MikroTik RouterOS™ v3.0

Reference Manual

Table Of Contents

Configuration Management

Document revision 1.10 (June 22, 2007, 16:49 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

This manual introduces you with commands which are used to perform the following functions:

- system backup;
- system restore from a backup;
- configuration export;
- configuration import;
- system configuration reset.

Description

The configuration backup can be used for backing up MikroTik RouterOS configuration to a binary file, which can be stored on the router or downloaded from it using FTP for future use. The configuration restore can be used for restoring the router's configuration, exactly as it was at the backup creation moment, from a backup file. The restoration procedure assumes the cofiguration is restored on the same router, where the backup file was originally created, so it will create partially broken configuration if the hardware has been changed.

The configuration export can be used for dumping out complete or partial MikroTik RouterOS configuration to the console screen or to a text (script) file, which can be downloaded from the router using FTP protocol. The configuration dumped is actually a batch of commands that add (without removing the existing configuration) the selected configuration to a router. The configuration import facility executes a batch of console commands from a script file.

System reset command is used to erase all configuration on the router. Before doing that, it might be useful to backup the router's configuration.

System Backup

Home menu level: ***/system backup***

Description

The **save** command is used to store the entire router configuration in a backup file. The file is shown in the **/file** submenu. It can be downloaded via ftp to keep it as a backup for your configuration.

To restore the system configuration, for example, after a **/system reset-configuration**, it is possible to upload that file via ftp and load that backup file using **load** command in **/system backup** submenu.

Command Description

load name=[filename] - Load configuration backup from a file

save name=[filename] - Save configuration backup to a file

Example

To save the router configuration to file **test**:

```
[admin@MikroTik] system backup> save name=test
Configuration backup saved
[admin@MikroTik] system backup>
```

To see the files stored on the router:

```
[admin@MikroTik] > file print
  # NAME                           TYPE         SIZE       CREATION-TIME
  0 test.backup                    backup       12567      sep/08/2004 21:07:50
[admin@MikroTik] >
```

Example

To load the saved backup file **test**:

```
[admin@MikroTik] > system backup load name=test
Restore and reboot? [y/N]:
y
Restoring system configuration
System configuration restored, rebooting now
```

Exporting Configuration

Command name: ***/export***

Description

The **export** command prints a script that can be used to restore configuration. The command can be invoked at any menu level, and it acts for that menu level and all menu levels below it. The output can be saved into a file, available for download using FTP.

Command Description

file=[filename] - saves the export to a file

Example

```
[admin@MikroTik] > ip address print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   10.1.0.172/24      10.1.0.0        10.1.0.255      bridge1
 1   10.5.1.1/24        10.5.1.0        10.5.1.255      ether1
[admin@MikroTik] >
```

To make an export file:

```
[admin@MikroTik] ip address> export file=address
[admin@MikroTik] ip address>
```

To see the files stored on the router:

```
[admin@MikroTik] > file print
 # NAME                              TYPE         SIZE        CREATION-TIME
0  address.rsc                       script       315         dec/23/2003 13:21:48
[admin@MikroTik] >
```

Importing Configuration

Command name: ***/import***

Description

The root level command **/import [file_name]** executes a script, stored in the specified file adds the configuration from the specified file to the existing setup. This file may contain any console comands, including scripts. is used to restore configuration or part of it after a **/system reset** event or anything that causes configuration data loss.

Note that it is impossible to import the whole router configuration using this feature. It can only be used to import a part of configuration (for example, firewall rules) in order to spare you some typing.

Command Description

file=[filename] - loads the exported configuration from a file to router

Example

To load the saved export file use the following command:

```
[admin@MikroTik] > import address.rsc
Opening script file address.rsc

Script file loaded and executed successfully
[admin@MikroTik] >
```

Configuration Reset

Command name: ***/system reset-configuration***

Description

The command clears all configuration of the router and sets it to the default including the login name and password ('admin' and no password), IP addresses and other configuration is erased, interfaces will become disabled. After the **reset** command router will reboot.

Command Description

reset - erases router's configuration

Notes

If the router has been installed using netinstall and had a script specified as the initial configuration, the reset command executes this script after purging the configuration. To stop it doing so, you will have to reinstall the router.

Example

```
[admin@MikroTik] > system reset-configuration
Dangerous! Reset anyway? [y/N]: n
action cancelled
[admin@MikroTik] >
```

FTP (File Transfer Protocol) Server

Document revision 2.6 (June 22, 2007, 15:59 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

MikroTik RouterOS implements File Transfer Protocol (FTP) server feature. It is intended to be used for software packages uploading, configuration script exporting and importing procedures, as well as for storing HotSpot servlet pages.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/file***
Standards and Technologies: ***FTP (RFC 959)***
Hardware usage: ***Not significant***

File Transfer Protocol Server

Home menu level: ***/file***

Description

MikroTik RouterOS has an industry standard FTP server facility. It uses ports 20 and 21 for communication with other hosts on the network.

Uploaded files as well as exported configuration or backup files can be accessed under /file menu. There you can delete unnecessary files from the router.

Authorization for FTP service uses router's system user account names and passwords. The **ftp** local user policy controls the access rights to the FTP server.

Property Description

contents (*text*) - file contents (for text files only; size limit - 4kB)

creation-time (*read-only: time*) - item creation date and time

name (*read-only: name*) - item name

package-architecture (*read-only: text*) - RouterOS software package target machine architecture (for package files only)

package-build-time (*read-only: date*) - RouterOS software package build time (for package files only)

package-name (*read-only: text*) - RouterOS software package name (for package files only)

package-version (*read-only: text*) - RouterOS software package version number (for package files only)

size (*read-only: integer*) - package size in bytes

type (*read-only: text*) - item type. Few file types are recognized by extension: backup, directory, package, script, ssh key, but other files are just marked by their extension (.html file, for example)

Command Description

print - shows a list of files stored - shows contents of files less that 4kB long - offers to edit file's contents with editor - sets the file's contents to 'content'

MAC Level Access (Telnet and Winbox)

Document revision 2.5 (June 22, 2007, 15:59 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

MAC telnet is used to provide access to a router that has no IP address set. It works just like IP telnet. MAC telnet is possible between two MikroTik RouterOS routers only.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/tool, /tool mac-server***
Standards and Technologies: ***MAC Telnet***
Hardware usage: ***Not significant***

MAC Telnet Server

Home menu level: ***/tool mac-server***

Property Description

interface (*name* | *all*; default: **all**) - interface name to which the mac-server clients will connect

- **all** - all interfaces

Notes

There is an interface list in this submenu level. If you add some interfaces to this list, you allow MAC telnet to that interface. Disabled (**disabled=yes**) item means that interface is not allowed to accept MAC telnet sessions on that interface. **all** interfaces iss the default setting to allow MAC teltet on any interface.

Example

To enable MAC telnet server on **ether1** interface only:

```
[admin@MikroTik] tool mac-server> print
Flags: X - disabled
 #   INTERFACE
 0   all
[admin@MikroTik] tool mac-server> remove 0
[admin@MikroTik] tool mac-server> add interface=ether1 disabled=no
[admin@MikroTik] tool mac-server> print
Flags: X - disabled
 #   INTERFACE
 0   ether1
[admin@MikroTik] tool mac-server>
```

MAC WinBox Server

Home menu level: ***/tool mac-server mac-winbox***

Property Description

interface (*name* | *all*; default: **all**) - interface name to which it is alowed to connect with Winbox using MAC-based protocol

- **all** - all interfaces

Notes

There is an interface list in this submenu level. If you add some interfaces to this list, you allow MAC Winbox to that interface. Disabled (**disabled=yes**) item means that interface is not allowed to accept MAC Winbox sessions on that interface.

Example

To enable MAC Winbox server on **ether1** interface only:

```
[admin@MikroTik] tool mac-server mac-winbox> print
Flags: X - disabled
 #   INTERFACE
 0   all
[admin@MikroTik] tool mac-server mac-winbox> remove 0
[admin@MikroTik] tool mac-server mac-winbox> add interface=ether1 disabled=no
[admin@MikroTik] tool mac-server mac-winbox> print
Flags: X - disabled
 #   INTERFACE
 0   ether1
```

```
[admin@MikroTik] tool mac-server mac-winbox>
```

Monitoring Active Session List

Home menu level: ***/tool mac-server sessions***

Property Description

interface (*read-only: name*) - interface to which the client is connected to

src-address (*read-only: MAC address*) - client's MAC address

uptime (*read-only: time*) - how long the client is connected to the server

Example

To see active MAC Telnet sessions:

```
[admin@MikroTik] tool mac-server sessions> print
 # INTERFACE SRC-ADDRESS       UPTIME
 0 wlan1     00:0B:6B:31:08:22 00:03:01
[admin@MikroTik] tool mac-server sessions>
```

MAC Scan

Command name: ***/tool mac-scan***

Description

This command discovers all devices, which support MAC telnet protocol on the given network.

Property Description

(*name*) - interface name to perform the scan on

MAC Telnet Client

Command name: ***/tool mac-telnet***

Property Description

(*MAC address*) - MAC address of a compatible device

Example

```
[admin@MikroTik] > /tool mac-telnet 00:02:6F:06:59:42
Login: admin
Password:
Trying 00:02:6F:06:59:42...
Connected to 00:02:6F:06:59:42

  MMM      MMM       KKK                          TTTTTTTTTTT      KKK
  MMMM    MMMM       KKK                          TTTTTTTTTTT      KKK
  MMM MMMM MMM  III  KKK  KKK  RRRRRR     OOOOOO      TTT     III  KKK  KKK
  MMM  MM  MMM  III  KKKKK     RRR  RRR  OOO  OOO     TTT     III  KKKKK
  MMM      MMM  III  KKK KKK   RRRRRR    OOO  OOO     TTT     III  KKK KKK
  MMM      MMM  III  KKK  KKK  RRR  RRR   OOOOOO      TTT     III  KKK  KKK

  MikroTik RouterOS 3.0beta10 (c) 1999-2007       http://www.mikrotik.com/

Terminal linux detected, using multiline input mode
[admin@MikroTik] >
```

Serial Console and Terminal

Document revision 2.3 (June 25, 2007, 19:43 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The Serial Console and Terminal are tools, used to communicate with devices and other systems that are interconnected via serial port. The serial terminal may be used to monitor and configure many devices - including modems, network devices (including MikroTik routers), and any device that can be connected to a serial (asynchronous) port.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: Command name: ***/system console, /system serial-terminal***
Standards and Technologies: ***RS-232***
Hardware usage: ***Not significant***

Description

The Serial Console feature is for configuring direct-access configuration facilities (monitor/keyboard and serial port) that are mostly used for initial or recovery configuration.

If you do not plan to use a serial port for accessing another device or for data connection through a modem, you can configure it as a serial console. The first serial port is configured as a serial console, but you can choose to unconfigure it to free it for other applications. A free serial port can also be used to access other routers' (or other equipment, like switches) serial consoles from a MikroTik RouterOS router. A special null-modem cable is needed to connect two hosts (like, two PCs, or two routers; not modems). Note that a terminal emulation program (e.g., **HyperTerminal** on Windows or **minicom** on linux) is required to access the serial console from another computer.

Several customers have described situations where the Serial Terminal (managing side) feature would be useful:

- on a mountaintop, where a MikroTik wireless installation sits next to equipment (including switches and Cisco routers) that can not be managed in-band (by telnet through an IP network)
- monitoring weather-reporting equipment through a serial port
- connection to a high-speed microwave modem that needed to be monitored and managed by a serial connection

With the serial-terminal feature of the MikroTik, up to 132 (and, maybe, even more) devices can be monitored and controlled.

Serial Console Configuration

Description

A special null-modem cable should be used for connecting to the serial console from another computer. The Serial Console cabling diagram for DB9 connectors is as follows:

Router Side (DB9f)	Signal	Direction	Side (DB9f)
1, 6	CD, DSR	IN	4
2	RxD	IN	3
3	TxD	OUT	2
4	DTR	OUT	1, 6
5	GND	-	5
7	RTS	OUT	8
8	CTS	IN	7

Note that the above diagram will not work if the software is configured to do hardware flow control, but the hardware does not support it (e.g., some RouterBOARD models have reduced seral port functionality). If this is the case, either turn off the hardware flow control or use a null-modem cable with loopback, which will simulate the other device's handshake signals with it's own. The diagram for such cable is as follows:

Router Side (DB9f)	Signal	Direction	Side (DB9f)
1, 4, 6	CD, DTR, DSR	LOOP	1, 4, 6
2	RxD	IN	3

3	**TxD**	**OUT**	**2**
5	**GND**	**-**	**5**
7, 8	**RTS, CTS**	**LOOP**	**7, 8**

Note that although it is recommended to have 5-wire cable for this connection, in many cases it is enough to have 3 wires (for unlooped signals only), leaving both loops to exist only inside the connectors. Other connection schemes exist as well.

Configuring Console

Home menu level: ***/system console***

Property Description

enabled (*yes* | *no*; default: **no**) - whether serial console is enabled or not

free (*read-only: flag*) - console is ready for use

port (*name*; default: **serial0**) - which port should the serial terminal listen to

term (*text*) - terminal type

used (*read-only: flag*) - console is in use

vcno (*read-only: integer*) - number of virtual console - [Alt]+[F1] represents '1', [Alt]+[F2] - '2', etc.

wedged (*read-only: flag*) - console is currently not available

Example

To disable all virtual consoles (available through the direct connection with keyboard and monitor) extept for the first one:

```
[admin@MikroTik] system console> print
Flags: X - disabled, W - wedged, U - used, F - free
 #   PORT    VCNO       TERM
 0 F serial0            MyConsole
 1 U         1          linux
 2 F         2          linux
 3 F         3          linux
 4 F         4          linux
 5 F         5          linux
 6 F         6          linux
 7 F         7          linux
 8 F         8          linux
[admin@MikroTik] system console> disable 2,3,4,5,6,7,8
[admin@MikroTik] system console> print
Flags: X - disabled, W - wedged, U - used, F - free
 #   PORT    VCNO       TERM
 0 F serial0            MyConsole
 1 U         1          linux
 2 X         2          linux
 3 X         3          linux
 4 X         4          linux
 5 X         5          linux
 6 X         6          linux
 7 X         7          linux
 8 X         8          linux
[admin@MikroTik] system console>
```

To check if the configuration of the serial port:

```
[admin@MikroTik] system serial-console> /port print detail
  0 name=serial0 used-by=Serial Console baud-rate=9600 data-bits=8 parity=none
    stop-bits=1 flow-control=none

  1 name=serial1 used-by="" baud-rate=9600 data-bits=8 parity=none stop-bits=1
    flow-control=none

[admin@MikroTik] system serial-console>
```

Using Serial Terminal

Command name: ***/system serial-terminal***

Description

The command is used to communicate with devices and other systems that are connected to the router via serial port.

All keyboard input is forwarded to the serial port and all data from the port is output to the connected device. After exiting with [Ctrl]+[Q], the control signals of the port are lowered. The speed and other parameters of serial port may be configured in the **/port** directory of router console. No terminal translation on printed data is performed. It is possible to get the terminal in an unusable state by outputting sequences of inappropriate control characters or random data. Do not connect to devices at an incorrect speed and avoid dumping binary data.

Property Description

port (*name*) - port name to use

Notes

The serial port to be used as a serial terminal needs to be free (e.g., there should not be any serial consoles, LCD or other configuration). Chack the previous chapter to see how to disable serial console on a particular port. Use `/port print` command to see if some other application is still using the port.

[Ctrl]+[Q] and [Ctrl]+[X] have special meaning and are used to provide a possibility of exiting from nested serial-terminal sessions:

To send [Ctrl]+[X] to to serial port, press [Ctrl]+[X] [Ctrl]+[X]

To send [Ctrl]+[Q] to to serial port, press [Ctrl]+[X] [Ctrl]+[Q]

Example

To connect to a device connected to the **serial1** port:

```
[admin@MikroTik] system> serial-terminal serial1

[Type Ctrl-Q to return to console]
[Ctrl-X is the prefix key]
```

Console Screen

Home menu level: ***/system console screen***

Description

This facility is created to change line number per screen if you have a monitor connected to router.

Property Description

line-count (*25* | *40* | *50*) - number of lines on monitor

Notes

This parameter is applied only to a monitor, connected to the router.

Example

To set monitor's resolution from 80x25 to 80x40:

```
[admin@MikroTik] system console screen> set line-count=40
[admin@MikroTik] system console screen> print
    line-count: 40
[admin@MikroTik] system console screen>
```

Software Package and Version Management

Document revision 1.5 (June 29, 2007, 19:19 GMT)

This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS is distributed in the form of software packages. The basic functionality of the router and the operating system itself is provided by the **system** software package. Other packages contain additional software features as well as support to various network interface cards.

Specifications

License required: ***level1***
Home menu level: ***/system package***
Standards and Technologies: ***FTP***
Hardware usage: ***Not significant***

Description

Features

The modular software package system of MikroTik RouterOS has the following features:

- Ability to extend RouterOS functions by installing additional software packages
- Optimal usage of the storage space and memory resources by employing modular/compressed system
- Unused software packages can be uninstalled
- The RouterOS functions and the system itself can be easily upgraded
- Multiple packages can be installed at once
- The package dependency is checked before installing a software package. The package will not be installed, if the required software package is missing
- The version of the feature package should be the same as that of the **system** package
- The packages can be uploaded on the router using ftp and installed only when the router is going for shutdown during the reboot process
- If the software package file can be uploaded to the router, then the disk space is sufficient for the installation of the package
- The system can be downgraded to an older version by uploading the needed packages to router via FTP binary mode. After that, execute command **/system package downgrade**

Installation (Upgrade)

Description

Installation or upgrade of the MikroTik RouterOS software packages can be done by uploading the appropriate version of the software package to the router and rebooting it. All packaged must have the same version number as the system package, otherwise they will not be installed (and will be uninstalled if you are just upgrading the system package).

The software package files are compressed binary files, which can be downloaded from the MikroTik's web page download section. The full name of the software package consists of a descriptive name, version number and extension **.npk**, e.g., **system-2.9.11.npk**. Package **routeros-x86** contains all necessary packages for RouterOS installation and upgrading for RouterBOARD 200 and PC. Package **routeros-rb500** contains all necessary packages for RouterOS installation and upgrading for RouterBOARD 100/500. These packages are preferred installation and upgrading method.

You should check the available hard disk space prior to uploading the package files by issuing **/system resource print** command. If there is not enough free disk space for storing the upgrade packages, it can be freed up by uninstalling some software packages, which provide functionality not required for upgrade to complete. If you have a sufficient amount of free space for storing the upgrade packages, connect to the router using ftp. Use user name and password of a user with full access privileges.

