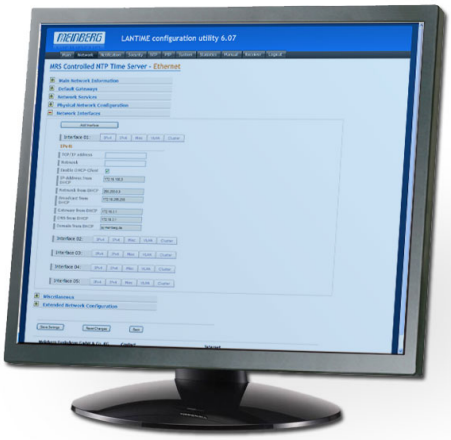

MANUAL

LANTIME OS Version 6

V6.24 Command Line Interface Reference

27th October 2017

Meinberg Radio Clocks GmbH & Co. KG

Table of Contents

1 Introduction

The configuration and management of a Meinberg LANTIME network time server can be performed using a number of different user interfaces. The graphical user interface of such a device is accessible using a web browser and offers the possibility to review the current status of the system and to visualize statistical values using the web based diagram software called xtrastats.

The command line interface (CLI) of the sixth LANTIME firmware generation is using a text-only (non-graphical) approach. It can be accessed using local connections (serial console ports) or remote network connections (SSH or Telnet). The CLI is based on a standard Unix shell interpreter called Bourne Again Shell (Bash), offering comfortable editing of a command line by using the cursor keys and delete/backspace. By accessing a command history using the up/down cursor keys, the shell allows to modify an already used command or simply execute it again, without modification, if required. The tabulator key (Tab) can be used to auto-complete a command and saves the user from having to type in the full command name.

By using a standard shell the LTOS6 firmware environment can benefit from a number of additional advantages. A Unix system administrator will certainly already know how to work with a shell and by making use of the script language elements of the shell, very sophisticated or recurring command sequences can be automated.

In addition to the Bourne Again Shell the so-called Debian Almquist Shell ("Dash") and the standard Almquist Shell ("ash") are available on the system and can be used in addition or as a replacement to the standard shell.

This reference manual does not contain a description for every of the more than 400 commands that are available on a LTOS6 system. It tries to cover the most popular commands, especially those that are LANTIME specific and that are not existing on other Unix- or GNU Linux based systems. A number of commands allows to read a short help text describing the parameters and use of the command by executing "commandname -h".

For any questions regarding the LANTIME command line interface, please contact your Meinberg Technical Support.

2 Accessing and Using the CLI

In order to access the CLI, you need to log in to one of the CLI-supporting user interfaces, by using a serial console port or a network connection with either the SSH or TELNET protocol. If only a web browser is available, it can also be used to access the CLI via the so-called WEBSHELL service.

Serial Console

Serial console ports are located on the front panel or (in modular systems) on the CPU module (some devices come with both a front port and a CPU console port). These ports can be accessed with a serial terminal running at 38400 baud and using 8 data bits, no parity, 1 stop bit (8N1).

Logging in

The default configuration knows one user account (root) with a standard password (timeserver). Other users can access the CLI only if their access level has been set to "Super User". If that is not the case, the system will reject the user and does not allow to access the command line interface.

A "Super User" will be presented with a system status overview after successfully authentication, followed by a shell prompt.

Automatic Logout

The CLI will automatically terminate a CLI session if a user does not enter a command for more than 300 seconds (5 minutes). This timeout can be disabled by entering the "**no_shell_timeout**" command. It can be changed by using the "**set TMOUT=x**" command (x represents the new timeout in seconds).

Entering CLI commands

Commands are case-sensitive, almost all commands are lower-case and do not contain any uppercase characters. It is possible to enter only the beginning of a command and then use the TAB key (CTRL+I) to let the system automatically complete it. If the entered part is not uniquely corresponding to one command, a list of all possible commands that match the entered text is shown.

To edit a command line, the left/right cursor keys (or, alternatively, CTRL+B and CTRL+F) can be used to move the cursor. Entering ESC+F and ESC+B will move the cursor to the beginning of the next or previous word. And BACKSPACE (CTRL+H) deletes the character to the left of the cursor.

Command History

Already entered and processed commands can be recalled within a CLI session by using the cursor up/down keys (or CTRL+P and CTRL+N). It is possible to search for an already entered command line by pressing CTRL+R and then starting to enter a search pattern. If more than one command line matches the entered pattern, repeatedly pressing CTRL+R will toggle through the matching entries.The "**history**" command lists all previously entered and processed commands.

Logging Out, Termination of a CLI session

To log out of the CLI, you can use the "**exit**" or the "**logout**" command. It is also possible to terminate a CLI session by pressing STRG+D at the shell prompt.

3 Command Reference

This chapter descibes all available CLI commands and their parameters.

Conventions

The command names in this chapter are shown in **bold** characters, parameters (if supported) are represented in *italic* characters. If a parameter or part of the commandline is optional, i.e. it does not have to be entered, it is surrounded by brackets [ ].

The character "#" at the beginning of a line represents the shell prompt, which will contain different characters depending on the configuration and status of the device.

Examples:

pwd
(shows the name of the currently selected directory)

ls [**path**]
(shows the content of the specified path *[path]* or - if no *[path]* parameter has been entered, the content of the current directory.

3.1 Configuration Management

The 6th LANTIME firmware generation offers a number of commands to manage the configuration of a device. This includes saving a configuration set under a certain name and, at a later time, restoring/reactivating it. Other commands are aimed at comparing the currently used configuration with the so-called startup configuration that is automatically loaded when the device is (re)starting.

3.1.1 lsconfig - List Saved Configsets

Purpose

This command lists all files included in a saved configuration (i.e. a so-called "configset") for a given package. It can also list all saved configsets for a given package.

Call and Parameters

lsconfig *package [configset]*

If both **package** and **configset** have been specified, the command will show all files included in the specified configset for the given package. If only the **package** is provided, lsconfig will output a list of all available configsets that include saved configuration files for the specified package. If "all" is specified as the package name, the command covers all installed packages.

Examples

```
# lsconfig network myconfig1
```

(shows all files of the network configuration which are included in the saved config set "myconfig1")

```
# lsconfig snmp startup
```

(lists the SNMP related configuration files in the startup configset. This configset is automatically loaded during system startup.)

```
# rmconfig all myconfig2
```

(this lists the files for all pacakges in the configset myconfig2)

3.1.2 rmconfig - Delete a Configset

Purpose

This command deletes a saved configuration (i.e. a so-called "configset"). Attention: rmconfig does not ask for a confirmation, it immediately removes the selected configutation from the flash memory of the device. This is non-reversible and therefore requires you to be very careful when using this command.

Call and Parameters

rmconfig *package configset*

The parameter **package** specifies the package whose configuration should be deleted. This could be "network" or "lantime" or "snmp". The available packages can be found by looking at the contents of the /package directory, in which you can find a subdirectory for each package installed. If the given package name is "all", the whole configuration set will be deleted.

The *configset* parameter defines the saved configuration ("configset") from which the package configuration shall be deleted. It is not allowed to use "default" because the default configuration of a package cannot be deleted. By specifying the configset "startup", the default package configuration will be restored during the next system start.

