. This comment tells you where you can find the inheritance. This will help if your config grows.

Service attributes

Starting point are the two hosts testhost1 and testhost2. For each host a service PING is defined.

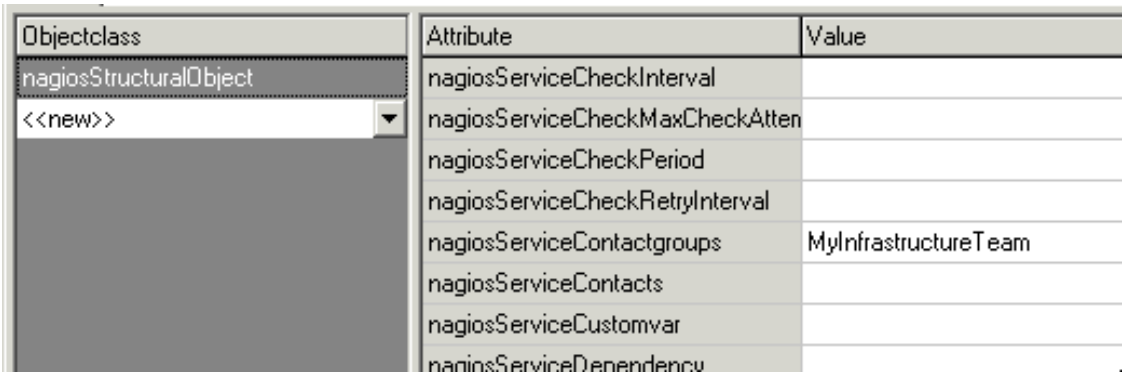


For this scenario, the generated config file will look like this:

```
define host {

use generic-host

host_name testhost1

alias testhost1

address 127.0.0.1

}
```

```
define service {
use generic-service
service_description PING
host_name testhost1
check_command check_ping!100.0,20%!500.0,60%
}

define host {
use generic-host
host_name testhost2
alias testhost2
address 127.0.0.1
}

define service {
use generic-service
service_description PING
host_name testhost2
check_command check_ping!100.0,20%!500.0,60%
}
```

To assign a service attribute to both PING services you have to edit the nagiosStructuralObject which is the lowest common denominator. In our simple example it's ou=NagiosConfig. For example, we want to assign the contactgroup MyInfrastructureTeam to both PING services, it will look like this in the LDAP editor:



After running LConfExport.pl the config will look like this:

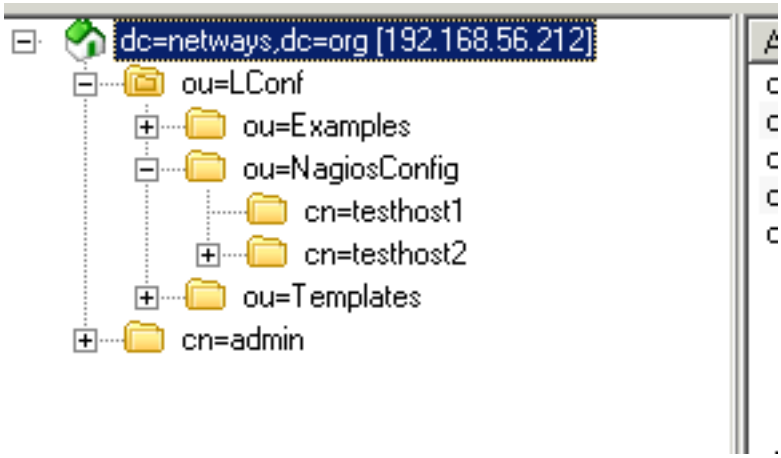
```
define host {
use generic-host
host_name testhost1
alias testhost1
address 127.0.0.1
}

define service {
use generic-service
service_description PING
host_name testhost1
# from: ou=NagiosConfig,ou=LConf,dc=netways,dc=org
contact_groups MyInfrastructureTeam
check_command check_ping!100.0,20%!500.0,60%
}

define host {
use generic-host
host_name testhost2
alias testhost2
address 127.0.0.1
}

define service {
use generic-service
service_description PING
host_name testhost2
# from: ou=NagiosConfig,ou=LConf,dc=netways,dc=org
contact_groups MyInfrastructureTeam
check_command check_ping!100.0,20%!500.0,60%
}
```

Of course, you can treat the attributes like a Icinga® / Nagios® config. Comma seperated lists like MyInfrastructureTeam,SecondLevel,Management are also possible.