Step-by-Step

- Connect to the router using ftp client
- Select the BINARY mode file transfer
- Upload the software package files to the router
- Check the information about the uploaded software packages using the **/file print** command
- Reboot the router by issuing the **/system reboot** command or by pressing **Ctrl+Alt+Del** keys at the router's local console
- After reboot, verify that the packages were installed correctly by issuing **/system package print** command. If the packages have not been installed, check the logs to see what went wrong.

Notes

The packages uploaded to the router should retain the original name and also be in lowercase.

The installation/upgrade process is shown on the console screen (monitor) attached to the router and on the serial console.

The Free Demo License does not allow software upgrades using ftp. You should do a complete reinstall, or purchase the license.

Before upgrading the router, please check the current version of the system package and the additional software packages. The versions of additional packages must match the version number of the system software package. The version of the MikroTik RouterOS system software is shown before the console login prompt and right after you log in. Information about the version numbers and build time of the installed MikroTik RouterOS software packages can be obtained using the **/system package print** command.

Do not use **routeros-x86** and **routeros-rb500** packges to upgrade from version 2.8 or older. To upgrade use regular packages.

Test packages, like **wireless-test**, **rstp-bridge-test** and so on, are included in the **routeros-x86** and **routeros-rb500** packages, but disabled by default.

Few special-purpose packages, like LCD, are not included in the combined packages, and you may need to download them separately.

Uninstallation

Command name: ***/system package uninstall***

Description

Usually, you do not need to uninstall software packages. However, if you have installed a wrong package, or you need additional free space to install a new one, you may need to uninstall some unused packages.

Notes

If a package is marked for uninstallation, but it is required for another (dependent) package, then the marked package cannot be uninstalled. You should uninstall the dependent package too. For the list of package dependencies see the 'Software Package List' section below. The system package will not be uninstalled even if marked for uninstallation.

Example

Suppose we need to uninstall **security** package from the router:

```
[admin@MikroTik] system package> print
Flags: X - disabled
 #   NAME                     VERSION                  SCHEDULED
 0   routeros-rb500           3.0beta10
 1   system                   3.0beta10
 2 X ipv6                     3.0beta10
 3   ntp                      3.0beta10
 4   wireless                 3.0beta10
 5   dhcp                     3.0beta10
 6   routing                  3.0beta10
 7   routerboard              3.0beta10
 8   advanced-tools           3.0beta10
 9   hotspot                  3.0beta10
10   ppp                      3.0beta10
11   security                 3.0beta10
[admin@MikroTik] system package> uninstall security
[admin@MikroTik] system package> .. reboot
```

Downgrading

Command name: ***/system package downgrade***

Description

Downgrade option allows you to downgrade the software via FTP without losing your license key or reinstalling the router. It is not recommended to use older versions, however, if the newest version introduced some unwanted behavior, you may try to downgrade. If you send a support question, you will probably be asked to upgrade to the latest version.

Step-by-Step

- Connect to the router using ftp client
- Select the BINARY mode file transfer
- Upload the software package files to the router
- Check the information about the uploaded software packages using the **/file print** command

- Execute command **/system package downgrade.** The router will downgrade and reboot.
- After reboot, verify that the packages were installed correctly by issuing **/system package print** command

Command Description

downgrade - this command asks your confirmation and reboots the router. After reboot the software is downgraded (if all needed packages were uploaded to the router)

Example

To downgrade the RouterOS (assuming that all needed packages are already uploaded):

```
[admin@MikroTik] system package> downgrade
Router will be rebooted. Continue? [y/N]:
y
system will reboot shortly
```

Disabling and Enabling

Command name: ***/system package disable, /system package enable***

Description

You can disable packages making them invisible for the system and later enable them, bringing the system back to the previous state. It is useful if you don't want to uninstall a package, but just turn off its functionality. This will save the RAM and processor resources for other applications, but will not free the diskspace used by the package files.

Notes

If a package is marked for disabling, but it is required for another (dependent) package, then the marked package cannot be disabled. You should disable or uninstall the dependent package too. For the list of package dependencies see the 'Software Package List' section below.

If any of the test packages will be enabled (for example wireless-test and routing-test packages, that are included in routeros-x86.npk and routeros-rb500.npk) system automaticly will disable regular packages that conflict with them.

Example

Suppose we need to test **ipv6** package features:

```
[admin@MikroTik] system package> print
Flags: X - disabled
 #   NAME                      VERSION                   SCHEDULED
 0   routeros-rb500            3.0beta10
 1   system                    3.0beta10
 2 X ipv6                      3.0beta10
 3   ntp                       3.0beta10
 4   wireless                  3.0beta10
 5   dhcp                      3.0beta10
 6   routing                   3.0beta10
 7   routerboard              3.0beta10
 8   advanced-tools           3.0beta10
 9   hotspot                  3.0beta10
10   ppp                      3.0beta10
11   security                 3.0beta10
[admin@MikroTik] system package> enable ipv6
[admin@MikroTik] system package> .. reboot
```

Unscheduling

Command name: ***/system package unschedule***

Description

Unschedule option allows to cancel pending uninstall, disable or enable actions for listed packages.

Notes

packages marked for uninstallation, disabling or enabling on reboot in column "schedule" will have a note, warning about changes.

Example

Suppose we need to cancel **security** package uninstallation action scheduled on reboot:

```
[admin@MikroTik] system package> print
Flags: X - disabled
 #   NAME                     VERSION                  SCHEDULED
 0   routeros-rb500           3.0beta10
 1   system                   3.0beta10
 2 X ipv6                     3.0beta10
 3   ntp                      3.0beta10
 4   wireless                 3.0beta10
 5   dhcp                     3.0beta10
 6   routing                  3.0beta10
 7   routerboard              3.0beta10
 8   advanced-tools           3.0beta10
 9   hotspot                  3.0beta10
10   ppp                      3.0beta10
11   security                 3.0beta10                scheduled for uninstall
[admin@MikroTik] system package> unschedule security
[admin@MikroTik] system package>
```

System Upgrade

Home menu level: ***/system upgrade***

Description

This submenu gives you the ability to download RouterOS software packages from a remote RouterOS router.

Step-by-Step

- Upload desired RouterOS packages to a router (not the one that you will upgrade).
- Add this router's IP address, user name and password to **/system upgrade**

upgrade-package-source on the router(s) you will be upgrading. This step will only be needed once, and you may continue using the same package source in future to upgrade the router(s) again. See the next section for details.

- Refresh available software package list **/system upgrade refresh**
- See available packages, using **/system upgrade print** command
- Download selected or all packages from the remote router, using the **download** or **download-all** command

Property Description

name (*read-only: name*) - package name

source (*read-only: IP address*) - source IP address of the router from which the package list entry is retrieved

status (*read-only: available* | *scheduled* | *downloading* | *downloaded* | *installed*) - package status

version (*read-only: text*) - version of the package

Command Description

download - download packages from list by specifying their numbers

download-all - download all packages that are needed for the upgrade (packages which are listed in the /system package print command output)

refresh - updates currently available package list

Example

See the available packages:

```
[admin@MikroTik] system upgrade> refresh
[admin@MikroTik] system upgrade> print
 # SOURCE          NAME             VERSION      STATUS      COMPLETED
 0 192.168.25.8    routeros-x86     2.9.44       available
 1 192.168.25.8    routeros-rb500   3.0beta10    available
[admin@MikroTik] system upgrade>
```

To upgrade chosen packages:

```
[admin@MikroTik] system upgrade> download 1
[admin@MikroTik] system upgrade> print
 # SOURCE          NAME             VERSION      STATUS      COMPLETED
 0 192.168.25.8    routeros-x86     2.9.44       available
 1 192.168.25.8    routeros-rb500   3.0beta10    downloading 16 %
[admin@MikroTik] system upgrade>
```

Adding Package Source

Home menu level: ***/system upgrade upgrade-package-source***

Description

In this submenu you can add remote routers from which to download RouterOS software packages.

Property Description

address (*IP address*) - source IP address of the router from which the package list entry will be retrieved

password (*text*) - password of the remote router

user (*text*) - username of the remote router

Notes

After specifying a remote router in **/system upgrade upgrade-package-source**, you can type **/system upgrade refresh** to refresh the package list and **/system upgrade print** to see all available packages.

Example

To add a router with IP address **192.168.25.8**, username **admin** and no password:

```
[admin@MikroTik] system upgrade upgrade-package-source> add \
\... address=192.168.25.8 user=admin
password:
[admin@MikroTik] system upgrade upgrade-package-source> print
# ADDRESS         USER
0 192.168.25.8    admin
[admin@MikroTik] system upgrade upgrade-package-source>
```

Software Package List

Description

System Software Package

The **system** software package provides the basic functionality of the MikroTik RouterOS, namely:

- IP address management, ARP, static IP routing, policy routing, firewall (packet filtering, content filtering, masquerading, and static NAT), traffic shaping (queues), IP traffic accounting, MikroTik Neighbour Discovery, IP Packet Packing, DNS client settings, IP service (servers)
- Ethernet interface support
- IP over IP tunnel interface support
- Ethernet over IP tunnel interface support
- driver management for Ethernet ISA cards
- serial port management
- local user management
- export and import of router configuration scripts
- backup and restore of the router's configuration
- undo and redo of configuration changes

- network diagnostics tools (ping, traceroute, bandwidth tester, traffic monitor)
- bridge support
- system resource management
- package management
- telnet client and server
- local and remote logging facility
- winbox server as well as winbox executable with some plugins

After installing the MikroTik RouterOS, a free license should be obtained from MikroTik to enable the basic system functionality.

Additional Software Feature Packages

The table below shows additional software feature packages, extended functionality provided by them, the required prerequisites and additional licenses, if any.

Name	Contents	Prerequisites
advanced-tools	email client, pingers, netwatch and other utilities	none
calea	Call Content Connection (CCC) data retention server for CALEA compliance	none
arlan	support for legacy DSSS 2.4GHz 2mbps Aironet ISA cards	none
dhcp	DHCP server and client support	none
dude	Dude server	none
gps	support for GPS devices	none
hotspot	HotSpot gateway	none
ipv6	IPv6 protocol	none
isdn	support for ISDN devices	ppp
lcd	support for informational LCD display	none
ntp	Network Time Protocol	none
ppp	support for PPP, PPTP, L2TP, PPPoE and ISDN PPP	none
radiolan	support for 5.8GHz RadioLAN cards	none
routerboard	support for	none

	RouterBoard-specific functions and utilities	
routing	support for RIP, OSPF and BGP4	none
security	support for IPSEC, SSH and secure WinBox connections	none
synchronous	support for Frame Relay and Moxa C101, Moxa C502, Farsync, Cyclades PC300, LMC SBE and XPeed synchronous cards	none
thinrouter-pcipc	forces PCI-to-CardBus Bridge to use IRQ 11 as in ThinRouters	none
ups	support for APC Smart UPS	none
user-manager	embedded RADIUS server with web interface	none
wireless	support for Cisco Aironet, PrismII and Atheros wireless cards	none

SSH (Secure Shell) Server and Client

Document revision 2.1 (July 5, 2007, 12:16 GMT)

This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

SSH Client authenticates server and encrypts traffic between the client and server. You can use SSH just the same way as telnet - you run the client, tell it where you want to connect to, give your username and password, and everything is the same after that. After that you won't be able to tell that you're using SSH. The SSH feature can be used with various SSH Telnet clients to securely connect to and administrate the router. Apart form regular password-based authentication, preshared key file may be used to authenticate a user.

The MikroTik RouterOS supports:

- SSH 1.3, 1.5, and 2.0 protocol standards
- server functions for secure administration of the router
- telnet session termination with 40 bit RSA SSH encryption is supported
- secure ftp is supported
- preshared DAS key authentication

The MikroTik RouterOS has been tested with the following SSH telnet terminals:

- MikroTik RouterOS embedded SSH client
- PuTTY

- Secure CRT
- OpenSSH GNU/Linux client

Specifications

Packages required: ***security***
License required: ***level1***
Home menu level: ***/system ssh***
Standards and Technologies: ***SSH***
Hardware usage: ***Not significant***

Additional Documents

- http://www.freessh.org/

SSH Server

Home menu level: ***/ip service***

Description

SSH Server is already up and running after MikroTik router installation. The default port of the service is 22. You can set a different port number or disable the service if you do not need it. See the **System Services** manual for the detailed instructions.

SSH Client

Command name: ***/system ssh***

Property Description

port (*integer*; default: **22**) - which TCP port to use for SSH connection to a remote host

user (*text*; default: **admin**) - username for the SSH login

Example

```
[admin@MikroTik] > /system ssh 192.168.0.1 user=admin
admin@192.168.0.1's password:

  MMM      MMM       KKK                          TTTTTTTTTTT      KKK
  MMMM    MMMM       KKK                          TTTTTTTTTTT      KKK
  MMM MMMM MMM  III  KKK  KKK  RRRRRR     OOOOOO      TTT     III  KKK  KKK
  MMM  MM  MMM  III  KKKKK     RRR  RRR  OOO  OOO     TTT     III  KKKKK
  MMM      MMM  III  KKK KKK   RRRRRR    OOO  OOO     TTT     III  KKK KKK
  MMM      MMM  III  KKK  KKK  RRR  RRR   OOOOOO      TTT     III  KKK  KKK

  MikroTik RouterOS 3.0beta10 (c) 1999-2007       http://www.mikrotik.com/

Terminal xterm detected, using multiline input mode
[admin@MikroTik] >
```

SSH Preshated Key

Home menu level: ***/user ssh-keys***

Description

You can use DSA keys (only DSA keys are supported) instead of password to log into the router. This method may be preferred for automated systems that congifure router(s) with SSH protocol using RouterOS console language. It is also useful if you just don't like remembering dozens of passwords and entering them to the login prompt all the time.

Property Description

key-owner (*read-only: text*) - remote user, as specifie in key file

user (*name*) - local user to associate the key with

Command Description

import - import a DSA key file (*name*) - filename to import the SSH key from (*name*) - local user to associate the key with

Notes

Only openssh DSA keys are supported. If you use puttygen, convert generated keys to right type.

Example

Generating the DSA key on a UNIX machine:

```
sh$ ssh-keygen -t dsa -f ./id_dsa
Generating public/private dsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ./id_dsa.
Your public key has been saved in ./id_dsa.pub.
The key fingerprint is:
91:d7:08:be:b6:a1:67:5e:81:02:cb:4d:47:d6:a0:3b admin-ssh@beka
```

Now, after you upload the key file onto the router, you can import it:

```
[admin@MikroTik] user ssh-keys> import file=id_dsa.pub user=admin-ssh
[admin@MikroTik] user ssh-keys> print
 # USER                  KEY-OWNER
 0 admin-ssh             admin-ssh@beka
[admin@MikroTik] user ssh-keys>
```

Telnet Server and Client

Document revision 2.4 (July 5, 2007, 13:33 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

MikroTik RouterOS has a build-in Telnet server and client features. These two are used to communicate with other systems over a network.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/system, /ip service***
Standards and Technologies: ***Telnet (RFC 854)***
Hardware usage: ***Not significant***

Telnet Server

Home menu level: ***/ip service***

Description

Telnet protocol is intended to provide a fairly general, bi-directional, eight-bit byte oriented communications facility. The main goal is to allow a standard method of interfacing terminal devices to each other.

MikroTik RouterOS implements industry standard Telnet server. It uses port 23, which must not be disabled on the router in order to use the feature.

You can enable/disable this service or allow the use of the service to certain IP addresses. See the **System Services** manual for the detailed instructions.

Telnet Client

Command name: ***/system telnet***

Description

MikroTik RouterOS telnet client is used to connect to other hosts in the network via Telnet protocol.

Property Description

(*IP address*) - IP address of the Telnet server to connect to

(*port*; default: **23**) - TCP port to connect to (if differs from the standard TCP port 23). May be useful to connect to SMTP or HTTP servers for debugging purposes

Example

An example of Telnet connection:

```
[admin@MikroTik] > system telnet 172.16.0.1
Trying 172.16.0.1...
Connected to 172.16.0.1.
Escape character is '^]'.

MikroTik v2.9
Login: admin
Password:

  MMM      MMM       KKK                          TTTTTTTTTTT      KKK
  MMMM    MMMM       KKK                          TTTTTTTTTTT      KKK
  MMM MMMM MMM  III  KKK  KKK  RRRRRR     OOOOOO      TTT     III  KKK  KKK
  MMM  MM  MMM  III  KKKKK     RRR  RRR  OOO  OOO     TTT     III  KKKKK
  MMM      MMM  III  KKK KKK   RRRRRR    OOO  OOO     TTT     III  KKK KKK
  MMM      MMM  III  KKK  KKK  RRR  RRR   OOOOOO      TTT     III  KKK  KKK

  MikroTik RouterOS 2.9 (c) 1999-2004              http://www.mikrotik.com/


Terminal unknown detected, using single line input mode
[admin@MikroTik] >
```

IP Addresses and ARP

Document revision 1.5 (September 10, 2007, 12:55 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The following Manual discusses IP address management and the Address Resolution Protocol settings. IP addresses serve as identification when communicating with other network devices using the TCP/IP version 4 protocol. In turn, communication between devices in one physical network proceeds with the help of Address Resolution Protocol and ARP addresses.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip address, /ip arp***
Standards and Technologies: ***IPv4***, ***ARP***
Hardware usage: ***Not significant***

IP Addressing

Home menu level: ***/ip address***

Description

IP addresses serve for a general host identification purposes in IP networks. Typical (IPv4) address consists of four octets. For proper addressing the router also needs the network mask value, *id est* which bits of the complete IP address refer to the address of the host, and which - to the address of the network. The network address value is calculated by binary **AND** operation from network mask and IP address values. It's also possible to specify IP address followed by slash "/" and the amount of bits that form the network address.

In most cases, it is enough to specify the address, the netmask, and the interface arguments. The network prefix and the broadcast address are calculated automatically.

It is possible to add multiple IP addresses to an interface or to leave the interface without any addresses assigned to it. In case of bridging or PPPoE connection, the physical interface may bot have any address assigned, yet be perfectly usable. Putting an IP address to a physical interface included in a bridge would mean actually putting it on the bridge interface itself. You can use **/ip address print detail** to see to which interface the address belongs to.