Examples

```
# rmconfig network myconfig1
```
(removes the network configuration from the saved config set "myconfig1")

```
# rmconfig snmp startup
```
(removes the startup configuration of the snmp package, the SNMP default configuration will be restored during the next boot process)

```
# rmconfig all myconfig2
```
(removes the entire configset "myconfig2", i.e. the configuration of all packages)

3.1.3 diffconfig - Show unsaved Configuration Changes

Purpose

With this command a saved configuration (i.e. a so-called "configset") can be compared with the current configuration. This allows to check for unsaved configuration changes.

Call and Parameters

diffconfig *package configset*

The parameter **package** specifies the package whose configuration should be compared. Examples would be "network" or "lantime" or "snmp". If the given package name is "all", the complete configuration set will be compared. This is the default behavior if no package name is specified on the command line.

The *configset* parameter defines with which saved configuration ("configset") the currently running configuration should be compared.If this is not specified on the command line, the configset "startup" is used as a default.

Examples

```
# diffconfig network myconfig1
```
(compares the current network configuration with the saved config set "myconfig1")

```
# diffconfig
```
(shows the differences between the current configuration to the "startup" configset, i.e. for all packages)

3.1.4 checkconfig - Check for unsaved Configuration Changes

Purpose

This command shows whether an unsaved configuration change has been detected or not. It does not show any details on configuration changes (like diffconfig).

Call and Parameters

checkconfig

This command must be run without any parameters.

Examples

```
# checkconfig
```
No configuration changes.

3.1.5 saveconfig - Save Configuration Changes

Purpose

This command saves the currently active configuration in a configuration set ("configset"). It can be used to persistently store configuration changes (by saving them to the "startup" configset that is loaded during the powerup/boot sequence) and to backup a configuration.

Call and Parameters

saveconfig *package configset*

package specifies the package whose configuration should be saved. Examples for this are "snmp" or "ssh" or "network". If no package name is provided on the command line, the configuration for all packages is saved as a standard behavior.

The *configset* parameter defines the saved configuration ("configset") to which the package configuration(s) shall be saved. It is not allowed to use "default" because the default configuration of a package cannot be overwritten/changed. By specifying the configset "startup", the configuration will be restored during the next system start. If no configset name is specified, the startup configuration set ("startup") is used as a default.

Examples

```
# saveconfig snmp
```
(saves the SNMP configuration to the startup configuration "startup")

```
# saveconfig network backup1
```
(saves the network configuration to a configset "backup1")

```
# saveconfig all myconfig2
```
(saves the entire configuration to the configset "myconfig2", i.e. the configuration of all packages)

3.1.6 loadconfig - Load Configset

Purpose

The loadconfig command loads the configuration for the whole system (all packages) or a given package from a previously saved configuration set ("configset"). It can be used to restore the default configuration, the startup configuration or a configuration backup.

Call and Parameters

loadconfig *package configset*

The **package** parameter specifies the package whose configuration should be loaded. Using "all" will load the configuration for all packages. If no package name is provided on the command line, "all" is assumed as the standard behavior.

With the *configset* parameter the name of the previously saved configuration ("configset") is defined. The package configuration(s) is loaded from this configuration set. Specifying "default" will load the default values of the package and "startup" loads the startup configuration set that is automatically loaded during the powerup/boot process. If no configset is given, the "startup" configset is loaded.

Examples

```
# loadconfig snmp
```
(loads the SNMP configuration from the startup configuration "startup")

```
# loadconfig network backup1
```
(loads the network configuration from the "backup1" configset)

```
# loadconfig all myconfig2
```
(loads the entire configset "myconfig2", i.e. the configuration of all packages)

3.2 File Management

The management of files in the flash memory of a LANTIME device is normally handled automatically by the LANTIME firmware itself. However, in certain situations it might be required that an administrator has to manually delete, copy or rename a file. From time to time the contents of a file have to be checked, for example when looking at a log file or a status file.

The following CLI commands enable you to perform these tasks.

3.2.1 pwd - Print Working Directory

Purpose

The pwd command prints the name and path of the current working directory.

Call and Parameters

pwd

This command does not require any parameters.

Examples

```
# pwd snmp
/var/run
```

3.2.2 cd - Change Working Directory

Purpose

The cd command changes the current working directory.

Call and Parameters

cd *[directory]*

The system changes the working directory to the given directory or, if no directory has been specified, to the home directory of the current user.

Examples

```
# cd /etc
```
(sets the working directory to /etc)

```
# cd
```
(changes to the home directory of the current user, e.g. /root for the root user)

3.2.3 ls - List Directory Contents

Purpose

With this command, the contents of a given directory can be listed.

Call and Parameters

ls *[Options] [directory]*

The content of the given directory are printed. A large number of options is available which control how the ls command lists all the files and subdirectories. Please use the "–help" option to get a list of all supported options.

Examples

```
# ls /var/log
```
(shows the content of the /var/log directory in standard output format)

```
# ls -l /var/run
```
(lists the files and subdirectories of the /var/run directory, using the "long" output format (-l) which shows a number of details like file sizes)

3.2.4 cp - Copy Files and/or Directories

Purpose

The "cp" command copies files or whole directories.

Call and Parameters

cp *[Options] [Source(s [Target]*

An overview with all supported options can be requested with

```
cp -help
```

The option "-v" ("verbose") for example shows the name and path of the file that is currently worked on during the copy operation.

One or more files can be specified as the source(s), wildcards (like * or ?) are allowed. The target can either be a directory or, if the source is one single file, a target filename.

Copying a whole directory structure is possible by using the "-r" (recursive) option.

Examples

```
# cp /etc/hosts /var/tmp
```

(copies the file hosts from the /etc directory into the target directory /var/tmp where it will be stored under the same name, i.e. hosts)

```
# cp /config/global_configuration /var/tmp/mycopy
```

(copies the file global_configuration from the /config directory into the target directory /var/tmp using the target filename mycopy)

```
# cp /etc/ssh/ssh_* /tmp/
```

(copies all files form /etc/ssh with a filename beginning with "ssh_" into the directory /tmp)

```
# cp -r /etc/udev /tmp/
```

(creates a copy of the /etc/udev directory with all subdirectories and containing files in the target directory /tmp)

3.2.5 mv - Move Files and/or Directories or Rename them

Purpose

The "mv" command moves files or whole directories from one location to another. It can be used to rename files and directories, too.

Call and Parameters

mv *[Options] [Source(s [Target]*

An overview with all supported options can be requested with

```
mv -help
```

One or more files can be specified as the source(s), wildcards (like * or ?) are allowed. The target can either be a directory or, if the source is one single file, a target filename. In this case, the original file will be moved and renamed at the same time.

Moving a whole directory structure is possible by specifying a directory as the source.

Examples

```
# mv /dir_a/file_a /dir_b/file_b
```
(moves the file file_a from the /dir_a directory into the target directory /dir_b and renames it to file_b)

```
# mv /dir_a/file_a /dir_b/
```
(moves the file file_a from the /dir_a directory into the target directory /dir_b but preserves the filename)

```
# mv /dir_a/file_*.txt /tmp/
```
(moves all files from /dir_a with a filename beginning with "file_" and ending on ".txt" into the directory /tmp)

```
# mv /dir_a/ /tmp/
```
(moves the whole directory /dir_a with all its subdirectories and included files into the /tmp directory)

3.2.6 rm - Delete Files and/or Directories

Purpose

The "rm" command deletes one or more files or whole directories (including all their content, i.e. files and subdirectories). It is possible to use wildcard characters to delete a group of similar named files, e.g. "*.bak" includes all filenames that end on ".bak". Since deleting files and directories can lead to system malfunction and all kinds of failures, the "rm" command should only be used if you are 100% sure that the specified files/directories are not required for proper operation of the LANTIME system. If you are in doubt, please contact Meinberg support.