As you have probably noticed, LConf adds a comment line above all inherited attributes. This comment tells you where you can find the inheritance. This will help if your config grows.

Services

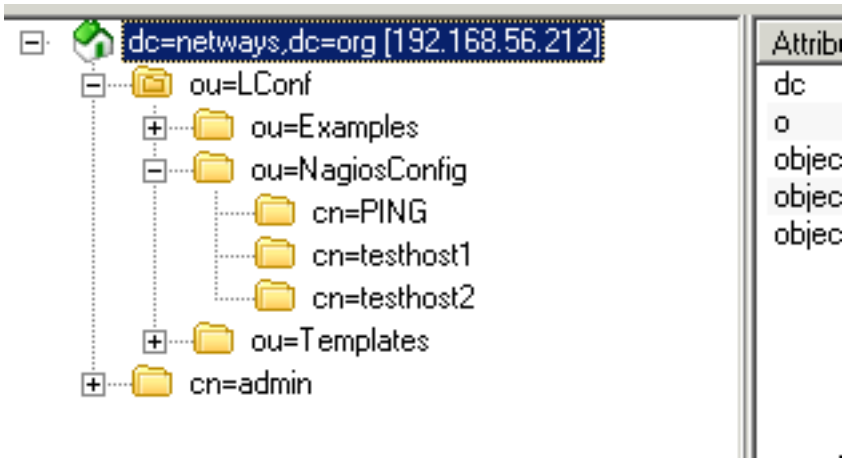
Starting point are the two hosts testhost1 and testhost2.



For this scenario, the generated config file will look like this:

```
define host {

use generic-host

host_name testhost1

alias testhost1

address 127.0.0.1

}


define host {

use generic-host

host_name testhost2

alias testhost2

address 127.0.0.1

}
```

To assign a service to both hosts you have to create an object class nagiosService below the nagiosStructuralObject which is the lowest common denominator. In our simple example it's ou=NagiosConfig. For example, we want to assign a service PING to both hosts, it will look like this in the LDAP editor:



After running LConfExport.pl the config will look like this:

```
define host {
use generic-host
host_name testhost1
alias testhost1
address 127.0.0.1
}

# from: cn=PING,ou=NagiosConfig,ou=LConf,dc=netways,dc=org
define service {
use generic-service
service_description PING
host_name testhost1
check_command check_ping!100.0,20%!500.0,60%
}

define host {
use generic-host
host_name testhost2
alias testhost2
address 127.0.0.1
}

# from: cn=PING,ou=NagiosConfig,ou=LConf,dc=netways,dc=org
define service {
```

use generic-service

service_description PING

host_name testhost2

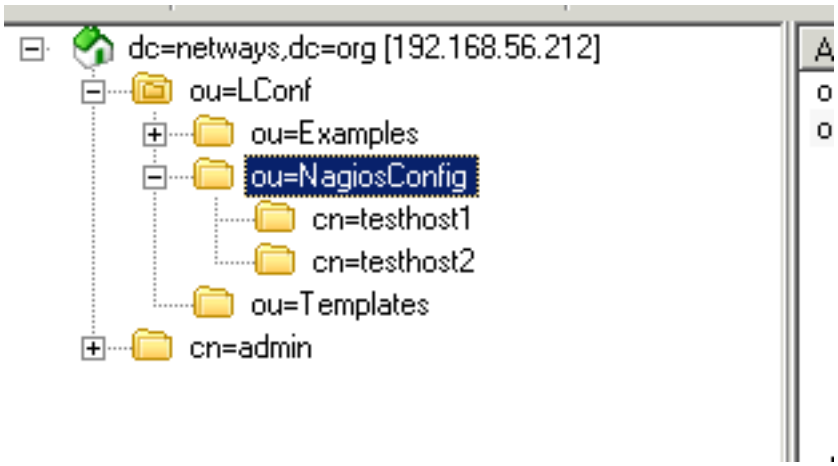
check_command check_ping!100.0,20%!500.0,60%

}

As you have probably noticed, LConf adds a comment line above all inherited services. This comment tells you where you can find the inherited service. This will help if your config grows.