MikroTik RouterOS has following types of addresses:

- **Static** - manually assigned to the interface by a user
- **Dynamic** - automatically assigned to the interface by DHCP or an estabilished PPP connections

Property Description

actual-interface (*read-only: name*) - name of the actual interface the logical one is bound to. For example, if the physical interface you assigned the address to, is included in a bridge, the actual interface will show that bridge.

address (*IP address*) - IP address

broadcast (*IP address*; default: **255.255.255.255**) - broadcasting IP address, calculated by default from an IP address and a network mask

disabled (yes | no; default: **no**) - specifies whether the address is disabled or not

interface (*name*) - interface name the IP address is assigned to

netmask (*IP address*; default: **0.0.0.0**) - delimits network address part of the IP address from the host part

network (*IP address*; default: **0.0.0.0**) - IP address for the network. For point-to-point links it should be the address of the remote end

Notes

You cannot have two different IP addresses from the same network assigned to the router. *Exempli gratia*, the combination of IP address **10.0.0.1/24** on the **ether1** interface and IP address **10.0.0.132/24** on the **ether2** interface is invalid (unless both interfaces are bridged together), because both addresses belong to the same network **10.0.0.0/24**. Use addresses from different networks on different interfaces.

Example

```
[admin@MikroTik] ip address> add address=10.10.10.1/24 interface=ether2
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
```

```
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   2.2.2.1/24         2.2.2.0         2.2.2.255       ether2
  1   10.5.7.244/24      10.5.7.0        10.5.7.255      ether1
  2   10.10.10.1/24      10.10.10.0      10.10.10.255    ether2
[admin@MikroTik] ip address>
```

Address Resolution Protocol

Home menu level: ***/ip arp***

Description

Even though IP packets are addressed using IP addresses, hardware addresses must be used to actually transport data from one host to another. Address Resolution Protocol is used to map OSI level 3 IP addreses to OSI level 2 MAC addreses. Router has a table of currently used ARP entries. Normally the table is built dynamically, but to increase network security, it can be partialy or completely built statically by means of adding static entries.

Property Description

address (*IP address*) - IP address to be mapped

interface (*name*) - interface name the IP address is assigned to

mac-address (*MAC address*; default: **00:00:00:00:00:00**) - MAC address to be mapped to

Notes

Maximal number of ARP entries is 8192.

If ARP feature is turned off on the interface, i.e., **arp=disabled** is used, ARP requests from clients are not answered by the router. Therefore, static arp entry should be added to the clients as well. For example, the router's IP and MAC addresses should be added to the Windows workstations using the **arp** command:

```
C:\> arp -s 10.5.8.254  00-aa-00-62-c6-09
```

If **arp** property is set to **reply-only** on the interface, then router only replies to ARP requests. Neighbour MAC addresses will be resolved using **/ip arp** statically, but there will be no need to add the router's MAC address to other hosts' ARP tables.

Example

```
[admin@MikroTik] ip arp> add address=10.10.10.10 interface=ether2 mac-address=06 \
\... :21:00:56:00:12
[admin@MikroTik] ip arp> print
Flags: X - disabled, I - invalid, H - DHCP, D - dynamic
  #   ADDRESS         MAC-ADDRESS       INTERFACE
  0 D 2.2.2.2         00:30:4F:1B:B3:D9 ether2
  1 D 10.5.7.242      00:A0:24:9D:52:A4 ether1
  2   10.10.10.10     06:21:00:56:00:12 ether2
[admin@MikroTik] ip arp>
```

If static arp entries are used for network security on an interface, you should set arp to 'reply-only' on that interface. Do it under the relevant **/interface** menu:

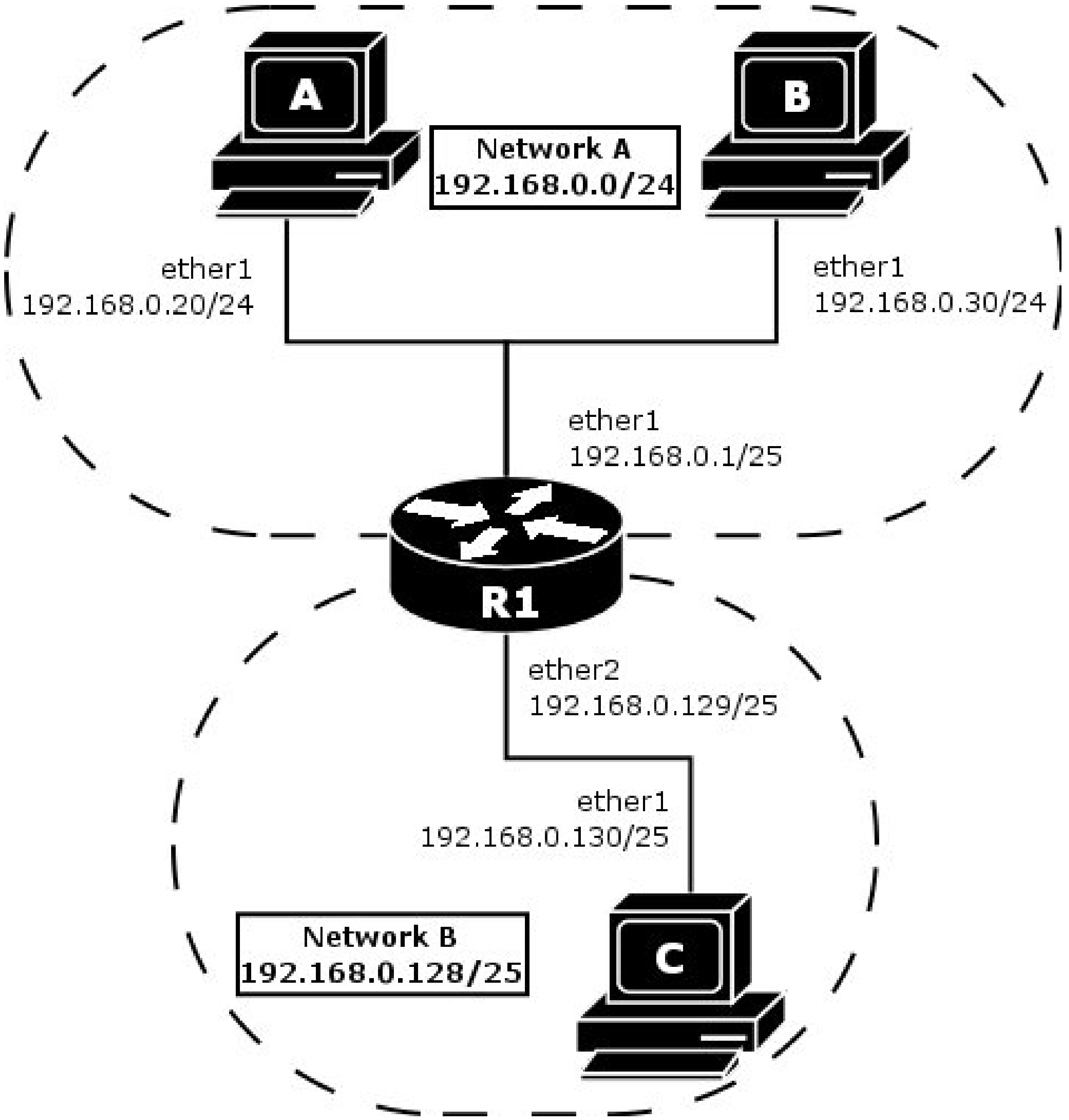
```
[admin@MikroTik] ip arp> /interface ethernet set ether2 arp=reply-only
```

```
[admin@MikroTik] ip arp> print
Flags: X - disabled, I - invalid, H - DHCP, D - dynamic
  #   ADDRESS         MAC-ADDRESS       INTERFACE
  0 D 10.5.7.242      00:A0:24:9D:52:A4 ether1
  1   10.10.10.10     06:21:00:56:00:12 ether2

[admin@MikroTik] ip arp>
```

Proxy-ARP feature

Description

A router with properly configured proxy ARP feature acts like a transparent ARP proxy between directly connected networks. Consider the following network diagram:



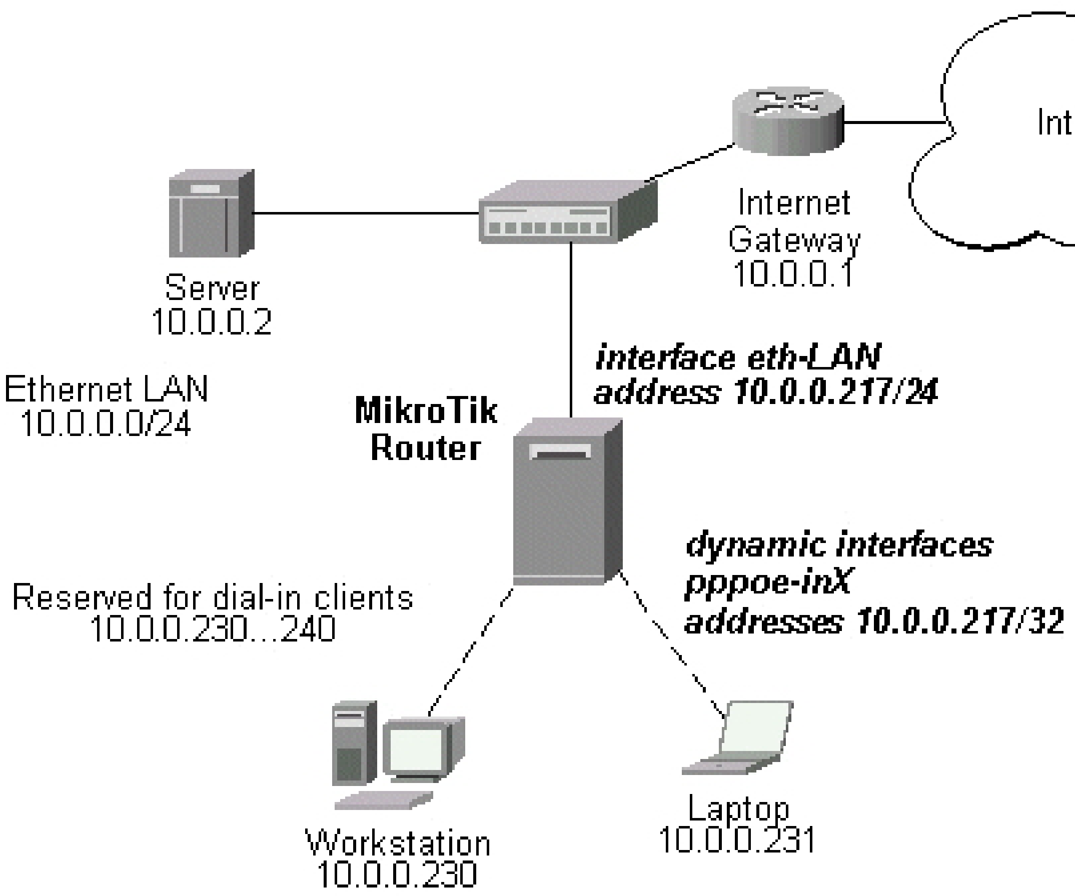
Suppose the host A needs to communicate to host C. To do this, it needs to know host's C MAC address. As shown on the diagram above, host A has /24 network mask. That makes host A to believe that it is directly connected to the whole 192.168.0.0/24 network. When a computer needs to communicate to another one on a directly connected network, it sends a broadcast ARP request. Therefore host A sends a broadcast ARP request for the host C MAC address.

Broadcast ARP requests are sent to the broadcast MAC address FF:FF:FF:FF:FF:FF. Since the ARP request is a broadcast, it will reach all hosts in the network A, including the router R1, but it will not reach host C, because routers do not forward broadcasts by default. A router with enabled proxy ARP knows that the host C is on another subnet and will reply with its own MAC adress. The router with enabled proxy ARP always answer with its own MAC address if it has a route to the destination.

This behaviour can be usefull, for example, if you want to assign dial-in (ppp, pppoe, pptp) clients IP addresses from the same address space as used on the connected LAN.

Example

Consider the following configuration:



The MikroTik Router setup is as follows:

```
admin@MikroTik] ip arp> /interface ethernet print
Flags: X - disabled, R - running
  #    NAME                 MTU   MAC-ADDRESS       ARP
  0  R eth-LAN              1500  00:50:08:00:00:F5 proxy-arp
[admin@MikroTik] ip arp> /interface print
Flags: X - disabled, D - dynamic, R - running
  #    NAME                 TYPE             MTU
  0    eth-LAN              ether            1500
  1    prism1               prism            1500
  2 D  pppoe-in25           pppoe-in
  3 D  pppoe-in26           pppoe-in
[admin@MikroTik] ip arp> /ip address print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS           NETWORK         BROADCAST       INTERFACE
  0   10.0.0.217/24     10.0.0.0        10.0.0.255      eth-LAN
  1 D 10.0.0.217/32     10.0.0.230      0.0.0.0         pppoe-in25
  2 D 10.0.0.217/32     10.0.0.231      0.0.0.0         pppoe-in26
[admin@MikroTik] ip arp> /ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DIS INTE...
 0 A S  0.0.0.0/0                          r 10.0.0.1        1   eth-LAN
 1 ADC  10.0.0.0/24        10.0.0.217                        0   eth-LAN
 2 ADC  10.0.0.230/32      10.0.0.217                        0   pppoe-in25
 3 ADC  10.0.0.231/32      10.0.0.217                        0   pppoe-in26
[admin@MikroTik] ip arp>
```

Troubleshooting

Description

- **Router shows that the IP address is invalid**
  Check whether the interface, the address is assigned to, is present, enabled and running.

Routes, Equal Cost Multipath Routing, Policy Routing

Document revision 2.4 (September 7, 2007, 8:37 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The following manual surveys the IP routes management, equal-cost multi-path (ECMP) routing technique, and policy-based routing.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip route***
Standards and Technologies: ***IP (RFC 791)***
Hardware usage: ***Not significant***

Description

MikroTik RouterOS has following types of routes:

- **dynamic routes** - automatically created routes for networks, which are directly accessed through an interface. They appear automatically, when adding a new IP address. Dynamic routes are also added by routing protocols.
- **static routes** - user-defined routes that specify the router which can forward traffic to the

specified destination network. They are useful for specifying the default gateway. The gateway for static routes may be checked (with either ARP or ICMP protocol) for reachability, so that different gateways with different priorities (costs) may be assigned for one destination network to provide failover.

ECMP (Equal Cost Multi-Path) Routing

This routing mechanism enables packet routing along multiple paths with equal cost and ensures load balancing. With ECMP routing, you can use more than one gateway for one destination network (this approach may also be configured to provide failover). With ECMP, a router potentially has several available next hops towards a given destination. A new gateway is chosen for each new source/destination IP pair. It means that, for example, one FTP connection will use only one link, but new connection to a different server will use another link. ECMP routing has another good feature - single connection packets do not get reordered and therefore do not kill TCP performance.

The ECMP routes can be created by routing protocols (RIP or OSPF), or by adding a static route with multiple gateways, separated by a comma (e.g., /ip route add gateway=192.168.0.1,192.168.1.1). The routing protocols may create multipath dynamic routes with equal cost automatically, if the cost of the interfaces is adjusted propery. For more information on using routing protocols, please read the corresponding Manual.

Policy-Based Routing

It is a routing approach where the next hop (gateway) for a packet is chosen, based on a policy, which is configured by the network administrator. In RouterOS the procedure the follwing:

- mark the desired packets, with a **routing-mark**
- choose a gateway for the marked packets

Note! In routing process, the router decides which route it will use to send out the packet. Afterwards, when the packet is masqueraded, its source address is taken from the **prefsrc** field.

Routes

Home menu level: ***/ip route***

Description

In this submenu you can configure Static, Equal Cost Multi-Path and Policy-Based Routing and see the routes.

Property Description

bgp-as-path (*text*) - manual value of BGP's as-path for outgoing route

bgp-atomic-aggregate (yes | no) - indication to receiver that it cannot "deaggregate" the prefix

bgp-communities (*multiple choice: integer*) - administrative policy marker, that can travel through different autonomous systems

- **internet** - communities value 0

bgp-local-pref (*integer*) - local preference value for a route

bgp-med (*integer*) - a BGP attribute, which provides a mechanism for BGP speakers to convey to an adjacent AS the optimal entry point into the local AS

bgp-origin (*incomplete* | *igp* | *egp*) - the origin of the route prefix

bgp-prepend (*integer*: 0..16) - number which indicates how many times to prepend AS_NAME to AS_PATH

check-gateway (*arp* | *ping*; default: **ping**) - which protocol to use for gateway reachability

distance (*integer*: 0..255) - administrative distance of the route. When forwarding a packet, the router will use the route with the lowest administrative distance and reachable gateway

dst-address (*IP addressnetmask*; default: **0.0.0.0/0**) - destination address and network mask, where netmask is number of bits which indicate network number. Used in static routing to specify the destination which can be reached, using a gateway

- **0.0.0.0/0** - any network

gateway (*IP address*) - gateway host, that can be reached directly through some of the interfaces. You can specify multiple gateways separated by a comma "," for ECMP routes

pref-src (*IP address*) - source IP address of packets, leaving router via this route

- **0.0.0.0** - pref-src is determined automatically

routing-mark (*name*) - a mark for packets, defined under /ip firewall mangle. Only those packets which have the according routing-mark, will be routed, using this gateway

scope (*integer*: 0..255) - a value which is used to recursively lookup the nexthop addresses. Nexthop is looked up only through routes that have scope <= target-scope of the nexthop

target-scope (*integer*: 0..255) - a value which is used to recursively lookup the next-hop addresses. Each nexthop address selects smallest value of target-scope from all routes that use this nexthop address. Nexthop is looked up only through routes that have scope <= target-scope of the nexthop

Notes

You can specify more than one or two gateways in the route. Moreover, you can repeat some routes in the list several times to do a kind of cost setting for gateways.