Deleted files and directores cannot be restored and are lost forever. Because of this, the "rm" command should be used with the greatest caution.

Call and Parameters

rm *[Options] [File1] [File2] ...*

An overview of all supported options can be requested with

```
rm -help
```

One or more files can be specified, wildcards (like * or ?) are allowed. If a whole directory and all its contents shall be deleted, the "-r" option needs to be specified.

There is no "Are you sure?" prompt shown before the deletion is carried out, the system will immediately delete the specified files. In order to avoid system failures, please triple check whether the file(s) and/or directories you specify are really OK to be deleted.

Examples

```
# rm /dir_a/file_a
```
(deletes the file file_a from the /dir_a directory)

```
# rm -r /dir_b/
```
(deletes the whole /dir_b directory and all included files and subdirectories - forever)

3.2.7 cat - Show File Contents

Purpose

The "cat" command shows the contents of a given file.

Call and Parameters

cat *[filename]*

The content of the given file is printed. The "cat" command can be combined with the "less" command to allow paginated output and offers an easier way to review a file.

Examples

```
# cat /var/log/messages
```
(shows the content of the file messages in the /var/log directory)

```
# cat /var/log/lantime_messages | less
```
(shows the contents of the file /var/log/lantime_messages, the "less" command offers a way to navigate the file using the arrow keys, space (=next page) and offers a search function ("/"). Closing the file can be achieved by pressing the "q" key).

3.3 Firmware Management

LTOS V6 allows the installation of multiple firmware images in parallel. Selecting which of the installed images is going to be loaded at the next system start - and commands that allow to remove or install a firmware release manually without using the web user interface - are described in this section.

3.3.1 fwlist - List Installed Firmware Images

Purpose

The "fwlist" command prints a list of all firmware images which are installed on the device.

Call and Parameters

fwlist *[-v] [searchpattern]*

The [searchpattern] parameter can be used to filter the list of installed firmware images. If no searchpattern is specified, all installed images are listed. The "-v" option will show the version number of each firmware image behind its name.

Examples

```
# fwlist
```
(shows all installed firmware images)

```
# fwlist -v fw_*
```
(shows all installed firmware images with a name beginning with "fw_" and their respective firmware revision)

```
# fwlist OSV
```
(shows the installed firmware image with the name "OSV")

3.3.2 fwselect - Select/Show Activated Firmware Image

Purpose

With the "fwselect" command it is possible to activate an installed firmware image, i.e. that firmware image is started during the next boot sequence. If an error occurs during the activation, the system will roll back to the previous state.

If fwselect is started without any parameters, it will show the name of the activated firmware image, i.e. the image that is going to be used at the next system start.

Call and Parameters

fwselect *[FWImage]*

The [FWImage] parameter specifies which firmware image is going to be used at the next system start. Without this parameter, "fwselect" will print the name of the currently selected image and exits.

Examples

```
# fwselect
```
(shows the currently activated firmware image)

```
# fwselect fw_6.12.004
```
(selects the image "fw_6.12.004" and tries to prepare the system to use this image at the next boot sequence)

3.3.3 fwrm - Delete Firmware Images

Purpose

The "fwrm" command can be used to delete one or more firmware images from the internal flash memory to regain space.

Call and Parameters

fwrm *[FWImage]*

or

fwrm *[–wipe-all [keep=X]] [FWImage]*

The *FWImage* parameter defines which image will be deleted. The secon form (–wipe-all) deletes *all* firmware images except the OSV image, the currently running image and - if different from the running image - the firmware image that has been selected to be activated at the next system start. The optional "keep" parameter allows to specify how many firmware images should be preserved in addition to the non-deletable images mentioned above.

The –wipe-all option can be shortened by using -W instead.

Examples

```
# fwrm fw_6.14.021
```
(deletes the firmware image fw_6.14.021)

```
# fwrm -wipe-all keep=2
```
(deletes all firmware images except the currently active image, the OSV image and the firmware image that has been selected (by fwselect) to be activated at the next system start)

3.3.4 fwuncompress - Extract Firmware Image

Purpose

Starting with version 6.15 all firmware updates will be installed in compressed form to preserve flash space. Such an image is read-only and cannot be modified, i.e. it is not possible to add or remove files or change their content in any way. Under normal circumstances this is not required and therefore it is recommended to use compressed images instead of uncompressed ("standard") ones. If it is necessary for a specific user requirement to change the contents of a firmware image, the "fwuncompress" command can extract the contents of a compressed image and create a new, uncompressed copy of it. The (compressed) source image will not be touched or changed in any way by "fwuncompress" and, if not required anymore, would have to be deleted manually afterwards using the "fwrm" command.

It is possible to uncompress the currently running firmware image without any problems.

Call and Parameters

fwuncompress *FWImage*

The specified image (name usually starts with "fw_") will be used to create an uncompressed copy of it. The newly created image will get a prefix "u", i.e. uncompressing a firmware image "fw_6.15.015" will create an uncompressed image named "ufw_6.15.015".

Examples

```
# fwuncompress fw_6.16.002
```

(extract the contents of the compressed firmware image fw_6.616.002, creating a new image named ufw_6.16.002)

3.4 User Account Management

The system supports multiple local user accounts and remote authentication methods using external RADIUS and TACACS+ servers. Managing the accounts and checking the current status is possible with several commands.

3.5 Network Configuration

A number of CLI commands enable you to change the LANTIME network parameters in addition to the front panel menu (if available) or, in case the initial network configuration has already been set up, by using the web interface. This can be very useful when network connectivity is lost or in case a special network setup is required that is not supported by the web UI.

The main configuration file for network related settings is */etc/mbg/net.cfg* which contains definitions and parameters for all physical and logical ("virtual") network interfaces. This file, its structure and content, is described in detail in the Configuration Files chapter.

3.5.1 netconfig - Check for Network Configuration Changes and Apply them

Purpose

With **netconfig** the system will compare the state of all network interfaces (both physical and virtual) with their configuration. If any required changes are detected, they will be applied.

If, for example, a virtual interface has been configured but is missing, it will be created and configured according to the net.cfg contents.

Call and Parameters

netconfig

This command does not support any parameters.

Examples

```
# netconfig
```

(checks all network interfaces and applies changes, if the configuration differs from the current state)

3.5.2 nicinfo - Show Physical Network Interface Configuration State

Purpose

nicinfo displays the configuration status of physical network interfaces. It lists the MAC address, the assigned bonding group, the link speed and duplex mode as well as the IPv6 mode.

Call and Parameters

nicinfo *[Optionen] [INTERFACE]*

This command understands the following optins:
-c Check Link Mode
Shows only the current link state (e.g. 100FDX) and whether the network port is monitored or not (LINK_CHECK).

-s Short Mode
This option leads to a very compact output, only indicating the current configuration state of an interface:

```
+ Not existing, needs to be created
!  Changed, requires reconfiguration
- Existing but not configured, needs to be removed
= Configuration is correct, no changes required
```

If an interface name is specified with the INTERFACE parameter, "**nicinfo**" will only show information about the specified interface (e.g. lan0). If this parameter is not specified or empty, the command will output information about all interfaces.