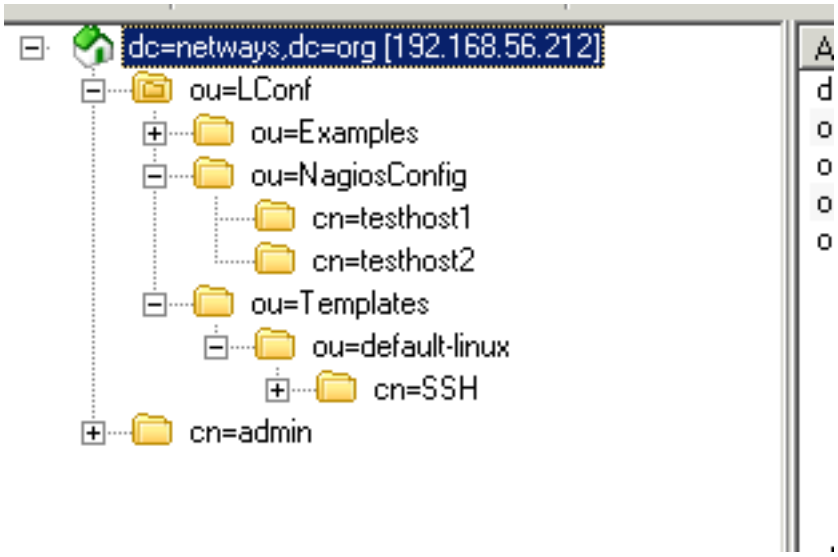
Templates / Profiles

Templates or Profiles. Name it the way you want... ;-) Here we go!



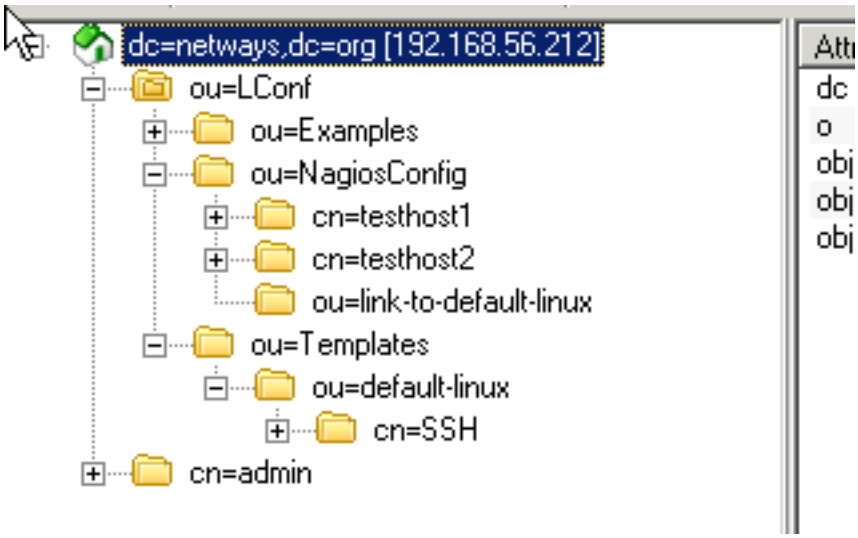
For both hosts are no services specified at the moment. First of all, we will create a template default-linux. For this purpose we create an object class nagiosStructuralObject below ou=Templates and add a service SSH (adding services, see Chapter 4, 4 Services) to default-linux.