Example

To add two static routes to networks 10.1.12.0/24 and 0.0.0.0/0 (the default destination address) on a router with two interfaces and two IP addresses:

```
[admin@MikroTik] ip route> add dst-address=10.1.12.0/24 gateway=192.168.0.253
[admin@MikroTik] ip route> add gateway=10.5.8.1
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DIS INTE...
 0 A S  10.1.12.0/24       r 192.168.0.253            Local
 1 ADC  10.5.8.0/24                                   Public
 2 ADC  192.168.0.0/24                                Local
 3 A S  0.0.0.0/0          r 10.5.8.1                 Public
[admin@MikroTik] ip route>
```

Policy Rules

Home menu level: ***/ip route rule***

Property Description

action (*drop* | *unreachable* | *lookup*; default: **unreachable**) - action to be processed on packets matched by this rule:

- **drop** - silently drop packet
- **unreachable** - reply that destination host is unreachable
- **lookup** - lookup route in given routing table

dst-address (*IP addressnetmask*) - destination IP address/mask

interface (*name*; default: "") - interface through which the gateway can be reached

routing-mark (*name*; default: "") - mark of the packet to be mached by this rule. To add a routing mark, use '/ip firewall mangle' commands

src-address (*IP addressnetmask*) - source IP address/mask

table (*name*; default: "") - routing table, created by user

Notes

You can use policy routing even if you use masquerading on your private networks. The source address will be the same as it is in the local network. In previous versions of RouterOS the source address changed to **0.0.0.0**

It is impossible to recognize peer-to-peer traffic from the first packet. Only already established connections can be matched. That also means that in case source NAT is treating Peer-to-Peer traffic differently from the regular traffic, Peer-to-Peer programs will not work (general application is policy-routing redirecting regular traffic through one interface and Peer-to-Peer traffic - through another). A known workaround for this problem is to solve it from the other side: making not Peer-to-Peer traffic to go through another gateway, but all other useful traffic go through another gateway. In other words, to specify what protocols (HTTP, DNS, POP3, etc.) will go through the gateway A, leaving all the rest (so Peer-to-Peer traffic also) to use the gateway B (it is not important, which gateway is which; it is only important to keep Peer-to-Peer together with all traffic except the specified protocols)

Example

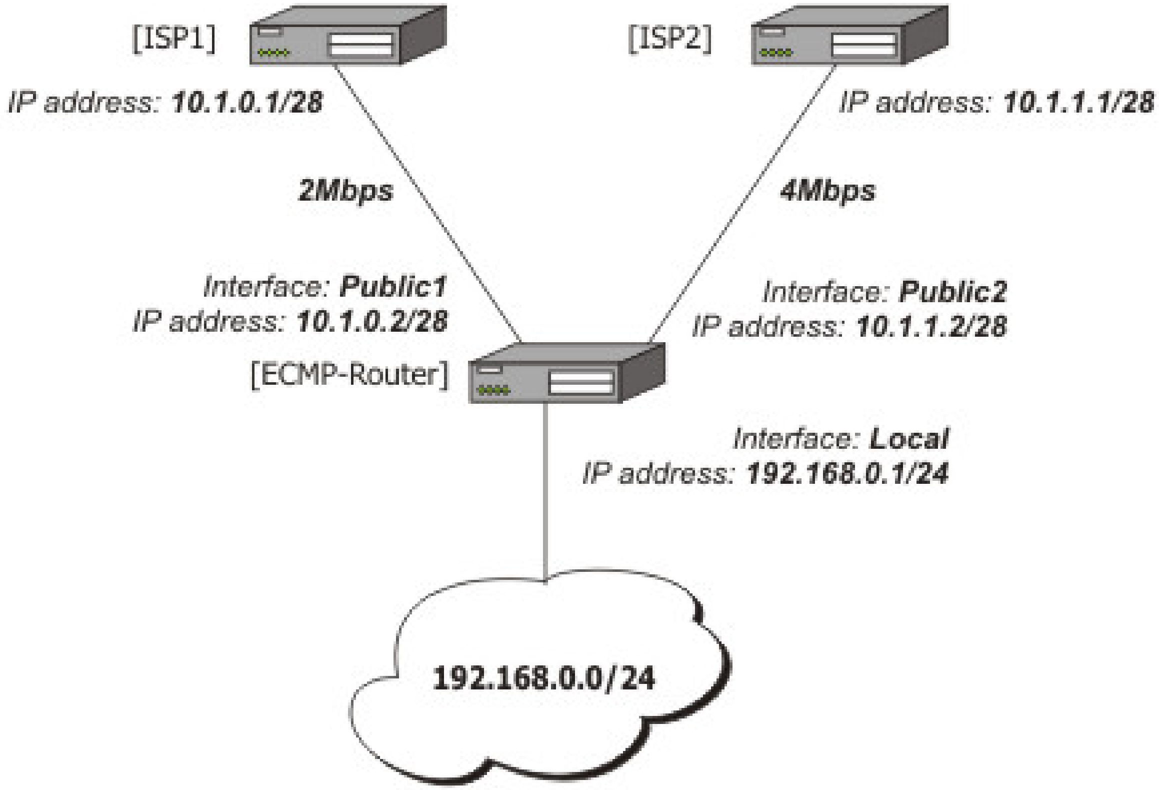
To add the rule specifying that all the packets from the 10.0.0.144 host should lookup the **mt** routing table:

```
[admin@MikroTik] ip firewall mangle add action=mark-routing new-routing-mark=mt \
\... chain=prerouting
[admin@MikroTik] ip route> add gateway=10.0.0.254 routing-mark=mt
[admin@MikroTik] ip route rule> add src-address=10.0.0.144/32 \
\... table=mt action=lookup
[admin@MikroTik] ip route rule> print
Flags: X - disabled, I - invalid
 0   src-address=192.168.0.144/32 action=lookup table=mt
[admin@MikroTik] ip route rule>
```

Application Examples

Static Equal Cost Multi-Path routing

Consider the following situation where we have to route packets from the network **192.168.0.0/24** to 2 gateways - **10.1.0.1** and **10.1.1.1**:



Note that the ISP1 gives us 2Mbps and ISP2 - 4Mbps so we want a traffic ratio 1:2 (1/3 of the source/destination IP pairs from **192.168.0.0/24** goes through ISP1, and 2/3 through ISP2).

IP addresses of the router:

```
[admin@ECMP-Router] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   192.168.0.254/24   192.168.0.0     192.168.0.255   Local
 1   10.1.0.2/28        10.1.0.0        10.1.0.15       Public1
 2   10.1.1.2/28        10.1.1.0        10.1.1.15       Public2
[admin@ECMP-Router] ip address>
```

Add the default routes - one for ISP1 and 2 for ISP2 so we can get the ratio 1:3:

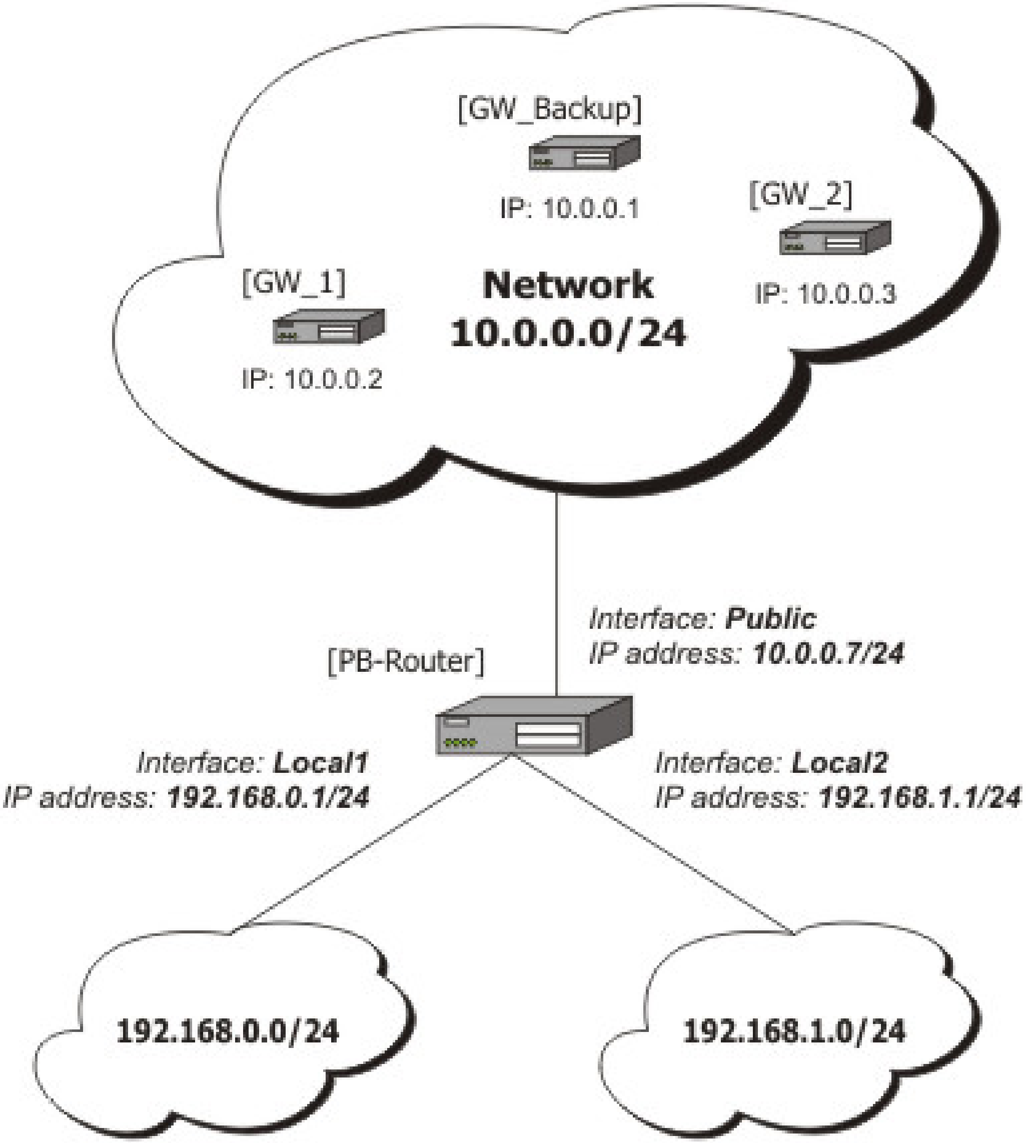
```
[admin@ECMP-Router] ip route> add gateway=10.1.0.1,10.1.1.1,10.1.1.1
[admin@ECMP-Router] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf
 #     DST-ADDRESS        G GATEWAY         DISTANCE INTERFACE
 0 ADC 10.1.0.0/28                                   Public1
 1 ADC 10.1.1.0/28                                   Public2
 2 ADC 192.168.0.0/24                                Local
 3 A S 0.0.0.0/0          r 10.1.0.1                 Public1
                          r 10.1.1.1                 Public2
                          r 10.1.1.1                 Public2
```

```
[admin@ECMP-Router] ip route>
```

Standard Policy-Based Routing with Failover

This example will show how to route packets, using an administrator defined policy. The policy for this setup is the following: route packets from the network **192.168.0.0/24**, using gateway 10.0.0.1, and packets from network **192.168.1.0/24**, using gateway 10.0.0.2. If GW_1 does not respond to pings, use GW_Backup for network 192.168.0.0/24, if GW_2 does not respond to pings, use GW_Backup also for network 192.168.1.0/24 instead of GW_2.

The setup:



Configuration of the IP addresses:

```
[admin@PB-Router] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
```

```
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   192.168.0.1/24     192.168.0.0     192.168.0.255   Local1
 1   192.168.1.1/24     192.168.1.0     192.168.1.255   Local2
 2   10.0.0.7/24        10.0.0.0        10.0.0.255      Public
[admin@PB-Router] ip address>
```

To achieve the described result, follow these configuration steps:

1. Mark packets from network 192.168.0.0/24 with a **new-routing-mark=net1**, and packets from network 192.168.1.0/24 with a **new-routing-mark=net2**:

```
[admin@PB-Router] ip firewall mangle> add src-address=192.168.0.0/24 \
\... action=mark-routing new-routing-mark=net1 chain=prerouting
[admin@PB-Router] ip firewall mangle> add src-address=192.168.1.0/24 \
\... action=mark-routing new-routing-mark=net2 chain=prerouting
[admin@PB-Router] ip firewall mangle> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=prerouting src-address=192.168.0.0/24 action=mark-routing
     new-routing-mark=net1

 1   chain=prerouting src-address=192.168.1.0/24 action=mark-routing
     new-routing-mark=net2
[admin@PB-Router] ip firewall mangle>
```

2. Route packets from network 192.168.0.0/24 to gateway GW_1 (10.0.0.2), packets from network 192.168.1.0/24 to gateway GW_2 (10.0.0.3), using the according packet marks. If GW_1 or GW_2 fails (does not reply to pings), route the respective packets to GW_Main (10.0.0.1):

```
[admin@PB-Router] ip route> add gateway=10.0.0.2 routing-mark=net1 \
\... check-gateway=ping
[admin@PB-Router] ip route> add gateway=10.0.0.3 routing-mark=net2 \
\... check-gateway=ping
[admin@PB-Router] ip route> add gateway=10.0.0.1
[admin@PB-Router] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf
 #     DST-ADDRESS        PREFSRC         G GATEWAY         DISTANCE INTERFACE
 0 ADC 10.0.0.0/24        10.0.0.7                                   Public
 1 ADC 192.168.0.0/24     192.168.0.1                                Local1
 2 ADC 192.168.1.0/24     192.168.1.1                                Local2
 3 A S 0.0.0.0/0                          r 10.0.0.2                 Public
 4 A S 0.0.0.0/0                          r 10.0.0.3                 Public
 5 A S 0.0.0.0/0                          r 10.0.0.1                 Public
[admin@PB-Router] ip route>
```

ARLAN 655 Wireless Client Card

Document revision 1.2 (September 7, 2007, 8:37 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS supports Arlan 655 Wireless Interface client cards. This card fits in the ISA expansion slot and provides transparent wireless communications to other network nodes.

Specifications

Packages required: ***arlan***
License required: ***level4***
Home menu level: ***/interface arlan***
Hardware usage: ***Not significant***

Installation

Example

To add the driver for Arlan 655 adapter, do the following:

```
[admin@MikroTik]> driver add name=arlan io=0xD000
[admin@MikroTik]> driver print
Flags: I - invalid, D - dynamic
  #   DRIVER                                IRQ IO       MEMORY   ISDN-PROTOCOL
  0 D RealTek 8139
  1   Arlan 655                                 0xD000

[admin@MikroTik] driver>
```

Wireless Interface Configuration

Home menu level: ***/interface arlan***

Description

The wireless card status can be obtained from the two LEDs: the **Status LED** and the **Activity LED**.

Status	Activity	Description
Amber	**Amber**	**ARLAN 655 is functional but nonvolatile memory is not configured**
Blinking Green	**Don't Care**	**ARLAN 655 not registered to an AP (ARLAN mode only)**
Green	**Off**	**Normal idle state**
Green	**Green Flash**	**Normal active state**
Red	**Amber**	**Hardware failure**
Red	**Red**	**Radio failure**

Property Description

add-name (*text*; default: **test**) - card name (optional). Must contain less than 16 characters.

arp (*disabled | enabled | proxy-arp | reply-only*; default: **enabled**) - Address Resolution Protocol setting

bitrate (*1000 | 2000 | 354 | 500*; default: **2000**) - data rate in Kbit/s

frequency (*2412 | 2427 | 2442 | 2457 | 2465*; default: **2412**) - channel frequency in MHz

mac-address (*MAC address*) - Media Access Control address

mtu (*integer*; default: **1500**) - Maximum Transmission Unit

name (*name*; default: **arlanN**) - assigned interface name

sid (*integer*; default: **0x13816788**) - System Identifier. Should be the same for all nodes on the radio network. Must be an even number with maximum length 31 character

tma-mode (*yes | no*; default: **no**) - Networking Registration Mode:

- **yes** - ARLAN
- **no** - NON ARLAN

Example

```
[admin@MikroTik] > interface print
Flags: X - disabled, D - dynamic, R - running
  #    NAME                                              TYPE             MTU
  0  R outer                                             ether            1500
  1  X arlan1                                            arlan            1500
[admin@MikroTik] interface> enable 1
[admin@MikroTik] > interface print
Flags: X - disabled, D - dynamic, R - running
```

```
  #    NAME                                                 TYPE             MTU
  0  R outer                                                ether            1500
  1  R arlan1                                               arlan            1500
```

More configuration and statistics parameters can be found under the **/interface arlan** menu:

```
[admin@MikroTik] interface arlan> print
Flags: X - disabled, R - running
  0  R name="arlan1" mtu=1500 mac-address=00:40:96:22:90:C8 arp=enabled
       frequency=2412 bitrate=2000 tma-mode=no card-name="test"
       sid=0x13816788

[admin@MikroTik] interface arlan>
```

You can monitor the status of the wireless interface:

```
[admin@MikroTik] interface arlan> monitor 0
      registered: no
    access-point: 00:00:00:00:00:00
        backbone: 00:00:00:00:00:00

[admin@MikroTik] interface arlan>
```

Suppose we want to configure the wireless interface to accomplish registration on the **AP** with a sid **0x03816788**. To do this, it is enough to change the argument value of **sid** to **0x03816788** and **tma-mode** to **yes**:

```
[admin@MikroTik] interface arlan> set 0 sid=0x03816788 tma-mode=yes
[admin@MikroTik] interface arlan> monitor 0
         registered: yes
    access-point: 00:40:88:23:91:F8
        backbone: 00:40:88:23:91:F9

[admin@MikroTik] interface arlan>
```

Troubleshooting

Description

Keep in mind, that not all combinations of I/O base addresses and IRQs may work on particular motherboard. It is recommended that you choose an IRQ not used in your system, and then try to find an acceptable I/O base address setting. As it has been observed, the IRQ 5 and I/O 0x300 or 0x180 will work in most cases.

- **The driver cannot be loaded because other device uses the requested IRQ.**
  Try to set different IRQ using the DIP switches.
- **The requested I/O base address cannot be used on your motherboard.**
  Try to change the I/O base address using the DIP switches.
- **The pc interface does not show up under the interfaces list**
  Obtain the required license for 2.4/5GHz Wireless Client feature.
- **The wireless card does not register to the Access Point**
  Check the cabling and antenna alignment.

Interface Bonding

Document revision 1.2 (September 10, 2007, 14:35 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Bonding is a technology that allows to aggregate multiple ethernet-like interfaces into a single virtual link, thus getting higher data rates and providing failover.

Quick Setup Guide

Let us assume that we have 2 NICs in each router (**Router1** and **Router2**) and want to get maximum data rate between 2 routers. To make this possible, follow these steps:

1. Make sure that you do not have IP addresses on interfaces which will be enslaved for bonding interface!
2. Add **bonding** interface on **Router1**:

```
[admin@Router1] interface bonding> add slaves=ether1,ether2
```

And on **Router2**:

```
[admin@Router2] interface bonding> add slaves=ether1,ether2
```

3. Add addresses to bonding interfaces:

```
[admin@Router1] ip address> add address=172.16.0.1/24 interface=bonding1
```

```
[admin@Router2] ip address> add address=172.16.0.2/24 interface=bonding1
```

4. Test the link from **Router1**:

```
[admin@Router1] interface bonding> /pi 172.16.0.2
172.16.0.2 ping timeout
172.16.0.2 ping timeout
172.16.0.2 ping timeout
172.16.0.2 64 byte ping: ttl=64 time=2 ms
172.16.0.2 64 byte ping: ttl=64 time=2 ms
```

Note that bonding interface needs a couple of seconds to get connectivity with its peer.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/interface bonding***
Standards and Technologies: ***None***
Hardware usage: ***Not significant***

Related Documents

- Linux Ethernet Bonding Driver mini-howto

Description

To provide a proper failover, you should specify **link-monitoring** parameter. It can be:

- MII (Media Independent Interface) type1 or type2 - Media Independent Interface is an abstract layer between the operating system and the NIC which detects whether the link is running (it performs also other functions, but in our case this is the most important).
- ARP - Address Resolution Protocol periodically (for **arp-interval** time) checks the link status.

link-monitoring is used to check whether the link is up or not.