Examples

```
# nicinfo

Please wait ...
Current state of physical interfaces:
lan0 matches configuration (lan0 00:13:95:00:6b:ef - 100FDX AUTO=ON
IPV6=ACTIVATED+AUTOCONF)
lan1 matches configuration (lan1 00:60:6e:7a:d3:4d - 10HDX AUTO=ON
IPV6=ACTIVATED)
lan2 matches configuration (lan2 00:60:6e:7a:d3:4e - 10HDX AUTO=ON
IPV6=DEACTIVATED)
lan3 matches configuration (lan3 00:60:6e:7a:d3:4f - 10HDX AUTO=ON
IPV6=DEACTIVATED)
```

(shows the status of all physical interfaces)

```
# nicinfo -s
=lan0
!lan1/1
=lan2
=lan3
```

(shows configuration state for all interfaces, in this case there is a pending change for lan1)

```
# nicinfo -c lan0
Please wait ...
Current state of lan0:
Status of physical interface lan0 is 100FDX LINK_CHECK=ON
```

(shows the link state of lan0 - 100Mbit/s Full Duplex - and whether it is monitored or not)

3.5.3 nicmgr - Management of Physical Network Interfaces

Purpose

The "nicmgr" command allows to add unconfigured physical network interfaces to the network configuration in order to be able to assign virtual network interfaces to them. This is necessary whenever new network interface cards are added to an existing system, for example by inserting a new LNE module into a device. It is also possible to remove network interfaces from the configuration with nicmgr, if those physical network interfaces have been permanently removed from the system.

Call and Parameters

nicmgr help

or

nicmgr assign [FREE_IF] [IFNUMBER]

or

nicmgr remove [IFNUMBER]

or

nicmgr autoassign

or

nicmgr autoreplace

The FREE_IF parameter represents an unconfigured/uninitialized network interface. These interfaces are named ethX (X is a running number which is assigned at startup or directly after a network expansion module has been inserted into the system). When a LANTIME Network Expansion (LNE) card is added to the system, the four new interfaces will be named "eth0, eth1, eth2 and eth3. As soon as they have been correctly added to the configuration, they will be renamed lanX (where X is also a number that has been assigned by the user or the system). The first physical network interface is always located on the management CPU module and is named "lan0".

IFNUMBER is the number of an already added (configured) port, therefore IFNUMBER=1 refers to the physical interface lan1, a "5" means lan5 and so on.

The two commands "autoassign" and "autoreplace" simplify the addition or the replacement of multiple ports. "autoassign" automatically adds all detected and currently unconfigured network interfaces to the configuration. The "autoreplace" command searches for configured but missing interfaces (e.g. if a LNE card has been removed due to a failure, its interfaces are still in the configuration but they are missing). If it finds missing interfaces and unconfigured interfaces, it will replace the configuration of the first missing interface with the first unconfigured interface, the second missing interface with the second unconfigured interface and so on.

Examples

```
# nicmgr assign eth0 7
```
(adds the currently unconfigured interface eth0 as lan7 to the system)

```
# nicmgr remove 6
```
(removes lan6 from the system configuration)

```
# nicmgr autoassign
```
(automatically adds all unconfigured/unassigned interfaces to the system configuration)

```
# nicmgr autoreplace
```

(replaces all missing physical network ports with available unconfigured ethX interfaces)

3.5.4 netinfo - Show Logical Network Interface Configuration State

Purpose

netinfo displays the configuration status of logical ("virtual") network interfaces, it shows IP addresses and the configuration state of the interface(s), i.e. if an interface state corresponds to the configured state.

Call and Parameters

netinfo *[Optionen] [INTERFACE]*

This command understands the following optins:
-a Advanced Info Mode
Shows more detailed information for each interface, e.g, the MAC address of the assigned physical interface and the administrative state.

-s Short Mode
This option leads to a very compact output, only indicating the current configuration state of an interface:

```
+ Not existing, needs to be created
!  Changed, requires reconfiguration
- Existing but not configured, needs to be removed
= Configuration is correct, no changes required
_ No Configuration, empty configuration for this interface
```

If an interface name is specified with the INTERFACE parameter, "**nicinfo**" will only show information about the specified interface (e.g. lan0:0). If this parameter is not specified or empty, the command will output information about all interfaces.

Examples

```
# netinfo
Current state of logical interfaces:
bond0:2 matches configuration (STATIC 10.99.109.11 255.255.255.0 - NONE)
lan0:0 matches configuration (DHCP - - - NONE)
lan1:1 [Virtual Interface 1] is not active and requires to be configured
bond0:3 has no configuration
```

(shows the status of all logical interfaces)

```
# netinfo -s
=bond0:2
=lan0:0
+lan1:1
_bond0:3
```

(shows configuration state for all logical interfaces)

```
# netcinfo -s -i lan0:0
=lan0:0/0
```

(like above, but now contains the interface number, too: /0)

3.6 System Services

The installed system services fulfill a number of duties, most of them provide a certain network protocol service like SSH (Secure SHell) oder HTTP (Hyper Text Transport Protocol, the web GUI). Starting and stopping these services is normally managed automatically depending on the configuration of the system. If the TELNET service has been disabled on all interfaces, it will automatically be stopped by the system. When the user re-enables it on at least one interface, the system will restart the corresponding TELNET service.

In certain situations, it can be necessary to check the status of a service or manually start or stop it. After a manual configuration file change it is often required to restart a related service to force it to apply the changed configuration.

With the command "status all", the running state of all registered services will be listed and therefore can be used to find out which services are available on a certain device.

This chapter describes the various CLI commands that control the system services.

3.6.1 status - Show Status of System Services

Purpose

The status command shows whether a specified system service is currently running or not. It is also possible to get the status for all services.

Call and Parameters

status *service*

The only parameter is the name of the service for which the running state should be shown. Specifying "all" instead of a certain service name will result in showing the state of all services.

Examples

```
# status ssh
```
(shows whether the SSH service is currently running or not)

```
# status all
```
(lists the state of all system services)

3.6.2 start - Start a System Service

Purpose

With this command, a specified system service can be started if it is not already running.

Call and Parameters

start *service*

The *service* parameter specifies which service should be started. The system first checks whether the service is already running or not. If it is, nothing will happen. You can check the running state of a service with the **status** command.

Examples

```
# start ssh
```
(starts the SSH service if it is not already running)

```
# start http
```
(starts the HTTP service)

3.6.3 stop - Stop a System Service

Purpose

The **stop** command will stop a specified system service, i.e. the relevant processes are terminated.

Call and Parameters

stop *service*

With the *service* parameter you can specify which service should be stopped. The system first checks whether the service is currently running or not. It will only try to stop the service if it is running, otherwise nothing will happen. The system stops a network related service automatically if it has been disabled on all interfaces. Stopping such a service will immediately result in terminating any active connections and disables connectivity on all interfaces.

Examples

```
# stop ssh
```
(stops the SSH service if it is running - ATTENTION: this will immediately terminate any active SSH connection, including the one that you used to enter this command)

```
# stop https
```
(stops the HTTPS service)

3.6.4 restart - Restart a System Service

Purpose

The **restart** command stops a service (if it is running) and then restarts it. If it was not running when the restart command has been called, it is started normally (omitting the "stop" command).

Call and Parameters

restart *service*

service specifies which service should be restarted. The system first checks whether the service is already running or not. If it is, it will stop the service and then restart it. A non-running service will simply be started.

