

The LISa Handbook

Alexander Neundorf



The LISa Handbook

Contents

1	Introduction	1
2	How it works	2
3	resLISa	4
4	The Configuration File	5
4.1	PingAddresses	5
4.2	PingNames	6
4.3	AllowedAddresses	6
4.4	BroadcastNetwork	7
4.5	SearchUsingNmblookup	7
4.6	FirstWait	7
4.7	SecondWait	7
4.8	UpdatePeriod	7
4.9	DeliverUnnamedHosts	8
4.10	MaxPingsAtOnce	8
4.11	Three more example files	9
5	Command Line Options and Other Usage	11
6	Credits and Licenses	12
A	Installation	13
A.1	Other Requirements	13

Abstract

LISa is intended to provide a kind of 'network neighborhood', but only relying on the TCP/IP protocol stack, no SMB or anything else required.

This is the handbook to both the LAN Information Server (LISa) and the Restricted LAN Information Server (resLISa)

Chapter 1

Introduction

LISa is intended to provide a kind of 'network neighborhood', but only relying on the TCP/IP protocol stack, no smb or whatever.

It is completely independent from KDE/Qt™.

The list of running hosts is provided via TCP port 7741.

LISa supports two ways of finding hosts:

1. You give LISa a range of IP addresses, then LISa will send ICMP echo requests to all given IP addresses, and wait for the answers.
2. You can tell LISa to execute **nmblookup** "*". The command line tool **nmblookup** must be installed from the Samba package. **nmblookup** "*" sends a broadcast to the attached networks, and all hosts running SMB services will answer this broadcast.

Chapter 2

How it works

In the configuration file you provide a range of IP-addresses which LISa should check to see whether they are running.

In the most simple case this could be your network address/subnetmask, then LISa would check every possible host of your network to see if it is running.

The hosts are checked using ICMP echo requests. To be able to send and receive ICMP echo requests and replies the program has to open a so-called 'raw socket'. Therefore it needs `root` privileges. This socket is opened right after the start of the program, after successfully opening the socket root privileges are dropped immediately (see `main.cpp` and `strictmain.cpp`).

If you configure LISa so that it also uses **nmblookup**, it will `popen("nmblookup\n*\n")` and then parse the results.

Since the ICMP requests and the broadcasts can cause some network traffic if there are more than one such server running in one network, the servers cooperate with each other. Before they start pinging (or **nmblookup**), they send a broadcast on port 7741.

If somebody answers this broadcast, they will retrieve the complete list of running hosts via TCP port 7741 from this host and will not start to ping (or **nmblookup**).

If nobody answers, the host which sent the broadcast will start pinging the hosts (or **nmblookup**) and then open a socket which listens for the mentioned broadcasts. If the host received an answer to his broadcast, it won't have the socket for listening to the broadcasts open. So usually exactly one of the servers will have this socket open and only this one will actually ping (or **nmblookup**) the hosts.

In other words, the servers are lazy, they work like 'I will only do something if nobody else can do it for me'.

There is another feature which reduces the network load.

Let's say you configured LISa to update every 10 minutes. Now you don't access your server very often. If nobody accesses the server for the last update

The LISa Handbook

period, the server will update (either itself or from the one which actually does the work) and then double its update period, i.e. the next update will happen after 20 minutes.

This will happen 4 times, so if nobody accesses the server with update period 10 minutes for a long time, its update interval will increase up to 160 minutes, almost three hours. If then somebody accesses the data from the server, he will get an old list (up to 160 minutes old). With accessing the server will reset its update interval to its initial value, i.e. 10 minutes and immediately start updating if the last update is more than these 10 minutes over. This means if you get a very old list, you can try some seconds later again and you should get a current version.

This will have fast effect for the servers, which don't ping (or `nmblookup`) theirselves, since only one user usually accesses them, and it will have less effect for the server which does the pinging (or `nmblookup`), since this server is accessed from all other servers in the network.

This way it is possible that many hosts in a network run this server, but the net load will remain low. For the user it is not necessary to know wether there is a server (i.e. a name server or fileserver or whatever) in the network which also runs LISa. He can always run LISa locally and LISa will detect if there is one, transparently to the user.

The first client for LISa is an `ioslave` for KDE 2, so the user can enter there `lan://localhost/` or `lan:/`, which will both contact LISa on the own system.