Property Description

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; default: **enabled**) - Address Resolution Protocol for the interface

- **disabled** - the interface will not use ARP
- **enabled** - the interface will use ARP
- **proxy-arp** - the interface will use the ARP proxy feature
- **reply-only** - the interface will only reply to the requests originated to its own IP addresses. Neighbour MAC addresses will be resolved using /ip arp statically set table only

arp-interval (*time*; default: **00:00:00.100**) - time in milliseconds which defines how often to monitor ARP requests

arp-ip-targets (*IP address*; default: **""**) - IP target address which will be monitored if link-monitoring is set to arp. You can specify multiple IP addresses, separated by comma

down-delay (*time*; default: **00:00:00**) - if a link failure has been detected, bonding interface is disabled for down-delay time. Value should be a multiple of mii-interval

lacp-rate (*1sec* | *30secs*; default: **30secs**) - Link Aggregation Control Protocol rate specifies how often to exchange with LACPDUs between bonding peer. Used to determine whether link is up or other changes have occured in the network. LACP tries to adapt to these changes providing failover.

link-monitoring (*arp* | *mii-type1* | *mii-type2* | *none*; default: **none**) - method to use for monitoring the link (whether it is up or down)

- **arp** - uses Address Resolution Protocol to determine whether the remote interface is reachable

- **mii-type1** - uses Media Independent Interface type1 to determine link status. Link status determenation relies on the device driver. If bonding shows that the link status is up, when it should not be, then it means that this card don't support this possibility.
- **mii-type2** - uses MII type2 to determine link status (used if mii-type1 is not supported by the NIC)
- **none** - no method for link monitoring is used. If a link fails, it is not considered as down (but no traffic passes through it, thus).

mac-address (*read-only: MAC address*) - MAC address of the bonding interface

mii-interval (*time*; default: **00:00:00.100**) - how often to monitor the link for failures (parameter used only if link-monitoring is mii-type1 or mii-type2)

mode (*802.3ad* | *active-backup* | *balance-alb* | *balance-rr* | *balance-tlb* | *balance-xor* | *broadcast*; default: **balance-rr**) - interface bonding mode. Can be one of:

- **802.3ad** - IEEE 802.3ad dynamic link aggregation. In this mode, the interfaces are aggregated in a group where each slave shares the same speed. If you use a switch between 2 bonding routers, be sure that this switch supports IEEE 802.3ad standard. Provides fault tolerance and load balancing.
- **active-backup** - provides link backup. Only one slave can be active at a time. Another slave becomes active only, if first one fails.
- **balance-alb** - adaptive load balancing. It includes balance-tlb and received traffic is also balanced. Device driver should support for setting the mac address, then it is active. Otherwise balance-alb doesn't work. No special switch is required.
- **balance-rr** - round-robin load balancing. Slaves in bonding interface will transmit and receive data in sequential order. Provides load balancing and fault tolerance.
- **balance-tlb** - Outgoing traffic is distributed according to the current load on each slave. Incoming traffic is received by the current slave. If receiving slave fails, then another slave takes the MAC address of the failed slave. Doesn't require any special switch support.
- **balance-xor** - Use XOR policy for transmit. Provides only failover (in very good quality), but not load balancing, yet.
- **broadcast** - Broadcasts the same data on all interfaces at once. This provides fault tolerance but slows down traffic throughput on some slow machines.

mtu (*integer*: 68..1500; default: **1500**) - Maximum Transmit Unit in bytes

name (*name*) - descriptive name of bonding interface

primary (*name*; default: **none**) - Interface is used as primary output media. If primary interface fails, only then others slaves will be used. This value works only with mode=active-backup

slaves (*name*) - at least two ethernet-like interfaces separated by a comma, which will be used for bonding

up-delay (*time*; default: **00:00:00**) - if a link has been brought up, bonding interface is disabled for up-delay time and after this time it is enabled. Value should be a multiple of mii-interval

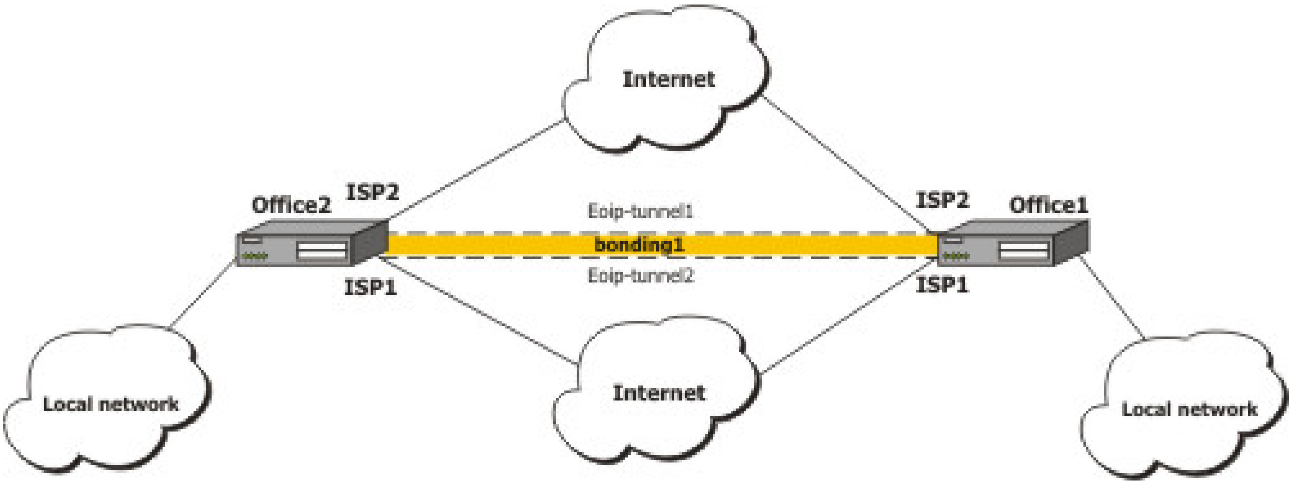
Notes

Link failure detection and failover is working significantly better with expensive network cards, for example, made by Intel, then with more cheap ones. For example, on Intel cards failover is taking place in less than a second after link loss, while on some other cards, it may require up to 20 seconds. Also, the Active load balancing (mode=**balance-alb**) does not work on some cheap cards.

Application Examples

Bonding two Eoip tunnels

Assume you need to configure the MikroTik router for the following network setup, where you have two offices with 2 ISP for each. You want combine links for getting double speed and provide failover:



We are assuming that connections to Internet through two ISP are configured for both routers.

- Configuration on routers
  - on **Office1**

```
[admin@office1] > /interface print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                                        TYPE             MTU
 0  R isp1                                        ether            1500
 1  R isp2                                        ether            1500

[admin@office1] > /ip address print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   1.1.1.1/24         1.1.1.0         1.1.1.255       isp2
 1   10.1.0.111/24      10.1.0.0        10.1.0.255      isp1
```

  - on **Office2**

```
[admin@office2] interface> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                                        TYPE             MTU
 0  R isp2                                        ether            1500
 1  R isp1                                        ether            1500
[admin@office2] interface> /ip add print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   2.2.2.1/24         2.2.2.0         2.2.2.255       isp2
 1   10.1.0.112/24      10.1.0.0        10.1.0.255      isp1
```

- Eoip tunnel confguration

- for **Office1** through ISP1

```
[admin@office1] > interface eoip add remote-address=10.1.0.112 tunnel-id=2
\... mac-address=FE:FD:00:00:00:04
[admin@office1] > interface eoip print
Flags: X - disabled, R - running
0  R name="eoip-tunnel2" mtu=1500 mac-address==FE:FD:00:00:00:04 arp=enabled
\... remote-address=10.1.0.112 tunnel-id=2
```

- for **Office2** through ISP1

```
[admin@office2] > interface eoip add remote-address=10.1.0.111 tunnel-id=2
\... mac-address=FE:FD:00:00:00:02
[admin@office2] > interface eoip print
Flags: X - disabled, R - running
0  R name="eoip-tunnel2" mtu=1500 mac-address=FE:FD:00:00:00:02 arp=enabled
\... remote-address=10.1.0.111 tunnel-id=2
```

- for **Office1**through ISP2

```
[admin@office1] > interface eoip add remote-address=2.2.2.1 tunnel-id=1
\... mac-address=FE:FD:00:00:00:03
[admin@office1] interface eoip> print
Flags: X - disabled, R - running
 0  R name="eoip-tunnel1" mtu=1500 mac-address=FE:FD:00:00:00:03 arp=enabled
      remote-address=2.2.2.1 tunnel-id=1

 1  R name="eoip-tunnel2" mtu=1500 mac-address=FE:FD:00:00:00:04 arp=enabled
      remote-address=10.1.0.112 tunnel-id=2
```

- for **Office2**through ISP2

```
[admin@office2] > interface eoip add remote-address=1.1.1.1 tunnel-id=1
\... mac-address=FE:FD:00:00:00:01
[admin@office2] interface eoip> print
Flags: X - disabled, R - running
 0  R name="eoip-tunnel1" mtu=1500 mac-address=FE:FD:00:00:00:01 arp=enabled
      remote-address=1.1.1.1 tunnel-id=1

 1  R name="eoip-tunnel2" mtu=1500 mac-address=FE:FD:00:00:00:02 arp=enabled
      remote-address=10.1.0.111 tunnel-id=2
```

- Bonding configuration
  - for **Office1**

```
[admin@office1] interface bonding> add slaves=eoip-tunnel1,eoip-tunnel2
[admin@office1] interface bonding> print
Flags: X - disabled, R - running
 0  R name="bonding1" mtu=1500 mac-address=00:0C:42:03:20:E7 arp=enabled
      slaves=eoip-tunnel1,eoip-tunnel2 mode=balance-rr primary=none
      link-monitoring=none arp-interval=00:00:00.100 arp-ip-targets=""
      mii-interval=00:00:00.100 down-delay=00:00:00 up-delay=00:00:00
      lacp-rate=30secs
[admin@office1] ip address> add address=3.3.3.1/24 interface=bonding1
[admin@office1] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   1.1.1.1/24         1.1.1.0         1.1.1.255       isp2
 1   10.1.0.111/24      10.1.0.0        10.1.0.255      isp1
 2   3.3.3.1/24         3.3.3.0         3.3.3.255       bonding1
```

- for **Office2**

```
[admin@office2] interface bonding> add slaves=eoip-tunnel1,eoip-tunnel2
[admin@office2] interface bonding> print
Flags: X - disabled, R - running
 0  R name="bonding1" mtu=1500 mac-address=00:0C:42:03:20:E7 arp=enabled
      slaves=eoip-tunnel1,eoip-tunnel2 mode=balance-rr primary=none
      link-monitoring=none arp-interval=00:00:00.100 arp-ip-targets=""
      mii-interval=00:00:00.100 down-delay=00:00:00 up-delay=00:00:00
      lacp-rate=30secs
[admin@office2] ip address> add address=3.3.3.2/24 interface=bonding1
[admin@office2] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   2.2.2.1/24         2.2.2.0         2.2.2.255       isp2
 1   10.1.0.112/24      10.1.0.0        10.1.0.255      isp1
 2   3.3.3.2/24         3.3.3.0         3.3.3.255       bonding1
[admin@office2] ip address> /ping 3.3.3.1
3.3.3.1 64 byte ping: ttl=64 time=2 ms
3.3.3.1 64 byte ping: ttl=64 time=2 ms
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 2/2.0/2 ms
```

CISCO/Aironet 2.4GHz 11Mbps Wireless Interface

Document revision 1.3 (February 6, 2008, 2:56 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS supports the following CISCO/Aironet 2.4GHz Wireless ISA/PCI/PC Adapter hardware:

- Aironet ISA/PCI/PC4800 2.4GHz DS 11Mbps Wireless LAN Adapters (100mW)
- Aironet ISA/PCI/PC4500 2.4GHz DS 2Mbps Wireless LAN Adapters (100mW)
- CISCO AIR-PCI340 2.4GHz DS 11Mbps Wireless LAN Adapters (30mW)
- CISCO AIR-PCI/PC350/352 2.4GHz DS 11Mbps Wireless LAN Adapters (100mW)

Specifications

Packages required: ***wireless***
License required: ***level4***
Home menu level: ***/interface pc***
Standards and Technologies: ***IEEE802.11b***
Hardware usage: ***Not significant***

Additional Documents

- CISCO Aironet 350 Series

For more information about the CISCO/Aironet PCI/ISA adapter hardware please see the relevant User's

Guides and Technical Reference Manuals in PDF format:

- 710-003638a0.pdffor PCI/ISA 4800 and 4500 series adapters
- 710-004239B0.pdffor PC 4800 and 4500 series adapters

Documentation about CISCO/Aironet Wireless Bridges and Access Points can be found in archives:

- AP48MAN.exe for AP4800 Wireless Access Point
- BR50MAN.exe for BR500 Wireless Bridge

Wireless Interface Configuration

Home menu level: ***/interface pc***

Description

CISCO/Aironet 2.4GHz card is an interface for wireless networks operating in IEEE 802.11b standard. If the wireless interface card is not registered to an AP, the green status led is blinking fast. If the wireless interface card is registered to an AP, the green status led is blinking slow. To set the wireless interface for working with an access point (register to the AP), typically you should set the following parameters:

- The **service set identifier**. It should match the ssid of the AP. Can be blank, if you want the wireless interface card to register to an AP with any ssid. The ssid will be received from the AP, if the AP is broadcasting its ssid.
- The data-rate of the card should match one of the supported data rates of the AP. Data rate 'auto' should work in most cases.

Loading the Driver for the Wireless Adapter

PCI and PC (PCMCIA) cards do not require a 'manual' driver loading, since they are recognized automatically by the system and the driver is loaded at the system startup.

The ISA card requires the driver to be loaded by issuing the following command:

```
[admin@MikroTik]> driver add name=pc-isa io=0x180
[admin@MikroTik]> driver print
Flags: I - invalid, D - dynamic
 #   DRIVER                            IRQ IO         MEMORY     ISDN-PROTOCOL
 0 D PCI NE2000
 1   Aironet ISAxx00                       0x180
[admin@MikroTik] driver>
```

There can be several reasons for a failure to load the driver:

- **The driver cannot be loaded because other device uses the requested IRQ.**
  Try to set different IRQ using the DIP switches.
- **The requested I/O base address cannot be used on your motherboard**
  Try to change the I/O base address using the DIP switches

Property Description

ap1 (*MAC address*) - forces association to the specified access point

ap2 (*MAC address*) - forces association to the specified access point

ap3 (*MAC address*) - forces association to the specified access point

ap4 (*MAC address*) - forces association to the specified access point

arp (*disabled | enabled | proxy-arp | reply-only*; default: **enabled**) - Address Resolution Protocol

beacon-period (*integer*: 20..976; default: **100**) - Specifies beaconing period (applicable to ad-hoc mode only)

card-type (*read-only: text*) - your CISCO/Aironet adapter model and type

client-name (*text*; default: **""**) - client name

data-rate (*1Mbit/s | 2Mbit/s | 5.5Mbit/s | 11Mbit/s | auto*; default: **1Mbit/s**) - data rate in Mbit/s

fragmentation-threshold (*integer*: 256..2312; default: **2312**) - this threshold controls the packet size at which outgoing packets will be split into multiple fragments. If a single fragment transmit error occurs, only that fragment will have to be retransmitted instead of the whole packet. Use a low setting in areas with poor communication or with a great deal of radio interference

frequency - Channel Frequency in MHz (applicable to ad-hoc mode only)

join-net (*time*; default: **10**) - an amount of time,during which the interface operating in ad-hoc mode will try to connect to an existing network rather than create a new one

- **0** - do not create own network

long-retry-limit (*integer*: 0..128; default: **16**) - specifies the number of times an unfragmented packet is retried before it is dropped

mode (*infrastructure | ad-hoc*; default: **infrastructure**) - operation mode of the card

modulation (*cck | default | mbok*; default: **cck**) - modulation mode

- **cck** - Complementary Code Keying
- **mbok** - M-ary Bi-Orthogonal Keying

mtu (*integer*: 256..2048; default: **1500**) - Maximum Transmission Unit

name (*name*) - descriptive interface name

rts-threshold (*integer*: 0..2312; default: **2312**) - determines the packet size at which the interface issues a request to send (RTS) before sending the packet. A low value can be useful in areas where many clients are associating with the access point or bridge, or in areas where the clients are far apart and can detect only the access point or bridge and not each other

rx-antenna (*both | default | left | right*; default: **both**) - receive antennas

short-retry-limit (*integer*: 0..128; default: **16**) - specifies the number of times a fragmented packet is retried before it is dropped

ssid1 (*text*; default: **tsunami**) - establishes the adapter's service set identifier This value must match the SSID of the system in order to operate in infrastructure mode

ssid2 (*text*; default: **""**) - service set identifier 2

ssid3 (*text*; default: **""**) - service set identifier 3

tx-antenna (*both | default | left | right*; default: **both**) - transmit antennas

tx-power (*1 | 5 | 20 | 50 | 100*; default: **100**) - transmit power in mW

world-mode (yes | no; default: **no**) - if set, client adapter automatically inherit channel configuration properties directly from the access point to which they associate. This feature enables a user to use a client adapter around the world while still maintaining regulatory compliance

Example

Interface informational printouts

```
[admin@MikroTik] > interface print
Flags: X - disabled, D - dynamic, R - running
  #    NAME                 TYPE             MTU
  0  R ether1               ether            1500
  1  X ether2               ether            1500
  2  X pc1                  pc               1500
[admin@MikroTik] interface> set 2 name aironet
[admin@MikroTik] interface> enable aironet
[admin@MikroTik] > interface print
Flags: X - disabled, D - dynamic, R - running
  #    NAME                 TYPE             MTU
  0  R ether1               ether            1500
  1  X ether2               ether            1500
  2  R aironet              pc               1500
[admin@MikroTik] > interface pc
[admin@MikroTik] interface pc> print
Flags: X - disabled, R - running
  0  R name="aironet" mtu=1500 mac-address=00:40:96:29:2F:80 arp=enabled
       client-name="" ssid1="tsunami" ssid2="" ssid3="" mode=infrastructure
       data-rate=1Mbit/s frequency=2437MHz modulation=cck tx-power=100
       ap1=00:00:00:00:00:00 ap2=00:00:00:00:00:00 ap3=00:00:00:00:00:00
       ap4=00:00:00:00:00:00 rx-antenna=right tx-antenna=right beacon-period=100
       long-retry-limit=16 short-retry-limit=16 rts-threshold=2312
       fragmentation-threshold=2312 join-net=10s card-type=PC4800A 3.65

[admin@MikroTik] interface pc>
```

Interface status monitoring

```
[admin@MikroTik] interface pc> monitor 0
         synchronized: no
           associated: no
         error-number: 0

[admin@MikroTik] interface pc>
```

Example

Suppose we want to configure the wireless interface to accomplish registration on the AP with a **ssid** 'mt'.