Examples

```
# restart ssh
```
(restarts the SSH service, active connections are terminated)

```
# restart http
```
(restarts the HTTP service)

3.6.5 reload - Reload Configuration of a System Service

Purpose

The **reload** command forces a service to reload its configuration, for most services this is achieved by restarting them. See "**restart**" command, but "**reload**" automatically chooses the applicable way for each service.

Call and Parameters

reload *service*

The *service* parameter specifies for which service the configuration should be reloaded.

Examples

```
# reload ssh
```
(reloads the SSH service configuration by restarting it)

```
# reload http
```
(reloads the HTTP service configuration by restarting it)

3.6.6 svcconfig - Check for System Service Configuration Changes and Apply them

Purpose

The **svcconfig** command will check for all services if one of the registered configuration files changed (since the last start of the service). If such a change is detected, the corresponding service is forced to reload its configuration (with the **reload** command). A list of all registered configuration changes can be found in the /var/run/services/svccfg.db file. This file can be inspected by using the **cat** command.

Call and Parameters

svcconfig

This command does not support any parameters. It will always check all registered files for all services.

Examples

```
# svcconfig
```

(checks all registered configuration files for all services and, if a file change has been detected, reloads the corresponding service)

3.7 Showing Current Status Information - show and monitor

LTOS offers a number of status information displays to allow the user to view the current status of the system. A whole range of detailed information options is available by using the standard system commands, but often these commands produce a very detailed output that contains a lot of unimportant or unnecessary information and uses a hard to read format for presenting the information. In order to overcome this limitation, LTOS V6 includes a number of commands that generate an optimized output in an easy to understand format, concentrating on the most important status information variables.

This information can be requested by using one of the two commands "show" and "monitor". While "show" will display the current status and then returns to the command prompt, "monitor" will keep running and updates its output in a fixed time interval. In order to return to the command prompt, "monitor" has to be stopped by pressing CTRL+C.

After the "show" or "monitor" command word it is necessary to specify which type of information should be displayed. The different types of information are provided by so-called plugins, each of them generating a specific type of status information. The "ip" plugin for example can generate and output a list of all IP addresses currently used by the system. In order to get this information, the user either has to enter the command "show ip" (will generate and show a list of all IP addresses and then returns to the command prompt) or "monitor ip" (the IP address list is shown and will for example be updated every 10 seconds until CTRL+C is pressed to stop the monitor command).

The following sections will explain the available plugins and, if necessary, their additional parameters.

3.7.1 cpuload - Show CPU Utilization Metrics

Purpose

The "cpuload" module shows the current CPU utilization metrics of the system.

Call and Parameters

 # show cpuload

or

 # monitor cpuload

This command does not have any additional parameters. The following output line will be generated once (using the "show" command) or every 5 seconds (using the "monitor" command):

```
Tue Jan 21 12:52:26 UTC 2014 Cpu(s):  3.0%us, 4.8%sy, 0.0%ni, 92.1%id,
0.0%wa, 0.0%hi, 0.1%si, 0.0%st Loadavg:  0.36 0.21 0.19 1/80 12803
```

"Tue Jan 21 12:52:26 UTC 2014" is the current date/time, "Cpu(s): 3.0%us, 4.8%sy, 0.0%ni, 92.1%id, 0.0%wa, 0.0%hi, 0.1%si, 0.0%st" indicates the CPU utilization for each CPU state (us=User, sy=System, ni=nice, wa=I/O wait, hi=Hardware IRQ, si=Software IRQ, st=Steal Time), "Loadavg: 0.36 0.21 0.19" represents the load average values for the last 1, 5 and 10 minutes and "1/80 12803" shows the number of currently running/total processes and the last assigned process ID.

Examples

```
# show cpuload
```
(shows the current CPU utilization)

```
# monitor cpuload
```
(continously shows the CPU utilization every 5 seconds)

```
# s c
```
(shortcut for "show cpuload")

```
# m c
```
(shortcut for "monitor cpuload")

3.7.2 devices - Meinberg Hardware Module and Components List

Purpose

The "devices" module shows the system details and a list of detected Meinberg hardware components.

Call and Parameters

 # show devices

or

 # monitor devices

This command does not have any additional parameters. The following output line will be generated once (using the "show" command) or every 10 seconds (using the "monitor" command):

```
System Details
System ID: M200
Backplane:  P
CPU Carrier:  V33
Platform:  AMDCONGA
CPU Board:  E900
CPU ID: CPU=AuthenticAMD CPUID=AuthenticAMD MODELID=...
RAM: 100868 kB

Found 1 reference clock[s]
GPS170 :2.29 S/N: 11123120 BinaryPort:2 TimeStrPort:0

System Components
Bus/Id Device Product Ver Serial Status
USB 001/005:  1938:0101 Meinberg CPC - Control Panel Controller 1.12 1.0.0
0x0001
```

Examples

```
# show devices
```
(shows the system details and the list of detected Meinberg components)

```
# monitor devices
```
(continously repeats the "show devices" command until CTRL+C has been pressed)

```
# s d
```
(shortcut for "show devices")

```
# m c
```
(shortcut for "monitor devices")

3.7.3 ip - List all active IPv4 and IPv6 Adresses

Purpose

The "ip" plugin generates a list of all currently active IP addresses.

Call and Parameters

show ip *[Filter]*

or

monitor ip [Filter]

The "[Filter]" parameter is optional and allows to filter the output of "show ip" or "monitor ip" using a search keyword.

In the "monitor" mode, the list of IP addresses is automatically refreshed every 10 seconds, until CTRL+C has been pressed.

The output of "show ip" / "monitor ip" looks like this (example):

```
Currently Active Network Interfaces:

lan0:0 linklocal ipv6 fe80::213:95ff:fe0a:580b/64
lan0.120 static ipv4 172.16.25.200/255.255.000.000
lan0:0 static ipv6 bad:babe:25::200/64
lan1:1 dhcp ipv4 192.168.10.12/255.255.255.000
```

The first column represents the interface name, which is composed from the name of the physical port (e.g. "lan0" for the first Ethernet port or "bond2" for an interface that is part of the bonding group 3) and the ID of the logical ("virtual") network interface. This can be either the interface number (":0" for the first virtual interface) or the VLAN ID (".120" for VLAN ID 120) .

The second column lists the type of the address, this can be "static" for manually configured static IP addresses, "dhcp" for IP addresses automatically assigned by DHCP/DHCPv6, "linklocal" for IPv6 Linklocal addresses or "ra" for IPv6 addresses assigned by a router advertiser.

Whether an IP address entry is an IPv4 or IPv6 address is specified in the third column.

The fourth and last column finally shows the IP address and either the netmask (for IPv4) or the prefix length (for IPv6).

Examples

```
# show ip
```
(shows the current list of all active IP addresses)

```
# monitor ip
```
(like "show ip", but automatically refreshes every 10s)

```
# show ip lan0
```
(shows all IP addresses assigned to the physical port "lan0")

```
# show ip ipv6
```
(shows only the IPv6 addresses)

 # show ip dhcp
(lists all IP addresses assigned by DHCP or DHCPv6)

 # monitor ip bond
(lists all IP addresses assigned to one of the bonding interfaces and refreshes automatically every 10s)

3.7.4 lantimelog - Show LANTIME Log File

Purpose

This "show" plugin lists the entries of the LANTIME log file (/var/log/lantime_messages), a file that only contains the most important events.