If there is a machine which runs all the time and the user knows that this machine also runs LISa, he can use his LISa client directly with this server (would be with the mentioned `ioslave lan://the_server_name/`).

If you don't want that your LISa takes part in the broadcasting, but always does the pinging itself, make it use another port with the command line option `--port` or `-p`. This is not recommended!

If you send **SIGHUP** to LISa, it will reread its configfile. If you send **SIGUSR1** to LISa, it will print some status information to `stdout`.

The data provided over the socket has a simple format: <decimal ip address in network byte order><one space 0x20><full name of the host><a terminating '\0'><newline '\n'> and the last line 0 succeeded<' \n'>

For example,

```
17302538 some_host.whatever.de
18285834 linux.whatever.de
17827082 nameserver.whatever.de
0 succeeded
```

This should make it easy parseable.

If there are very strict security rules in your network, some people might consider the pinging a potential attack. If you have problems with this, try the restricted version, `resLISa`.

Chapter 3

resLISa

If you have very strict security rules in your network or you don't want to have another port open or whatever, you can use resLISa.

With resLISa you can't ping whole networks and address ranges, you can give resLISa up to currently 64 hosts by their names in its config file. These will be pinged. You are still able to use **nmblookup**.

resLISa will also only provide the information over a unix domain socket, i.e. not over the network. The name of the socket is `/tmp/resLisa-YourLoginname` so resLISa can be safely run by more users on one machine.

Since it should also not produce a security risk of any kind it is safe to install resLISa `setuid root`. `root` privileges will be dropped right after startup (see `strictmain.cpp`), they are only needed to create a raw socket for sending the ICMP echo requests.

It will also not send or receive broadcasts. The first client for this is also an `ioslave` for KDE 2 (`r1an:/` in Konqueror for example.)

Chapter 4

The Configuration File

Now an example config file:

```
PingAddresses = ↔
    192.168.100.0/255.255.255.0;192.168.100.10-192.168.199.19;192.168.200.1;192-192.168-1

PingNames = bb_mail;
AllowedAddresses = 192.168.0.0/255.255.0.0
BroadcastNetwork = 192.168.100.0/255.255.255.0
SearchUsingNmblookup = 1                #also try nmblookup
FirstWait = 30                          #30 hundredth seconds
SecondWait = -1                          #only one try
#SecondWait = 60                         #try twice, and the ↔
    second time wait 0.6 seconds
UpdatePeriod = 300                       #update after 300 ↔
    secs
DeliverUnnamedHosts = 0                  #don't publish hosts ↔
    without name
MaxPingsAtOnce = 256                     #send up to 256 ICMP ↔
    echo requests at once
```

4.1 PingAddresses

This is probably the most important entry.

Here you say which addresses will be pinged. You can specify multiple ranges, they are divided by semicolons.

There are four possible ways to define addresses:

net address/network mask 192.168.100.0/255.255.255.0, i.e. an IP address and the assigned network mask.

The LISa Handbook

This doesn't have to be the network address and netmask of your machine. For example, if you have 10.0.0.0/255.0.0.0 as your own address, you could specify 10.1.2.0/255.255.255.0 if you are only interested in these addresses. The combination IP address-network mask must be divided by a slash '/' and the address does not have to be a real network address, it can also be a host address of the desired network, i.e. 10.12.34.67/255.0.0.0 is the same as 10.0.0.0/255.0.0.0 .

a range of IP addresses For example: 192.168.100.10-192.168.199.19

An IP-address where pinging will start and an IP-address where pinging will end.

Both addresses must be divided by a '-'.

In this example this would produce $199-100+1=100$, $100*256=25.600$, $25.600+(19-10+1)=25.590$ addresses

An IP address, as represented by ranges of each of the four decimal numbers

An IP address can be represented by its four decimal numbers, and you can specify ranges for each of these four numbers: 192-192.169-171.100-199.0-9

In this example all IP addresses with first number 192, second number from 168 to 168, third number from 100 up to 199 and last number from 0 up to 9 will be pinged. This would give $1*1*100*10=1.000$ addresses.

This is probably only useful in very seldom cases. Here you have to provide ranges for every four numbers, always divided by '-'.

Single IP addresses or host names The IP address or host name of any machine you are particularly interested in.

It is also valid to leave this entry empty.