We need to change the value of ssid property to the corresponding value.

To view the results, we can use **monitor** feature.

```
[admin@MikroTik] interface pc> set 0 ssid1 mt
[admin@MikroTik] interface pc> monitor 0
         synchronized: yes
           associated: yes
            frequency: 2412MHz
            data-rate: 11Mbit/s
                 ssid: "mt"
         access-point: 00:02:6F:01:5D:FE
    access-point-name: ""
       signal-quality: 132
      signal-strength: -82
         error-number: 0
[admin@MikroTik] interface pc>
```

Troubleshooting

Description

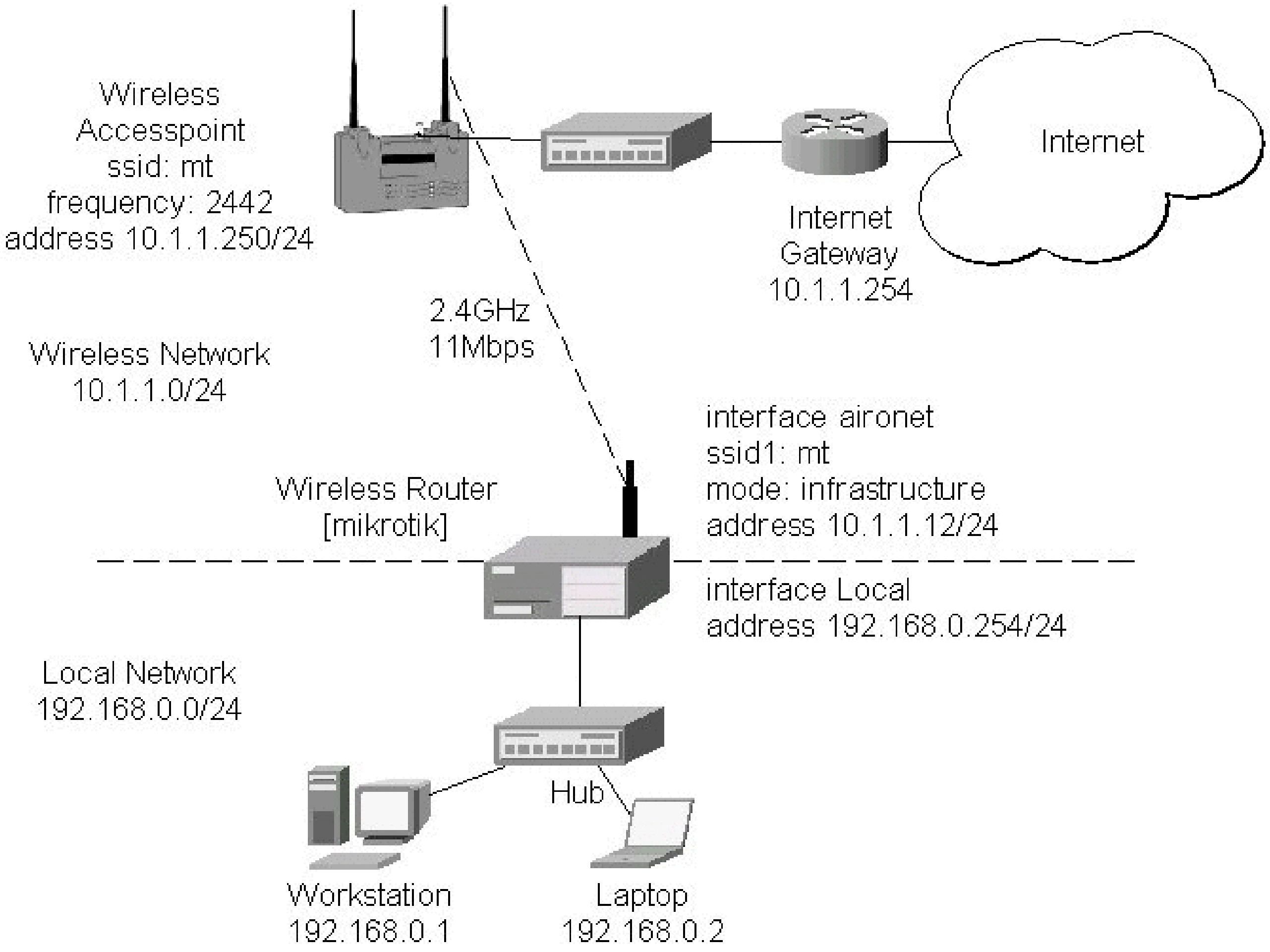
Keep in mind, that not all combinations of I/O base addresses and IRQs may work on particular motherboard. It is recommended that you choose an IRQ not used in your system, and then try to find an acceptable I/O base address setting. As it has been observed, the IRQ 5 and I/O 0x300 or 0x180 will work in most cases.

- **The driver cannot be loaded because other device uses the requested IRQ.**
  Try to set different IRQ using the DIP switches.
- **The requested I/O base address cannot be used on your motherboard.**
  Try to change the I/O base address using the DIP switches.
- **The pc interface does not show up under the interfaces list**
  Obtain the required license for 2.4/5GHz Wireless Client feature.
- **The wireless card does not register to the Access Point**
  Check the cabling and antenna alignment.

Application Examples

Point-to-Multipoint Wireless LAN

Let us consider the following network setup with CISCO/Aironet Wireless Access Point as a base station and MikroTik Wireless Router as a client:



The access point is connected to the wired network's HUB and has IP address from the network 10.1.1.0/24.

The minimum configuration required for the AP is:

1. Setting the Service Set Identifier (up to 32 alphanumeric characters). In our case we use ssid "mt".
2. Setting the allowed data rates at 1-11Mbps, and the basic rate at 1Mbps.
3. Choosing the frequency, in our case we use 2442MHz.
4. (For CISCO/Aironet Bridges only) Set Configuration/Radio/Extended/Bridge/mode=access_point. If you leave it to 'bridge_only', it wont register clients.
5. Setting the identity parameters Configuration/Ident: Inaddr, Inmask, and Gateway. These are required if you want to access the AP remotely using telnet or http.

The IP addresses assigned to the wireless interface should be from the network 10.1.1.0/24:

```
[admin@MikroTik] ip address> add address 10.1.1.12/24 interface aironet
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.1.1.12/24       10.1.1.0        10.1.1.255      aironet
  1   192.168.0.254/24   192.168.0.0     192.168.0.255   Local
[admin@MikroTik] ip address>
```

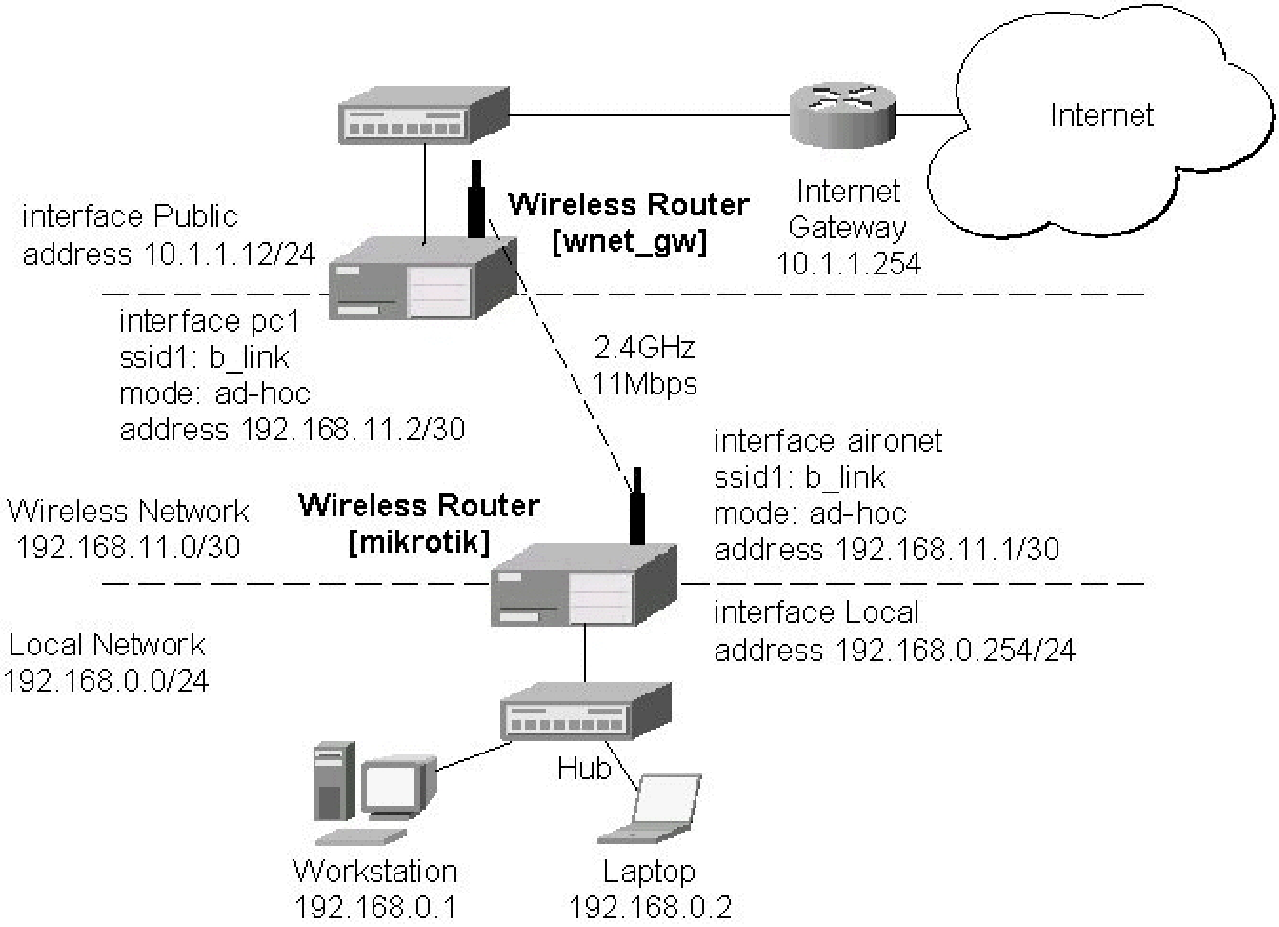
The default route should be set to the gateway router 10.1.1.254 (! not the AP 10.1.1.250 !):

```
[admin@MikroTik] ip route> add gateway=10.1.1.254
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DISTANCE INTER...
 0 A S 0.0.0.0/0                           r 10.1.1.254      1        aironet
 1 ADC 192.168.0.0/24      192.168.0.254   r 0.0.0.0         0        Local
 2 ADC 10.1.1.0/24         10.1.1.12       r 0.0.0.0         0        aironet
[admin@MikroTik] ip route>
```

Point-to-Point Wireless LAN

Point-to-Point links provide a convenient way to connect a pair of clients on a short distance.

Let us consider the following point-to-point wireless network setup with two MikroTik wireless routers:



To establish a point-to-point link, the configuration of the wireless interface should be as follows:

- A unique Service Set Identifier should be chosen for both ends, say "mt"
- A channel frequency should be selected for the link, say 2412MHz
- The operation mode should be set to ad-hoc
- One of the units (slave) should have wireless interface property join-net set to 0s (never create a network), the other unit (master) should be set to 1s or whatever, say 10s. This will enable the master

unit to create a network and register the slave unit to it.

The following command should be issued to change the settings for the pc interface of the master unit:

```
[admin@MikroTik] interface pc> set 0 mode=ad-hoc ssid1=mt frequency=2442MHz \
\... bitrate=auto
[admin@MikroTik] interface pc>
```

For 10 seconds (this is set by the property **join-net**) the wireless card will look for a network to join. The status of the card is not synchronized, and the green status light is blinking fast. If the card cannot find a network, it creates its own network. The status of the card becomes synchronized, and the green status led becomes solid.

The monitor command shows the new status and the MAC address generated:

```
[admin@MikroTik] interface pc> monitor 0
         synchronized: yes
           associated: yes
            frequency: 2442MHz
            data-rate: 11Mbit/s
                 ssid: "mt"
         access-point: 2E:00:B8:01:98:01
    access-point-name: ""
       signal-quality: 35
      signal-strength: -62
         error-number: 0
[admin@MikroTik] interface pc>
```

The other router of the point-to-point link requires the operation mode set to **ad-hoc**, the System Service Identifier set to 'mt', and the channel frequency set to 2412MHz. If the cards are able to establish RF connection, the status of the card should become synchronized, and the green status led should become solid immediately after entering the command:

```
[admin@wnet_gw] interface pc> set 0 mode=ad-hoc ssid1=b_link frequency=2412MHz \
\... bitrate=auto
[admin@wnet_gw] interface pc> monitor 0
         synchronized: yes
           associated: no
            frequency: 2442MHz
            data-rate: 11Mbit/s
                 ssid: "b_link"
         access-point: 2E:00:B8:01:98:01
    access-point-name: ""
       signal-quality: 131
      signal-strength: -83
         error-number: 0

[admin@wnet_gw] interface pc>
```

As we see, the MAC address under the **access-point** property is the same as on the first router.

If desired, IP addresses can be assigned to the wireless interfaces of the pint-to-point linked routers using a smaller subnet, say 30-bit one:

```
[admin@MikroTik] ip address> add address 192.168.11.1/30 interface aironet
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   192.168.11.1/30    192.168.11.0    192.168.11.3    aironet
  1   192.168.0.254/24   192.168.0.0     192.168.0.255   Local
[admin@MikroTik] ip address>
```

The second router will have address 192.168.11.2. The network connectivity can be tested by using ping or bandwidth test:

```
[admin@wnet_gw] ip address> add address 192.168.11.2/30 interface aironet
[admin@wnet_gw] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   192.168.11.2/30    192.168.11.0    192.168.11.3    aironet
  1   10.1.1.12/24       10.1.1.0        10.1.1.255      Public
[admin@wnet_gw] ip address> /ping 192.168.11.1
192.168.11.1 64 byte ping: ttl=255 time=3 ms
192.168.11.1 64 byte ping: ttl=255 time=1 ms
192.168.11.1 64 byte ping: ttl=255 time=1 ms
4 packets transmitted, 3 packets received, 25% packet loss
round-trip min/avg/max = 1/1.5/3 ms
[admin@wnet_gw] interface pc> /tool bandwidth-test 192.168.11.1 protocol tcp
                  status: running
              rx-current: 4.61Mbps
    rx-10-second-average: 4.25Mbps
        rx-total-average: 4.27Mbps

[admin@wnet_gw] interface pc> /tool bandwidth-test 192.168.11.1 protocol udp size 1500
                  status: running
              rx-current: 5.64Mbps
    rx-10-second-average: 5.32Mbps
        rx-total-average: 4.87Mbps

[admin@wnet_gw] interface pc>
```

Cyclades PC300 PCI Adapters

Document revision 1.3 (February 6, 2008, 2:58 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS supports the following Cyclades PC300 Adapter hardware:

- RSV/V.35 (RSV models) with 1 or 2 RS-232/V.35 interfaces on standard DB25/M.34 connector, 5Mbps, internal or external clock
- T1/E1 (TE models) with 1 or 2 T1/E1/G.703 interfaces on standard RJ48C connector, Full/Fractional, internal or external clock
- X.21 (X21 models) with 1 or 2 X.21 on standard DB-15 connector, 8Mbps, internal or external clock

Specifications

Packages required: ***synchronous***
License required: ***level4***
Home menu level: ***/interface cyclades***
Standards and Technologies: ***X.21, X.35, T1/E1/G.703, Frame Relay, PPP, Cisco-HDLC***
Hardware usage: ***Not significant***

Synchronous Interface Configuration

Home menu level: ***/interface cyclades***

Description

You can install up to four Cyclades PC300 PCI Adapters in one PC box, if you have so many adapter slots and IRQs available.

The Cyclades PC300/RSV Synchronous PCI Adapter comes with a V.35 cable. This cable should work for all standard modems, which have V.35 connections. For synchronous modems, which have a DB-25 connection, you should use a standard DB-25 cable.

Connect a communication device, e.g., a baseband modem, to the V.35 port and turn it on. The MikroTik driver for the Cyclades Synchronous PCI Adapter allows you to unplug the V.35 cable from one modem and plug it into another modem with a different clock speed, and you do not need to restart the interface or router.