Call and Parameters

 # show lantimelog [**Filter**]

or

 # monitor lantimelog [**Filter**]

In the "monitor" mode, only the most recent protocol entries are shown, afterwards the command will wait for new entries and prints them as soon as they are created. The CTRL+C key combination aborts waiting for new events and returns to the command prompt.

If a Filter parameter has been added, only those lines in the log file will be printed that contain the given filter string (this is not case sensitive). If no filter is specified, all entries will be listed.

The output of this command looks like this:

```
# show lantimelog
2014-11-20 13:26:55 UTC: LANTIME -> OSCILLATOR ADJUSTED [Refclock:  1 ]
2014-11-20 13:26:07 UTC: LANTIME -> NORMAL OPERATION
2014-11-20 13:26:03 UTC: LANTIME -> NETWORK LINK UP [Affected LAN
Interface:  1 ]
2014-11-20 13:25:57 UTC: LANTIME -> NTP RESTART
2014-11-20 13:25:57 UTC: LANTIME -> NTP SYNC TO GPS
#
```

Examples

 # show lantimelog
(shows the full LANTIME protocol file)

 # show lantimelog ntp
(shows all protocol entries in the log file that contain the string "NTP")

 # monitor lantimelog
(lists the last 20 entries and then waits for new entries, cancel waiting with CTRL+C)

 # s la
(short form of "show lantimelog")

 # m la
(short form of "monitor lantimelog")

3.7.5 linkstate - Show Connection Status of Physical Network Interfaces

Purpose

The "linkstate" plugin shows the current network connection state of one or more physical network interfaces.

Call and Parameters

show linkstate ***[Filter]***

or

monitor linkstate ***[Filter]***

The "[Filter]" parameter is optional and can be specified to limit the output to only those network interfaces containing the filter string either in their name or MAC address.

The "monitor" mode automatically refreshes the output every 10 seconds until CTRL+C has been pressed.

The output of "show linkstate" or "monitor linkstate" looks like this:

```
Current LINK state:...
lan0:  [00:13:95:12:65:36] 100FDX
lan1:  [ec:46:70:ef:3f:e8] NO_LINK
lan2:  [ec:46:70:ef:3f:e9] 1000FDX
lan3:  [ec:46:70:ef:3f:ea] NO_LINK
```

The first column contains the name of the interface, e.g. "lan0" for the first physical port.

The second column represents the MAC address of the interface and the third column indicates the current connection state. This can be either "NO_LINK", if no active connection could be established, or it shows the current connection speed (10, 100, 1000 or 10000) in MBit/s plus the duplex mode (FDX for full duplex or HDX for half duplex).

Examples

```
# show linkstate
```
(shows the connection state of all physical network interfaces)

```
# monitor linkstate
```
(like "show linkstate", but refreshing the output every 10s until CTRL+C has been pressed)

```
# show linkstate lan0
```
(shows only the state of lan0)

```
# s li
```
(short form of "show linkstate")

```
# m li
```
(short form of "monitor linkstate")

3.7.6 modules - Kernel Driver and Module List

Purpose

This "modules" plugin shows a list of all loaded kernel modules and drivers. If you want to show all detected hardware modules and components in your system, please check out the "show devices" command.

Call and Parameters

 # show modules

or

 # monitor modules

This command does not have any additional parameters. The following output line will be generated once (using the "show" command) or every 10 seconds (using the "monitor" command):

```
Loaded kernel modules:
Module Size Used by
ip6table_filter 708 0
ip6_tables 8129 1 ip6table_filter
usb_storage 31278 0
rndis_host 3875 0
cdc_subset 1165 0
cdc_ether 2996 1 rndis_host
bonding 63801 0
ax88179_178a 10848 0
dmfe 13263 0
xt_state 780 0
8021q 11207 0
ipv6 174754 20
squashfs 16978 1
pata_cs5536 2138 1
ext3 85915 0
mbcache 3228 1 ext3
jbd 28294 1 ext3
libahci 14214 0
cgosdrv 18409 0
mdio_bitbang 1515 0
libphy 13845 1 mdio_bitbang
usbnet 10270 4 rndis_host,cdc_subset,cdc_ether,ax88179_178a
ftdi_sio 25482 2
```

Examples

 # show modules
(shows all currently loaded kernel modules)

 # monitor modules
(continously repeats the "show modules" command until CTRL+C has been pressed)

 # s m
(shortcut for "show modules")

 # m m
(shortcut for "monitor modules")

3.7.7 network - Show current Network State

Purpose

This command shows an overview of the currently active network configuration, including the assigned IP addresses, the link state of the physical network interfaces and, if appropriate, the state of bonding groups.

Call and Parameters

 # show network

This command does not have any additional parameters. The following output line will be generated:

```
=== Physical Interface lan0 :  00:13:95:12:65:36
Speed:  100Mb/s
Duplex:  Full
Link detected:  yes

Assigned Virtual Interfaces:  [:0]
Active Virtual Interfaces:  -------------
lan0:0 static ipv4 172.16.25.204/255.255.000.000

=== Physical Interface lan1 :  ec:46:70:00:3f:e8
Speed:  10Mb/s
Duplex:  Half
Link detected:  no

Assigned Virtual Interfaces:  [:1]
```

Examples

```
# show network
```
(shows the currently active network configuration)

```
# s n
```
(shortcut for "show network")

3.7.8 Processes - List System Processes

Purpose

The "show processes" command generates a list of all processes currently running on the system. In combination with a filter string it is possible to check if a certain command or software component has been started and is still running.

Call and Parameters

show processes ***[Filter]***

or

monitor processes ***[Filter]***

The "[Filter]" parameter is optional and allows to filter the output of "show processes" or "monitor processes" using a search keyword (case insensitive). The filter string can contain either a part of a command name or a process ID.

In the "monitor" mode, the list of processes is automatically refreshed every second until CTRL+C is pressed.

The output of "show processes" / "monitor processes" looks like this (example):

```
PID TTY STAT TIME COMMAND
1 ?  Ss 1:23 /sbin/init
2 ?  S 0:00 [kthread
3 ?  S 16:12 [ksoftirqd/0]
5 ?  S< 0:00 [kworker/0:0H]
7 ?  S< 0:00 [kworker/u:0H]
8 ?  S< 0:00 [khelper]
9 ?  S 0:00 [kworker/u:1]
146 ?  S 0:00 [bdi-default]
147 ?  S< 0:00 [kblock
155 ?  S< 0:00 [ata_sff]
162 ?  S 0:00 [khub
268 ?  S< 0:00 [rpcio
279 ?  S 0:00 [kswapd0]
280 ?  S 0:00 [fsnotify_mark]
281 ?  S< 0:00 [nfsio
282 ?  S< 0:00 [crypto]
552 ?  S< 0:00 [deferwq]
554 tty4 Ss+ 0:00 /sbin/getty 38400 tty4
556 tty2 Ss+ 0:00 /sbin/getty 115200 tty2
557 tty3 Ss+ 0:00 /sbin/getty 38400 tty3
600 ?  S 0:00 cat /proc/kmsg
615 ?  S 1694 ?  S 0:00 [scsi_eh_0]
1699 ?  S 0:00 [scsi_eh_1]
1704 ?  S 1:22 [kworker/u:2]
1777 ?  S< 0:00 [kworker/0:1H]
1941 ?  S< 0:00 [loop0]
4861 ?  S 0:01 [kworker/0:2]
6056 ?  S< 0:00 [bond0]
6091 ?  S< 0:00 [bond1]
6126 ?  S< 0:00 [bond2]
6161 ?  S< 0:00 [bond3]
6196 ?  S< 0:00 [bond4]
7885 ?  Ss 2:55 crond
7903 ?  Ss 0:00 /sbin/dbus-daemon -config-file=/etc/dbus-1/system.conf
8998 ?  S 0:02 [kworker/0:1]
9153 ?  S 1:53 ifplugd -M -f -a -b -d 1 -p -q -i lan0
9179 ?  S 1:51 ifplugd -M -f -a -b -d 1 -p -q -i lan1
9205 ?  S 1:51 ifplugd -M -f -a -b -d 1 -p -q -i lan2
...
```

The first column represents the process ID and the second column contains the terminal name (TTY), which can be "?" for internal processes not bound to a specific terminal.