4.2 PingNames

Here you can additionally specify hosts to ping using their names. The names have to be divided by semicolons.

It is also valid to leave this entry empty.

4.3 AllowedAddresses

This is also very important. LISa will only ping addresses, accept clients and answer broadcasts from addresses, which are covered by the addresses given in this line. You can add up to 32 network addresses/network masks or single addresses. Divide them by ; and don't put empty space between the addresses!

For example, 192.168.0.0/255.255.0.0;192.169.0.0

A complete network and a single address are valid. Always make this as strict as possible, usually your network address/subnetmask is a good choice.

4.4 BroadcastNetwork

This entry contains exactly one network address/subnet mask. To this network broadcasts will be sent. Usually this should be your own network address/-subnetmask, for example: 192.168.0.0/255.255.0.0

4.5 SearchUsingNmblookup

Here you can give *0* or *1*. *1* means that LISa will execute **nmblookup** "*" and parse the output from this command. This produces less network traffic than the ping, but you will only get hosts which have a SMB service running (Windows® machines or machines running samba).

If you enable this option and also give IP addresses to ping, then **nmblookup** will be executed first and then the ping will start. Then only addresses will be pinged, which were not already delivered from **nmblookup**. This should slightly decrease the network load.

4.6 FirstWait

If LISa pings, i.e. if it sends the ICMP echo requests, it sends a bunch of requests at once, and then it will wait for the number of hundredth seconds you specify here. Usually values from 5 to 50 should be good, the maximum is 99 (gives 0.99 seconds, a very long time). Try to make this value as small as possible while still finding all running hosts.

4.7 SecondWait

After LISa has sent the echo requests the first time, it can be possible that some hosts were not found. To improve the results, LISa can ping a second time. This time it will only ping hosts, from which it didn't receive answers. If you have good results with pinging only once, you can disable the second time with setting SecondWait to -1.

Otherwise it might be a good idea to make this value a little bit bigger than the value for FirstWait, since the hosts which were not found on the first try, are probably slower or further away so they might take some milliseconds longer to answer. Usually values from 5 to 50 should be good or -1 to disable the second scan. The maximum is 99 (gives 0.99 seconds, a very long time).

4.8 UpdatePeriod

This is the interval after which LISa will update. After this time LISa will again ping or **nmblookup** or get the list of hosts from the LISa server which actually

does the pinging.

Valid values are between 30 seconds and 1800 seconds (half an hour). If you have a big network, don't make the interval too small (to keep network load low). Values from 300 to 900 seconds (5 to 15 minutes) might be a good idea.

Keep in mind that the update period is doubled if nobody accesses the server, up to 4 times, so the interval will become 16 times the value given here and will be reseted to the value given here if somebody accesses the server.

4.9 `DeliverUnnamedHosts`

If an answer to an echo request from an IP address was received, were LISa could not determine a name, it will be only delivered over the port if you set this to 1.

I am not really sure if this is a useful feature, but maybe there are some infrastructure devices in your network without assigned names, so they don't have to be published. Set this to 0 if you want to keep them secret ;-). If unsure, say 0.

4.10 `MaxPingsAtOnce`

When sending the pings (echo requests), LISa sends a bunch of these at once and then waits for the answers. By default there are 256 pings sent at once, usually you should not need the change this value. If you make it much bigger, the internal receive buffers for the answers to the echo requests may become too small, if you make it too small, the updating will be slower.

4.11 Three more example files

Example 4.1 FIXME

You are member of a small network with 24 bit network mask, i.e. up to 256 hosts:

```
PingAddresses = 192.168.100.0/255.255.255.0
AllowedAddresses = 192.168.100.0/255.255.255.0
BroadcastNetwork = 192.168.100.0/255.255.255.0
SearchUsingNmblookup = 0 #don't use ↔
    nmblookup
FirstWait = 20 #20 ↔
    hundredth seconds
SecondWait = 30 #30 ↔
    hundredth seconds on the seconds try
UpdatePeriod = 300 #update ↔
    after 300 secs
DeliverUnnamedHosts = 0 #don't ↔
    publish hosts without name
```