Property Description

chdlc-keepalive (*time*; default: **10s**) - Cisco-HDLC keepalive interval in seconds

clock-rate (*integer*; default: **64000**) - internal clock rate in bps

clock-source (*internal* | *external* | *tx-internal*; default: **external**) - source clock

frame-relay-dce (*yes* | *no*; default: **no**) - specifies whether the device operates in Data Communication Equipment mode. The value yes is suitable only for T1 models

frame-relay-lmi-type (*ansi* | *ccitt*; default: **ansi**) - Frame Relay Line Management Interface Protocol type

framing mode (*CRC4* | *D4* | *ESF* | *Non-CRC4* | *Unframed*; default: **ESF**) - for T1/E1 channels only. The frame mode:

- **CRC4** - Cyclic Redundancy Check 4-bit (E1 Signaling, Europe)
- **D4** - Fourth Generation Channel Bank (48 Voice Channels on 2 T-1s or 1 T-1c)
- **ESF** - Extended Superframe Format
- **Non-CRC4** - plain Cyclic Redundancy Check
- **Unframed** - do not check frame integrity

line-build-out (*0dB* | *7.5dB* | *15dB* | *22.5dB*; default: **0**) - for T1 channels only. Line Build Out Signal Level.

line-code (*AMI* | *B8ZS* | *HDB3* | *NRZ*; default: **B8ZS**) - for T1/E1 channels only. Line modulation method:

- **AMI** - Alternate Mark Inversion
- **B8ZS** - Binary 8-Zero Substitution
- **HDB3** - High Density Bipolar 3 Code (ITU-T)
- **NRZ** - Non-Return-To-Zero

line-protocol (*cisco-hdlc* | *frame-relay* | *sync-ppp*; default: **sync-ppp**) - line protocol

media-type (*E1* | *T1* | *V24* | *V35* | *X21*; default: **V35**) - the hardware media used for this interface

mtu (*integer*; default: **1500**) - Maximum Transmission Unit for the interface

name (*name*; default: **cycladesN**) - descriptive interface name

rx-sensitivity (*long-haul* | *short-haul*; default: **short-haul**) - for T1/E1 channels only. Numbers of active channels (up to 32 for E1 and up to 24 for T1)

Troubleshooting

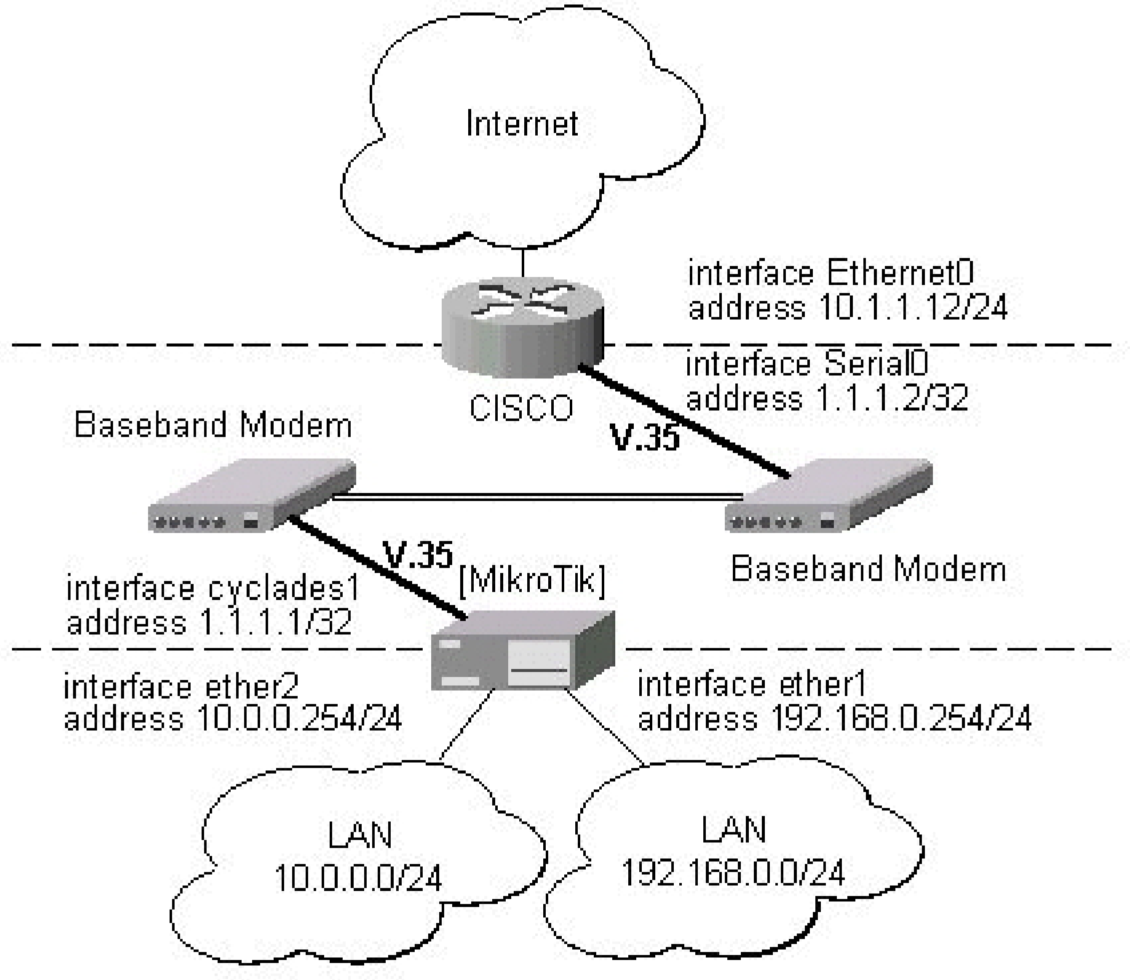
Description

- **The cyclades interface does not show up under the interfaces list**
  Obtain the required license for synchronous feature
- **The synchronous link does not work**
  Check the V.35 cabling and the line between the modems. Read the modem manual

RSV/V.35 Synchronous Link Applications

Example

Let us consider the following network setup with MikroTik Router connected to a leased line with baseband modems and a CISCO router at the other end:



The driver for the Cyclades PC300/RSV Synchronous PCI Adapter should load automatically. The interface should be enabled according to the instructions given above. The **IP addresses** assigned to the cyclades interface should be as follows:

```
[admin@MikroTik] ip address> add address=1.1.1.1/32 interface=cyclades1
```

```
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.219/24      10.0.0.0        10.0.0.255      ether1
  1   1.1.1.1/32         1.1.1.1         1.1.1.1         cyclades1
  2   192.168.0.254/24   192.168.0.0     192.168.0.255   ether2
[admin@MikroTik] ip address> /ping 1.1.1.2
1.1.1.2 64 byte ping: ttl=255 time=12 ms
1.1.1.2 64 byte ping: ttl=255 time=8 ms
1.1.1.2 64 byte ping: ttl=255 time=7 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 7/9.0/12 ms
[admin@MikroTik] ip address> /tool flood-ping 1.1.1.2 size=1500 count=50
        sent: 50
    received: 50
     min-rtt: 1
     avg-rtt: 1
     max-rtt: 9

[admin@MikroTik] ip address>
```

Note that for the point-to-point link the network mask is set to 32 bits, the argument **network** is set to the **IP address** of the other end, and the broadcast address is set to 255.255.255.255. The default route should be set to gateway router 1.1.1.2:

```
[admin@MikroTik] ip route> add gateway=1.1.1.2 interface=cyclades1
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DISTANCE INTER...
 0 A S  0.0.0.0/0                          r 1.1.1.2         1        cyclades1
 1 ADC  10.0.0.0/24        10.0.0.219      r                 0        ether1
 2 ADC  192.168.0.0/24     192.168.0.254   r                 0        ether2
 3 ADC  1.1.1.2/32         1.1.1.1         r                 0        cyclades1
[admin@MikroTik] ip route>
```

The configuration of the CISCO router at the other end (part of the configuration) is:

```
CISCO#show running-config
Building configuration...

Current configuration:
...
!
interface Ethernet0
 description connected to EthernetLAN
 ip address 10.1.1.12 255.255.255.0
!
interface Serial0
 description connected to MikroTik
 ip address 1.1.1.2 255.255.255.252
 serial restart-delay 1
!
ip classless
ip route 0.0.0.0 0.0.0.0 10.1.1.254
!
...
end

CISCO#

Send ping packets to the MikroTik router:


CISCO#ping 1.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/32/40 ms
```

```
CISCO#
```

Driver Management

Document revision 2.2 (February 11, 2008, 4:14 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Device drivers represent the software interface part of installed network devices. Some drivers are included in the system software package and some in additional feature packages.

For complete list of supported devices and respective device driver names please consult the 'Related Documents' section.

The device drivers for PCI, miniPCI, PC (PCMCIA) and CardBus cards are loaded automatically. Other network interface cards (most ISA and PCI ISDN cards) require the device drivers to be loaded manually using the **/driver add command.**

Users cannot add their own device drivers, only drivers included in the Mikrotik RouterOS software packages can be used. If you need a support for a device, which hasn't a driver yet, you are welcome to suggest it at suggestion page on our web site.

Home menu level: ***/driver***
Standards and Technologies: ***PCI, ISA, PCMCIA, miniPCI, CardBus***
Hardware usage: ***Not significant***

Loading Device Drivers

Home menu level: ***/driver***

Description

In order to use network interface card which has a driver that is not loaded automatically, *exempli gratia* NE2000 compatible ISA card, you need to add driver manually. This is accomplished by issuing **add** command under the **driver** submenu level.

To see system resources occupied by the installed devices, use the **/system resource io print** and **/system resource irq print** commands.

Property Description

io (*integer*) - input-output port base address

irq (*integer*) - interrupt request number

isdn-protocol (*euro* | *german*; default: **euro**) - line protocol setting for ISDN cards

memory (*integer*; default: **0**) - shared memory base address

name (*name*) - driver name

Notes

Not all combinatios of **irq** and **io** base addresses might work on your particular system. It is recommended, that you first find an acceptable irq setting and then try different i/o base addresses.

If you need to specify hexadecimal values instead of decimal for the argument values, put **0x** before the number.

To see the list of available drivers, issue the **/driver add name ?** command.

The resource list shows only those interfaces, which are enabled.

Typical io values for ISA cards are **0x280**, **0x300** and **0x320**

Example

To view the list of available drivers, do the following:

```
[admin@MikroTik] driver> add name ?
3c509  c101  lance  ne2k-isa  pc-isa
[admin@MikroTik] driver> add name
```

To see system resources occupied by the devices, use the **/system resource io print** and **/system resource irq print** commands:

```
[admin@MikroTik] system resource> io print
 PORT-RANGE          OWNER
 0x20-0x3F           APIC
 0x40-0x5F           timer
 0x60-0x6F           keyboard
 0x80-0x8F           DMA
 0xA0-0xBF           APIC
 0xC0-0xDF           DMA
 0xF0-0xFF           FPU
 0x100-0x13F         [prism2_cs]
 0x180-0x1BF         [orinoco_cs]
 0x1F0-0x1F7         IDE 1
 0x3D4-0x3D5         [cga]
 0x3F6-0x3F6         IDE 1
 0x3F8-0x3FF         serial port
 0xCF8-0xCFF         [PCI conf1]
```

```
  0x1000-0x10FF      [National Semiconductor Corporation DP83815 (MacPhyter) Et...
  0x1000-0x10FF      ether1
  0x1400-0x14FF      [National Semiconductor Corporation DP83815 (MacPhyter) Et...
  0x1400-0x14FF      ether2
  0x1800-0x18FF      [PCI device 100b:0511 (National Semiconductor Corporation)]
  0x1C00-0x1C3F      [PCI device 100b:0510 (National Semiconductor Corporation)]
  0x1C40-0x1C7F      [PCI device 100b:0510 (National Semiconductor Corporation)]
  0x1C80-0x1CBF      [PCI device 100b:0515 (National Semiconductor Corporation)]
  0x1CC0-0x1CCF      [National Semiconductor Corporation SCx200 IDE]
  0x4000-0x40FF      [PCI CardBus #01]
  0x4400-0x44FF      [PCI CardBus #01]
  0x4800-0x48FF      [PCI CardBus #05]
  0x4C00-0x4CFF      [PCI CardBus #05]

[admin@MikroTik] system resource> irq print
Flags: U - unused
   IRQ OWNER
   1   keyboard
   2   APIC
 U 3
   4   serial port
 U 5
 U 6
 U 7
 U 8
   9   ether1
   10  ether2
   11  [Texas Instruments PCI1250 PC card Cardbus Controller]
   11  [Texas Instruments PCI1250 PC card Cardbus Controller (#2)]
   11  [prism2_cs]
   11  [orinoco_cs]
   12  [usb-ohci]
 U 13
   14  IDE 1

[admin@MikroTik] system resource>
```

Suppose we need to load a driver for a NE2000 compatible ISA card. Assume we had considered the information above and have checked avalable resources in our system. To add the driver, we must do the following:

```
[admin@MikroTik] driver> add name=ne2k-isa io=0x280
[admin@MikroTik] driver> print
Flags: I - invalid, D - dynamic
  #   DRIVER                                IRQ IO       MEMORY    ISDN-PROTOCOL
  0 D RealTek 8139
  1 D Intel EtherExpressPro
  2 D PCI NE2000
  3   ISA NE2000                            280
  4   Moxa C101 Synchronous                              C8000
[admin@MikroTik] driver>
```

Removing Device Drivers

Description

You can remove only statically loaded drivers, *id est* those which do not have the **D** flag before the driver name. The device drivers can be removed only if the appropriate interface has been disabled.

To remove a device driver use the **/driver remove** command. Unloading a device driver is useful when you swap or remove a network device - it saves system resources by avoiding to load drivers for removed devices.

The device driver needs to be removed and loaded again, if some parameters (memory range, i/o base address) have been changed for the network interface card.

Notes on PCMCIA Adapters

Description

Currently only the following PCMCIA-ISA and PCMCIA-PCI adapters are tested to comply with MikroTik RouterOS:

- RICOH PCMCIA-PCI Bridge with R5C475 II or RC476 II chip (one or two PCMCIA ports)
- CISCO/Aironet PCMCIA adapter (ISA and PCI versions) for CISCO/Aironet PCMCIA cards only

Other PCMCIA-ISA and PCMCIA-PCI adapters might not function properly.

Notes

The Ricoh adapter might not work properly with some older motherboards. When recognized properly by the BIOS during the boot up of the router, it should be reported under the PCI device listing as "PCI/CardBus bridge". Try using another motherboard, if the adapter or the PCMCIA card are not recognized properly.

The maximum number of PCMCIA ports for a single system is equal to 8. If you will try to install 9 or more ports (no matter one-port or two-port adapters), no one will be recognized.

Troubleshooting

Description

- **My router shows that the ISA interface is invalid**
  The system cannot load driver for the card. Try to specify different IO or IRQ number

Ethernet Interfaces

Document revision 1.4 (September 10, 2007, 11:48 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

MikroTik RouterOS supports various types of Ethernet Interfaces. The complete list of supported Ethernet NICs can be found in the Device Driver List.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/interface ethernet***
Standards and Technologies: ***IEEE 802.3***
Hardware usage: ***Not significant***

Additional Documents

- http://www.ethermanage.com/ethernet/ethernet.html
- http://www.dcs.gla.ac.uk/~liddellj/nct/ethernet_protocol.html

Ethernet Interface Configuration

Home menu level: ***/interface ethernet***

Property Description

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; default: **enabled**) - Address Resolution Protocol

auto-negotiation (*yes* | *no*; default: **yes**) - when enabled, the interface "advertises" its maximum capabilities to achieve the best connection possible

cable-setting (*default* | *short* | *standard*; default: **default**) - changes the cable length setting (only applicable to NS DP83815/6 cards)

- **default** - suport long cables
- **short** - support short cables
- **standard** - same as default

disable-running-check (*yes* | *no*; default: **yes**) - disable running check. If this value is set to 'no', the router automatically detects whether the NIC is connected with a device in the network or not

full-duplex (*yes* | *no*; default: **yes**) - defines whether the transmission of data appears in two directions simultaneously

mac-address (*MAC address*) - set the Media Access Control number of the card

mdix-enable (yes | no) - whether the MDI/X auto crosscable correction feature is enabled for the port (if applicable)

mtu (*integer*; default: **1500**) - Maximum Transmission Unit

name (*name*; default: **etherN**) - assigned interface name, whrere 'N' is the number of the ethernet interface

speed (*10 Mbps* | *100 Mbps* | *1 Gbps*) - sets the data transmission speed of the interface. By default, this value is the maximal data rate supported by the interface

Command Description

blink (*name*) - blink the port's LEDs for about 10 seconds. Useful if you want to discover, which of the physical Ethernet ports is named as specified

reset-mac (*name*) - set the MAC address of the NIC to the factory default setting

Notes

When **disable-running-check** is set to **no**, the router automatically detects whether the NIC is connected to a device in the network or not. When the remote device is not connected (the leds are not blinking), the route which is set on the specific interface, becomes invalid.

Example

```
[admin@MikroTik] > interface print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                                              TYPE             MTU
 0  X ether1                                            ether            1500
[admin@MikroTik] > interface enable ether1
[admin@MikroTik] > interface print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                                              TYPE             MTU
 0  R ether1                                            ether            1500
[admin@MikroTik] > interface ethernet
[admin@MikroTik] interface ethernet> print
```

```
Flags: X - disabled, R - running
 #    NAME                                   MTU   MAC-ADDRESS       ARP
 0  R ether1                                 1500  00:0C:42:03:00:F2 enabled
[admin@MikroTik] interface ethernet> print detail
Flags: X - disabled, R - running
 0  R name="ether1" mtu=1500 mac-address=00:0C:42:03:00:F2 arp=enabled
      disable-running-check=no auto-negotiation=yes full-duplex=yes
      cable-settings=default mdix-enable=yes speed=100Mbps
[admin@MikroTik] interface ethernet>
```

Monitoring the Interface Status

Command name: ***/interface ethernet monitor***

Property Description

auto-negotiation (*done* | *incomplete*) - fast link pulses (FLP) to the adjacent link station to negotiate the SPEED and MODE of the link. Both stations choose the maximal speed boh support.

- **done** - negotiation done
- **incomplete** - negotiation failed

default-cable-setting (*read-only: short* | *standard*) - default cable length setting (only applicable to NS DP83815/6 cards)

- **short** - support short cables
- **standard** - same as default

full-duplex (*yes* | *no*) - whether transmission of data occurs in two directions simultaneously

rate (*10 Mbps* | *100 Mbps* | *1 Gbps*) - the actual data rate of the connection

status (*link-ok* | *no-link* | *unknown*) - status of the interface, one of the:

- **link-ok** - the card is connected to the network
- **no-link** - the card is not connected to the network (cable is not plugged in or faulty)
- **unknown** - the connection is not recognized (if the card does not report connection status)

Notes

See the IP Addresses and ARP section of the manual for information how to add **IP addresses** to the interfaces.

Example

```
[admin@MikroTik] interface ethernet> monitor ether1,ether2
                   status: link-ok link-ok
         auto-negotiation: done    done
                     rate: 100Mbps 100Mbps
              full-duplex: yes     yes
    default-cable-setting: standard standard
```

Troubleshooting

Description

- **Interface monitor shows wrong information**
  In some very rare cases it is possible that the device driver does not show correct information, but it does not affect the NIC's performance (of course, if your card is not broken)

FarSync X.21 Interface

Document revision 1.2 (February 6, 2008, 2:56 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS supports FarSync T-Series X.21 synchronous adapter hardware. These cards provide versatile high performance connectivity to the Internet or to corporate networks over leased lines.