The third column shows the current process state:

```
D Uninterruptible sleep (usually IO)
R Running or runnable (on run queue)
S Interruptible sleep (waiting for an event to complete)
T Stopped, either by a job control signal or because it is being traced.
X dead (should never be seen)
Z Defunct ("zombie") process, terminated but not reaped by its parent.
```

In the fourth column the cumulative CPU time used by this process and - after that - the command itself, typically with its parameters, is listed.

Examples

```
# show processes
```

(shows the current list of all processes currently running on the system)

```
# monitor processes
```

(like "show ip", but automatically refreshes every second)

```
# show processes ntp
```

(shows all processes which contain the search term "ntp" in their command line, this is not case sensitive)

```
# s p
```

(short form of "show processes")

```
# m p ntp
```

(short form of "monitor processes ntp")

3.7.9 route - List all active IPv4 and IPv6 Network Routes

Purpose

The "route" plugin show all currently active IP routes and routing rules.

Call and Parameters

 # show route ***[Filter]***
or
 # monitor route [Filter]

The "[Filter]" parameter is optional and allows to filter the output of "show route" or "monitor route" using a search keyword. This can be used to limit the output to only those entries that contain the specified search term.

In the "monitor" mode, the routing entry list is automatically refreshed every 10 seconds, until CTRL+C has been pressed.

The output of "show route" / "monitor route" looks like this (example):

```
Routing Table Entries:
TABLE DEV TARGET
main lan0 default
main lan0 172.16.0.0/16 proto kernel scope link src 172.16.25.204
local lo broadcast 127.0.0.0 proto kernel scope link src 127.0.0.1
local lo local 127.0.0.0/8 proto kernel scope host src 127.0.0.1
local lo local 127.0.0.1 proto kernel scope host src 127.0.0.1
local lo broadcast 127.255.255.255 proto kernel scope link src 127.0.0.1
local lan0 broadcast 172.16.0.0 proto kernel scope link src 172.16.25.204
local lan0 local 172.16.25.204 proto kernel scope host src 172.16.25.204
local lan0 broadcast 172.16.255.255 proto kernel scope link src
172.16.25.204
main lo local ::1 proto none metric 0
0 lo unreachable default proto kernel metric -1 error -101

Routing Rules:
0:  from all lookup local
32766:  from all lookup main
32767:  from all lookup default
```

Examples

 # show route
(shows the current list of all active IP network routes)

 # monitor route
(like "show route", but automatically refreshes every 10s)

 # show route lan0
(shows all network routes assigned to the physical port "lan0")

 # s r
(short form of "show route")

 # m r
(short form of "monitor route")

3.7.10 syslog - Show System Log File

Purpose

This "show" plugin lists the entries of the system log file (/var/log/messages), a file that contains all events and status changes.

Call and Parameters

 # show syslog [Filter]

or

 # monitor syslog [Filter]

In the "monitor" mode, only the most recent protocol entries are shown, afterwards the command will wait for new entries and prints them as soon as they are created. The CTRL+C key combination aborts waiting for new events and returns to the command prompt.

If a Filter parameter has been added, only those lines in the log file will be printed that contain the given filter string (this is not case sensitive). If no filter is specified, all entries will be listed.

The output of this command looks like this:

```
# show syslog
Dec 15 10:41:08 timeserver root:  Restarting syslog due to configuration
change ...
Dec 15 10:41:08 timeserver syslog-ng[7864]:  Termination requested via
signal, terminating;
Dec 15 10:41:08 timeserver syslog-ng[7864]:  syslog-ng shutting down;
version='2.0.9'
Dec 15 10:41:08 test_tr0_lt04 syslog-ng[22289]:  syslog-ng starting up;
version='2.0.9'
Dec 15 11:19:40 test_tr0_lt04 sshd[5061]:  Accepted password for root from
172.16.3.120 port 41449 ssh2
Dec 15 11:19:40 test_tr0_lt04 sshd[5061]:  pam_unix(sshd:session):  session
opened for user root by (uid=0)
Dec 15 11:19:46 test_tr0_lt04 sshd[5061]:  Received disconnect from
172.16.3.120:  11:  PECL/ssh2 (http://pecl.php.net/packages/ssh2)
Dec 15 11:19:46 test_tr0_lt04 sshd[5061]:  pam_unix(sshd:session):  session
closed for user root
#
```

Examples

 # show syslog
(shows the full system protocol file)

 # show syslog failed
(shows all protocol entries in the log file that contain the string "failed")

 # monitor syslog
(lists the last 20 entries and then waits for new entries, cancel waiting with CTRL+C)

 # s s
(short form of "show syslog")

 # m s
(short form of "monitor syslog")

```
# s
```
(short form of "s s")

```
# m
```
(short form of "m s")

3.7.11 version - Show Current Firmware Version

Purpose

The "version" module shows the firmware version of the currently running firmware image.

Call and Parameters

 # show version

```
Running LTOS V6.16.005 [standar
System Version :  Linux heiko_tr0_lt04 3.7.1 #16 Wed Jul 16 10:33:54 UTC
2014 i586 unknown
```

Examples

```
# show version
```
(shows the firmware version)

```
# s v
```
(shortcut for "show version")

3.8 System Commands

The monitoring of system ressouces like flash or RAM capacity is supported by a group of CLI commands that are described in this section. Most of them are intended to show the status of certain ressources (like "free RAM space"), assisting you with detecting and diagnosing a problem.

3.8.1 reboot - Full System Restart

Purpose

The **reboot** command initiates a restart of the whole system. This includes stopping all services and resetting the CPU. Please note that any unsaved configuration changes are not automatically saved, therefore the system comes back up with the last startup configuration that was saved using **saveconfig**.

It is possible to specify a delay, i.e. the **reboot** process waits for a given time before carrying out the system restart. Such a delay can be applied in the background, allowing a user to continue to work in the foreground, e.g. changing configuration files and applying changes. A backgrounded **reboot** process kan be canceled at any time during the waiting period, allowing a user to set a reboot time before trying to change the system configuration. If one of these changes results in the system becoming unreachable (e.g due to a network IP address configuration error), the backgrounded **reboot** process will automatically restart the system after the specified time and restores the last saved startup configuration, resulting in a restore of the network connectivity. Once the user completed and tested all configuration changes successfully and verified that the system is still reachable, the waiting **reboot** process can be canceled.

reboot notifies all logged in users in active SSH, TELNET and serial console sessions about the reboot and the specified waiting period. This enables everyone to save any changes made and log out correctly or cancel the restart, as long as the **reboot** process is still in a waiting state.