This should look like this:

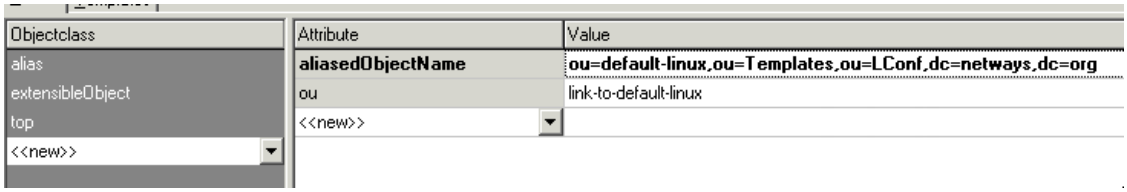


Now, our template default-linux contains a SSH check. To link a template to a list of hosts we need a LDAP alias. An example of a LDAP alias could be found in ou=Examples. Copy the alias to ou=NagiosConfig (the alias will be inherited to testhost1 and testhost2) and rename it the way you want.

After that, the LDAP tree should look like this:



At this point we have to edit the target of the alias. In our example the target should be ou=default-linux,ou=Templates,ou=LConf,dc=netways,dc=org



During config export with LConfExport.pl, the script will find the alias link-to-default-linux with it's target ou=default-linux and will assign all services below ou=default-linux to testhost1 and testhost2.

After running LConfExport.pl hosts config file will look like this:

```
define host {

use generic-host

host_name testhost1

alias testhost1

address 127.0.0.1

}


#                       from:                       cn=SSH,ou=default-
linux,ou=Templates,ou=LConf,dc=netways,dc=org

define service {

use generic-service
```

```
service_description SSH

host_name testhost1

check_command check_ssh

}


define host {

use generic-host

host_name testhost2

alias testhost2

address 127.0.0.1

}


#                       from:                       cn=SSH,ou=default-
linux,ou=Templates,ou=LConf,dc=netways,dc=org

define service {

use generic-service

service_description SSH

host_name testhost2

check_command check_ssh

}
```

nagiosHostDisable / nagiosServiceDisable

To disable a host and/or a service simply fill the attributes nagiosHostDisable or nagiosServiceDisable with any text. If LConf detects during config export that a disable attribute is set (not empty and not 0), the host and/or service will be excluded.

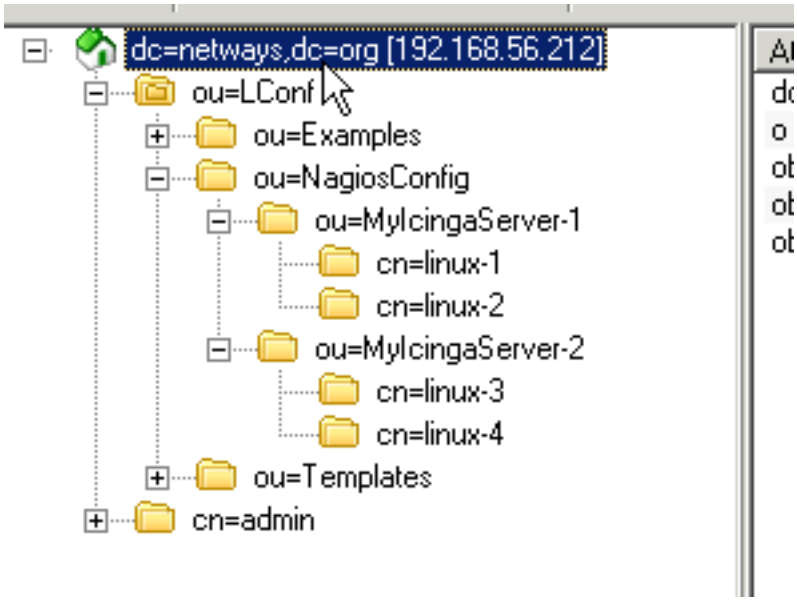
Of course you can find nagiosHostDisable and nagiosServiceDisable in object class nagiosStructuralObject, too. So inheritance is also possible.

Chapter 5. Multiple Icinga® / Nagios® Servers

Managing multiple Nagios® / Icinga® servers with LConf is not difficult. Starting point is this scenario...

We have two Icinga servers, called MyIcingaServer-1 and MyIcingaServer-2. Both are monitoring different linux servers.

The "folders" MyIcingaServer-1 and MyIcingaServer-2 are nagiosStructuralObject object classes. They are to form a structure...