Example 4.2 Configuration file for hosts running SMB only

You are only interested in hosts running SMB services and you don't have routers in your network:

```
AllowedAddresses = 192.168.100.0/255.255.255.0
BroadcastNetwork = 192.168.100.0/255.255.255.0
SearchUsingNmblookup = 1 #use nmblookup
UpdatePeriod = 300 #update after 300 ↔
    secs
DeliverUnnamedHosts = 0 #don't publish hosts ↔
    without name
```

The LISa Handbook

Example 4.3 Configuration file using both nmblookup and ping

The same network, but here both nmblookup and ping is used.

```
PingAddresses = 192.168.100.0/255.255.255.0
PingNames = bb_mail
AllowedAddresses = 192.168.0.0/255.255.0.0
BroadcastNetwork = 192.168.100.0/255.255.255.0
SearchUsingNmblookup = 1           #also try nmblookup
FirstWait = 30                     #30 hundredth seconds
SecondWait = -1                    #only one try
#SecondWait = 60                   #try twice, and the ←
    second time wait 0.6 seconds
UpdatePeriod = 300                 #update after 300 ←
    secs
DeliverUnnamedHosts = 0            #don't publish hosts ←
    without name
MaxPingsAtOnce = 256               #send up to 256 ICMP ←
    echo requests at once
```

Example 4.4 Configuration file for resLISa

And now a configuration file for resLISa, PingAddresses is not used by resLISa, neither is BroadcastNetwork.

```
PingNames = bb_mail;some_host;some_other_host
AllowedAddresses = 192.168.0.0/255.255.0.0
SearchUsingNmblookup = 1           # use nmblookup
FirstWait = 30                     #30 hundredth seconds
SecondWait = -1                    #only one try
#SecondWait = 60                   #try twice, and the ←
    second time wait 0.6 seconds
UpdatePeriod = 300                 #update after 300 ←
    secs
DeliverUnnamedHosts = 1            #also publish hosts ←
    without name
MaxPingsAtOnce = 256               #send up to 256 ICMP ←
    echo requests at once
```

Chapter 5

Command Line Options and Other Usage

The following command line options are supported:

- v, --version** Prints brief version information.
- h, --help** Gives an overview of the command line options
- u, --unix** Search at first for `$HOME/.lisarc`, then for `/etc/lisarc`. This is the default behavior.
- k, --kde1** Search first for `$HOME/.kde/share/config/lisarc`, then for `$KDE-
DIR/share/config/lisarc`.
- K, --kde2** Looks for the file `lisarc` in every folder returned by running `kde-config --path config`
- c, --config=FILE** Read `FILE` and no other configuration file.
- p, --port PORTNR** Start the server on this portnumber. If you use this, LISa won't be able to cooperate with other LISa's on the network. This option is not available for resLISa

If you send the Hangup-Signal to LISa or resLISa, it will reread its configuration file (`killall -HUP lisa`).

If you send the User1-Signal to LISa or resLISa, it will print some status information to the standard output (`killall -USR1 lisa`). You won't see anything if the console from which LISa/resLISa was started has terminated.

Chapter 6

Credits and Licenses

LISa and resLISa copyright 2000, 2001, Alexander Neundorf

Have fun, Alexander Neundorf neundorf@kde.org

This documentation is licensed under the terms of the [GNU Free Documentation License](#).

This program is licensed under the terms of the [GNU General Public License](#).

Appendix A

Installation

LISa and resLISa need a libstdc++ (it uses only the string-class from it), it *does not* need either Qt™ nor KDE.

In order to compile and install LISa on your system, type the following in the base directory of the LISa distribution:

```
% ./configure
% make
% make install
```

Since LISa uses **autoconf** and **automake** you should have no trouble compiling it. Should you run into problems please report them to the KDE mailing lists.

A.1 Other Requirements

Both resLISa and LISa open a so called 'raw socket' to send and receive ICMP echo requests (pings). To do this, they need `root` privileges.

LISa offers a service on TCP port 7741, and should be installed by `root` and started when the system comes up. It depends greatly on your operating system how to achieve this.

resLISa is intended to be started per user, it doesn't offer anything to the network. It needs to be installed `setuid root`.

If you use the `rlan ioslave` from KDE 2, resLISa can be started automatically.

LISa reads the file `lisarc`, resLISa reads the file `reslisarc`. If you want to be able to configure both from KControl, you have to start them using the command line switch `-K`.

For more information where they look for configuration files read the chapter on chapter 5.