Call and Parameters

reboot [*DELAY*|*stop*]

If a delay is specified, the reboot-command will wait for the given time before restarting the system. This delay can be specified as a simple numeric value representing the number of seconds to wait. It can also be specified in minutes or hours by using a "m" or "h" suffix to the numeric value (see examples).

In order to be able to continue to work in the same SSH/TELNET/serial console session, the **reboot** command can be told to wait in the background instead of blocking the shell prompt. This background mode is enabled by adding a "&" character at the end of the command line.

A waiting **reboot** process can be canceled by specifying "stop" instead of a delay. This can be used to stop a restart process that is waiting in the background but it can also be used to cancel the reboot process of another user.

If no parameter is given, **reboot** will restart immediately, i.e. after the default waiting time of 2 seconds.

Examples

```
# reboot
```
(immediately restarts the system, i.e. after the 2s default waiting period)

```
# reboot 20
```
(restarts in 20 seconds)

```
# reboot 1h
```
(restarts in 1 hour)

```
# reboot 5m &
```
(restarts in 5 minutes, but waits in the background allowing the user to enter additional commands in the

meantime)

```
# reboot stop
```

(cancels any waiting reboot process, no matter if that is a backgrounded process or had been initiated by a different user)

3.8.2 make_noise - Visual and Audio Identification of a Device

Purpose

The **make_noise** command allows to identify a device in a server room or rack via beep sounds and periodical blinking (Alarm LED). This is useful if a device needs to be physically identified, for example in a large server room.

The audio-visual signals can be switched off if the device has a display and front panel buttons. In that case the display shows a note saying that the F2 button can be used to stop this mode. It is also possible to cancel the command by pressing CTRL+C, which will also result in stopping the audio-visual identification mode.

Call and Parameters

 # make_noise

This command does not support any parameters. It causes the device to beep every 2 seconds and switch the red Alarm LED on and off peridocally.

Examples

```
# make_noise
```

(initiates the audio-visual identification mode, can be stopped/canceled by pressing CTRL+C or the F2 front panel button of the device)

4 Text Editors

Manually modifying a configuration file is an often used task within the CLI environment. LTOS V6 provides two different text editors for this: **nano** and **edit**. Both can be used to edit textfiles, the main difference between them is their feature list and the way the functions are accessed. It is completely up to you which one you choose, it may happen that one of the two offers a better compatibility with your terminal software or that you simply prefer the operating concept of one of them.

On earlier versions of LANTIME OS the nano text editor could be started with the command

vi *[filename]*

which is still available on V6 for compatibility reasons. However, this command will show a short note telling you about the two possible editors available on V6 and then asks you to choose which one to use for editing the file *[filename]* you specified on the commandline.

4.1 nano

The nano text editor is a fast, small and easy-to-use opensource program (see http://www.nano-editor.org/ for further information). This editor has been used as the standard CLI tool for modifying text files in earlier LTOS versions (in which it was started using the vi command).

Start and Parameters

 # nano [filename]

starts nano and opens the file [filename]. If no filename is specified, nano will start with an empty file and will ask for a filename when you use the save/close function afterwards.

Using the Editor - Main Functions

Command Keys
The "nano" editor uses key combinations to access its functions. Most key combinations use the control key (CTRL) which has to be held down while pressing and releasing another key to execute a certain editor command.

Saving Modified Files
In order to save the currently modified file you have to press CTRL+O (for WriteOut). After pressing CTRL+O you will be asked for the filename and path where the changed file should be saved. If you want to overwrite the original file, just press ENTER. You can cancel the save function and return to the editor by pressing CTRL+C.

Closing the Editor
With CTRL+X the editor can be closed. If there are unsaved changes, you will be asked whether you want to save the changes (press "Y") or not ("N") and you can cancel leaving the editor by pressing CTRL+C at this point.

Search/Replace
To find a certain search term in the current file, press CTRL+W. If you want to replace a search string, use the CTRL+\ key combination. The nano Editor supports regular expressions as search terms.

More Functions

A number of additional editor commands and functions can be accessed with specific key combinations. A help screen listing all of them is available with the CTRL+G command key combination.

4.2 edit

The edit text editor is a part of the Midnight Commander opensource project and is normally called mcedit (see http://www.midnight-commander.org/ for further information). This editor has a rich feature set, a menu system and color options (if supported by the terminal).

Start and Parameters

edit [**filename**]

starts edit and opens the file [filename]. If no filename is specified, the editor will start with an empty file and will ask for a filename when you use the save/close function afterwards.

Using the Editor - Main Functions

Function Keys

This editor uses Function Keys to perform most program functions. If your terminal does not support sending the correct key codes for the function keys or if you cannot use the function keys for some other reason, you can emulate a function key by pressing the escape key (ESC) first, followed by the digit 1-9 (for F1 to F9) or 0 (for F10). F10 is important as it is used to quit the editor and return to the CLI prompt.

Saving Modified Files
In order to save a modified file the F2 key needs to be pressed. This will not leave the editor. After pressing F2 (or ESC+2), a confirmation dialogue appears in which you can enter "S" (save) or "C" (cancel, do not save).

Closing the Editor
With F10 (or ESC+0) the editor can be closed and you are returned to the CLI prompt. If the currently opened file has been modified and the changes have not been saved yet, the editor will show a dialoge in which you can choose to save the changes and close ("Y"), close with saving any unsaved changes ("N") or cancel and return to the editor ("C").

Search/Replace
In order to search for a certain search string in the file, please press F7 (ESC+7). If a search string needs to be replaced by another string, press F4 to open the search/replace dialogue. This function has a large number of options which can be selected, for example the "prompt on replace" option to bring up a confirmation dialogue before each replacement is performed or the "replace all" flag to select that multiple/all occurrences of the search string shall be replaced.

More Functions
The mcedit Editor has a large number of functions and useful features, most of them are accessible via the on screen menu. In order to open the menu, please press F9 (or ESC+9) and then navigate with the cursor keys and ENTER to select a menu option or ESC to leave the menu and return to the file editor.

4.3 vim

The VImproved text editor ("vim") is a powerful but complex opensource program (see http://www.vim.org/ for further information). It is not recommended for beginners and requires a lot of training and learning to become useful.

Start and Parameters

vim [**filename**]

starts vim and opens the file [filename]. If no filename is specified, vim will start with an empty file and requires you to specify a file name later, when you save the file.

Using the Editor - Main Functions

Editor Commands and Modes
THe "vim" editor is a modal editor and has three basic modes of operation. The "normal mode" is the mode which is active after starting vim from the command line. You can call most editor functions in this mode by pressing a alphanumeric key. Entering the command mode is possible by pressing the colon (":") key. To enter the text edit/insert mode, press "i" for insert or use the Insert key on your keyboard (Ins). You can always return to the "normal mode" by pressing escape (ESC) multiple times.

Saving Modified Files
Saving the current file is performed in command mode. Enter the command "w" and press ENTER to save the file without leaving the editor. After the save operation has been completed, you will return to normal mode.

Closing the Editor
To close the editor, use the "q" command in command mode. If there are unsaved changes, you need to use either the "wq" command (to save and exit) or the "q!" command (to exit without saving). In normal mode you can also press "z" twice to save and exit, without having to enter command mode first.
More Functions
The vim editor is a very powerful text processing editor and offers a large feature set. More about vim and its functions can be found on the Internet. The freely available PDF eBook "The Vim Tutorial and Reference" by Steve Oualline has 800 pages.