Export the whole Icinga® / Nagios® configuration

First of all you have to export the config in a folder. We'll take /usr/local/icinga/etc/MyCustomConfig in this example.

```
nagios@lconf-doku:/usr/local/LConf$ ./LConfExport.pl -o /tmp/LConf
```

Now, the configuration of both servers is stored in /tmp/LConf.

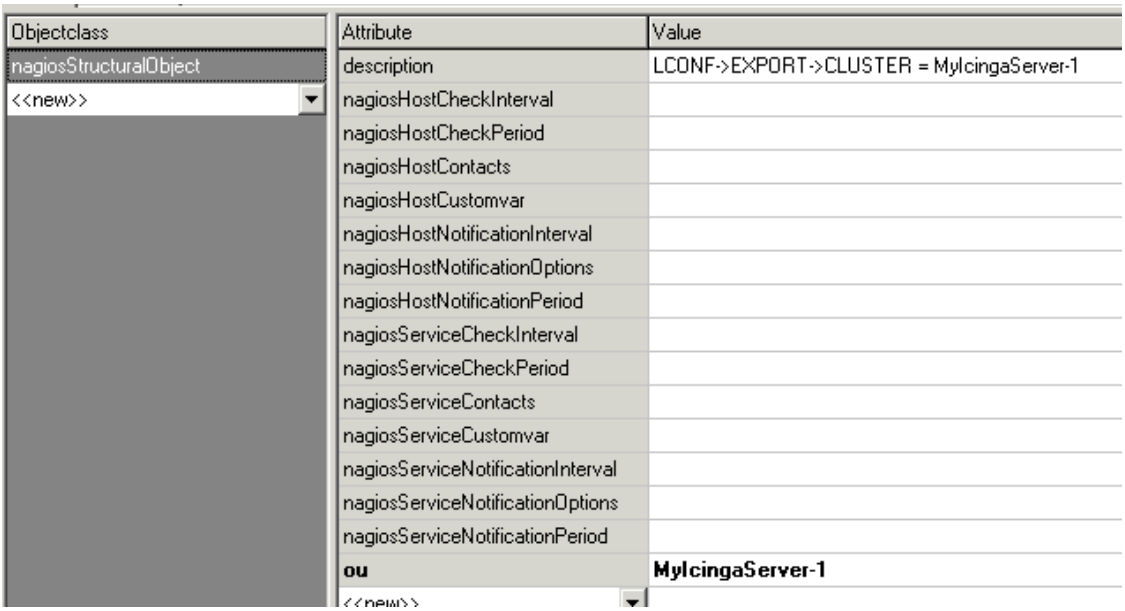
By the way... this config could be used to create a master / slave environment :-)

Tell LConf about multiple servers

Due to editing of the suitable nagiosStructuralObject, LConf knows which part of the config should be exported to each host.

As you can see below, you have to enter the information into a description line. The description attribute can be specified multiple times.

It's important to write the right syntax into attributes line. LCONF->EXPORT->CLUSTER = <name of my server>.



If you have a structural object which should be exported to all servers (e.g. one that includes commands, contacts, ...), simply describe it with LCONF->EXPORT->GLOBAL.

Deploy config per server

To export each Icinga servers config, LConfSlaveExport is your friend!

```
nagios@lconf-doku:/usr/local/LConf$ ./LConfSlaveExport.pl -H
MyIcingaServer-1 -s /tmp/LConf -t /usr/local/icinga/etc/MyConfig
```

The script will now search after descriptions like LCONF->EXPORT->GLOBAL and LCONF->EXPORT->CLUSTER = MyIcingaServer-1 and will push the config to MyIcingaServer-1 via SSH and SCP (ssh pub key exchange needed).

LConfSlaveExport.pl can also reload Icinga® / Nagios® after config deployment. This feature is disabled per default, because not every admin is happy about it. To enable the feature you have to edit LConfSlaveExport.pl.

There are the following lines:

```
# reload nagios on slave

#beVerbose("RELOAD", "ssh $optSlave '/etc/init.d/icinga reload'");

#my $result = qx(ssh $optSlave '/etc/init.d/icinga reload');

#print $result;
```

Uncomment them and LConfSlaveExport.pl will reload Icinga® / Nagios®