Specifications

Packages required: ***synchronous***
License required: ***level4***
Home menu level: ***/interface farsync***
Standards and Technologies: ***X.21, Frame Relay, PPP***
Hardware usage: ***Not significant***

Additional Documents

- http://www.farsite.co.uk/

Synchronous Interface Configuration

Home menu level: ***/interface farsync***

Description

You can change the interface name to a more descriptive one using the **set** command. To enable the interface, use the **enable** command.

Property Description

clock-rate (*integer*; default: **64000**) - the speed of internal clock
clock-source (*external* | *internal*; default: **external**) - clock source
disabled (*yes* | *no*; default: **yes**) - shows whether the interface is disabled
frame-relay-dce (*yes* | *no*; default: **no**) - operate in Data Communications Equipment mode
frame-relay-lmi-type (*ansi* | *ccitt*; default: **ansi**) - Frame Relay Local Management Interface type
hdlc-keepalive (*time*; default: **10s**) - Cisco HDLC keepalive period in seconds
line-protocol (*cisco-hdlc* | *frame-relay* | *sync-ppp*; default: **sync-ppp**) - line protocol
media-type (*V24* | *V35* | *X21*; default: **V35**) - type of the media
mtu (*integer*; default: **1500**) - Maximum Transmit Unit
name (*name*; default: **farsyncN**) - assigned interface name

Example

```
[admin@MikroTik] interface farsync> print
Flags: X - disabled, R - running
  0    name="farsync1" mtu=1500 line-protocol=sync-ppp media-type=V35
       clock-rate=64000 clock-source=external chdlc-keepalive=10s
       frame-relay-lmi-type=ansi frame-relay-dce=no

  1    name="farsync2" mtu=1500 line-protocol=sync-ppp media-type=V35
       clock-rate=64000 clock-source=external chdlc-keepalive=10s
       frame-relay-lmi-type=ansi frame-relay-dce=no
[admin@MikroTik] interface farsync>
```

You can monitor the status of the synchronous interface:

```
[admin@MikroTik] interface farsync> monitor 0
           card-type: T2P FarSync T-Series
               state: running
         firmware-id: 2
    firmware-version: 0.7.0
      physical-media: V35
               cable: detected
               clock: not-detected
       input-signals: CTS
      output-signals: RTS DTR
[admin@MikroTik] interface farsync>
```

Troubleshooting

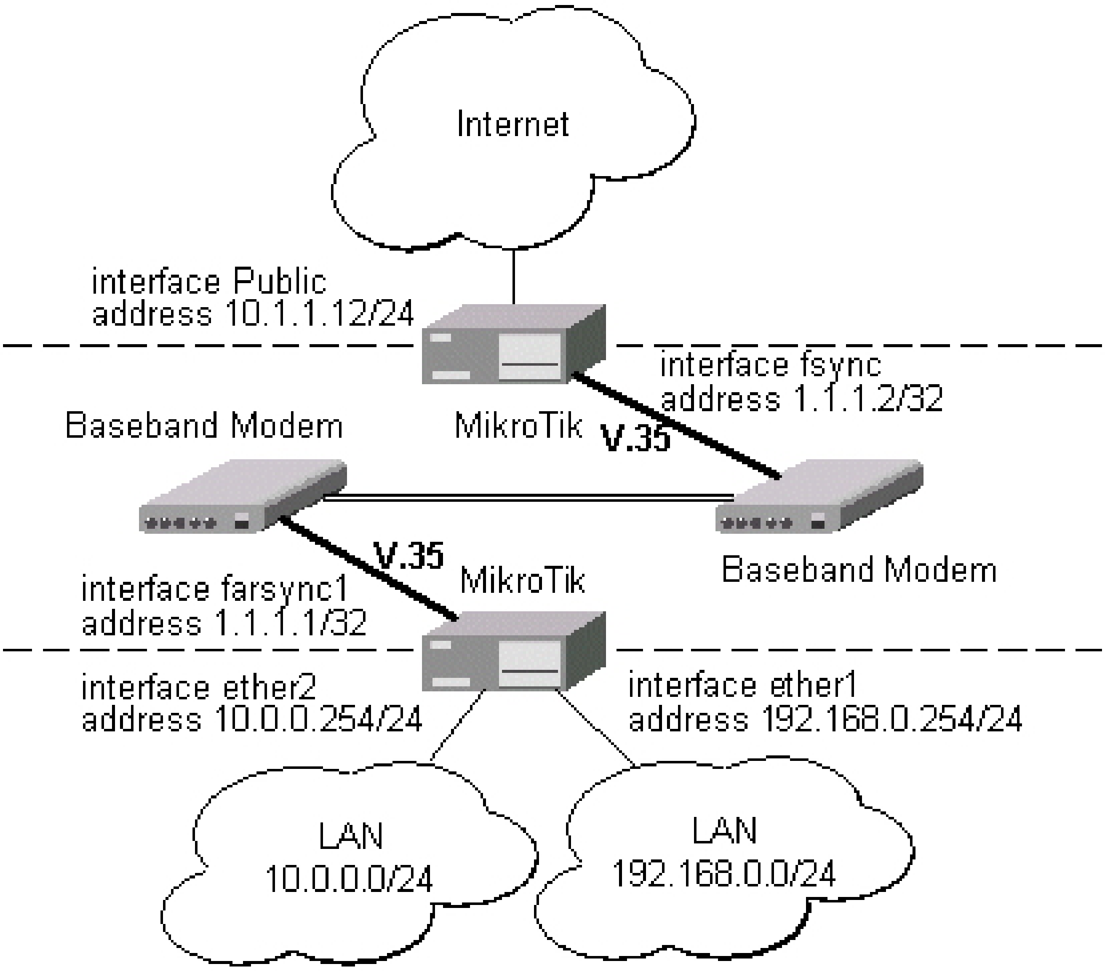
Description

- **The farsync interface does not show up under the interface list**
  Obtain the required license for synchronous feature
- **The synchronous link does not work**
  Check the cabling and the line between the modems. Read the modem manual

Synchronous Link Applications

MikroTik router to MikroTik router

Let us consider the following network setup with two MikroTik routers connected to a leased line with baseband modems:



The interface should be enabled according to the instructions given above. The **IP addresses** assigned to the synchronous interface should be as follows:

```
[admin@MikroTik] ip address> add address=1.1.1.1/32 interface=farsync1 \
\... network=1.1.1.2 broadcast=255.255.255.255
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.254/24      10.0.0.254      10.0.0.255      ether2
  1   192.168.0.254/24   192.168.0.254   192.168.0.255   ether1
  2   1.1.1.1/32         1.1.1.2         255.255.255.255 farsync1
[admin@MikroTik] ip address> /ping 1.1.1.2
1.1.1.2 64 byte ping: ttl=255 time=31 ms
1.1.1.2 64 byte ping: ttl=255 time=26 ms
1.1.1.2 64 byte ping: ttl=255 time=26 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 26/27.6/31 ms
[admin@MikroTik] ip address>
```

Note that for the point-to-point link the network mask is set to 32 bits, the argument **network** is set to the **IP address** of the other end, and the broadcast address is set to 255.255.255.255. The default route should be set to the gateway router 1.1.1.2:

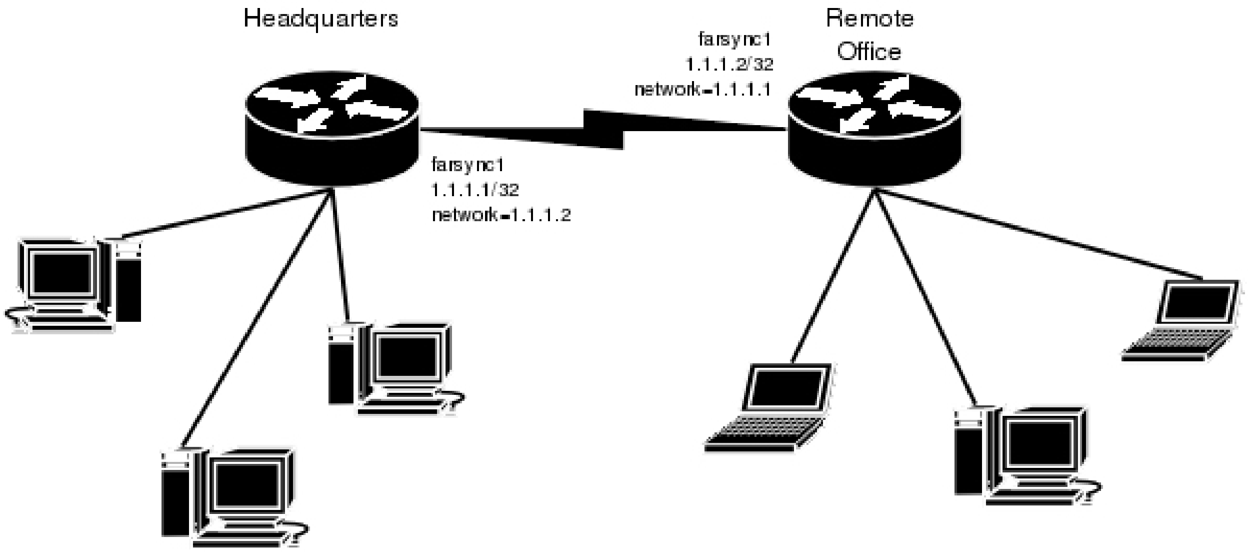
```
[admin@MikroTik] ip route> add gateway 1.1.1.2
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DISTANCE INTER...
 0 A S  0.0.0.0/0                          r 1.1.1.2         1        farsync1
 1 ADC  10.0.0.0/24        10.0.0.254      r                 0        ether2
 2 ADC  192.168.0.0/24     192.168.0.254   r                 0        ether1
 3 ADC  1.1.1.2/32         1.1.1.1         r                 0        farsync1
[admin@MikroTik] ip route>
```

The configuration of the MikroTik router at the other end is similar:

```
[admin@MikroTik] ip address> add address=1.1.1.2/32 interface=fsync \
\... network=1.1.1.1 broadcast=255.255.255.255
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.1.1.12/24       10.1.1.12       10.1.1.255      Public
  1   1.1.1.2/32         1.1.1.1         255.255.255.255 fsync
[admin@MikroTik] ip address> /ping 1.1.1.1
1.1.1.1 64 byte ping: ttl=255 time=31 ms
1.1.1.1 64 byte ping: ttl=255 time=26 ms
1.1.1.1 64 byte ping: ttl=255 time=26 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 26/27.6/31 ms
[admin@MikroTik] ip address>
```

MikroTik router to MikroTik router P2P using X.21 line

Consider the following example:



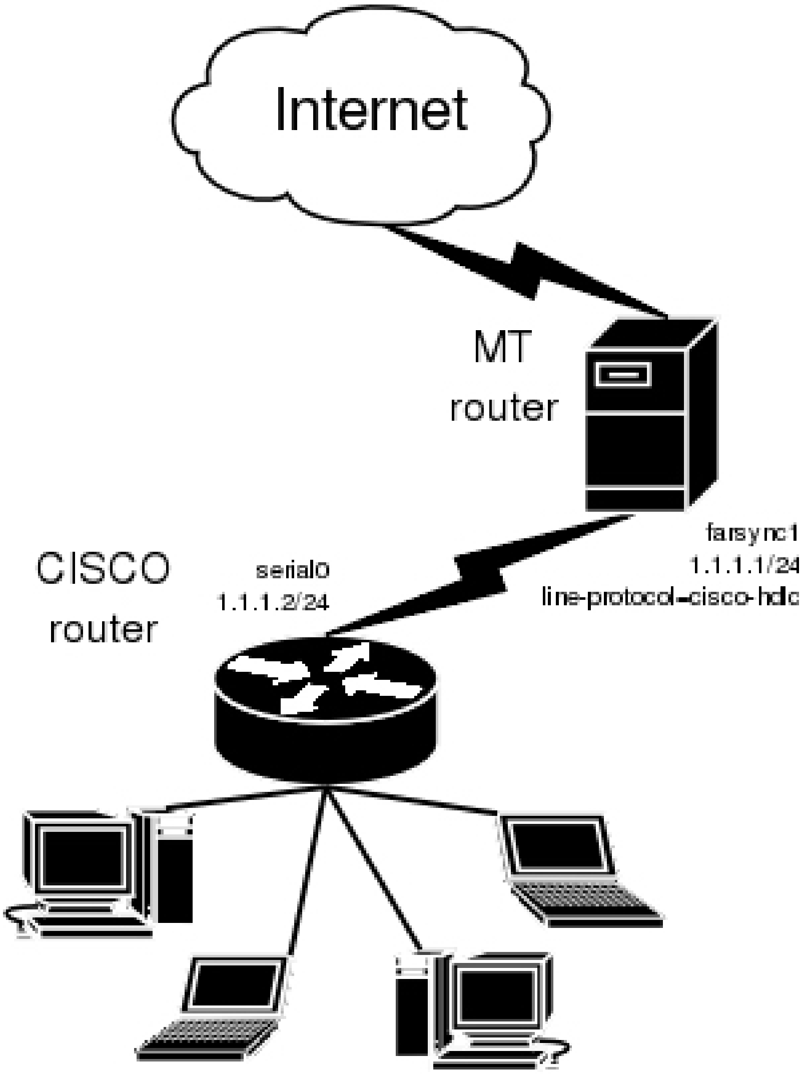
The default value of the property **clock-source** must be changed to **internal** for one of the cards. Both cards must have **media-type** property set to **X21**.

IP address configuration on both routers is as follows (by convention, the routers are named **hq** and **office** respectively):

```
[admin@hq] ip address> pri
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   192.168.0.1/24     192.168.0.0     192.168.0.255   ether1
  1   1.1.1.1/32         1.1.1.2         1.1.1.2         farsync1
[admin@hq] ip address>

[admin@office] ip address>
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.112/24      10.0.0.0        10.0.0.255      ether1
  1   1.1.1.2/32         1.1.1.1         1.1.1.1         farsync1
[admin@office] ip address>
```

MikroTik router to Cisco router using X.21 line

Assume we have the following configuration:



The configuration of MT router is as follows:

```
[admin@MikroTik] interface farsync> set farsync1 line-protocol=cisco-hdlc \
\... media-type=X21 clock-source=internal
[admin@MikroTik] interface farsync> enable farsync1
[admin@MikroTik] interface farsync> print
```

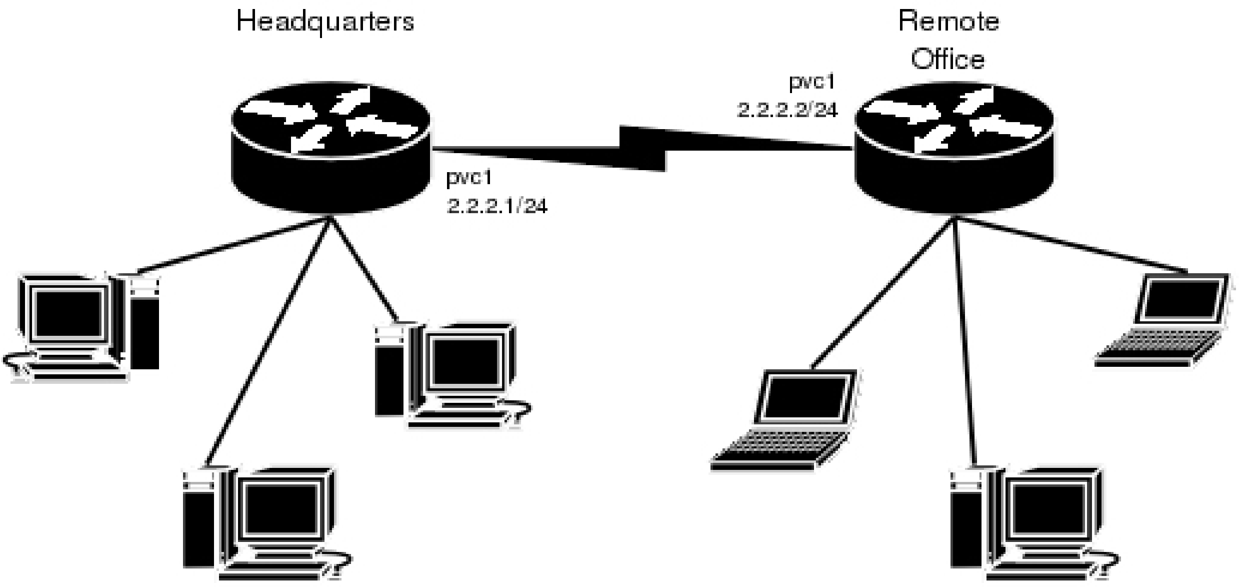
```
Flags: X - disabled, R - running
  0  R name="farsync1" mtu=1500 line-protocol=cisco-hdlc media-type=X21
       clock-rate=64000 clock-source=internal chdlc-keepalive=10s
       frame-relay-lmi-type=ansi frame-relay-dce=no

  1 X  name="farsync2" mtu=1500 line-protocol=sync-ppp media-type=V35
       clock-rate=64000 clock-source=external chdlc-keepalive=10s
       frame-relay-lmi-type=ansi frame-relay-dce=no
[admin@MikroTik] interface farsync>
[admin@MikroTik] interface farsync> /ip address add=address=1.1.1.1/24 \
\... interface=farsync1
```

The essential part of the configuration of Cisco router is provided below:

```
interface Serial0
 ip address 1.1.1.2 255.255.255.0
 no ip route-cache
 no ip mroute-cache
 no fair-queue
!
ip classless
ip route 0.0.0.0 0.0.0.0 1.1.1.1
```

MikroTik router to MikroTik router using Frame Relay

Consider the following example:



The default value of the property **clock-source** must be changed to **internal** for one of the cards. This card also requires the property **frame-relay-dce** set to **yes**. Both cards must have **media-type** property set to **X21** and the **line-protocol** set to **frame-relay**.

Now we need to add **pvc** interfaces:

```
[admin@hq] interface pvc> add dlci=42 interface=farsync1
[admin@hq] interface pvc> print
Flags: X - disabled, R - running
  #    NAME                                                MTU  DLCI INTERFACE
  0 X  pvc1                                                1500 42   farsync1
[admin@hq] interface pvc>
```

Similar routine has to be done also on **office** router:

```
[admin@office] interface pvc> add dlci=42 interface=farsync1
[admin@office] interface pvc> print
Flags: X - disabled, R - running
  #    NAME                                                 MTU  DLCI INTERFACE
  0 X  pvc1                                                 1500 42   farsync1
[admin@office] interface pvc>
```

Finally we need to add **IP addresses** to **pvc** interfaces and enable them.

On the **hq** router:

```
[admin@hq] interface pvc> /ip addr add address 2.2.2.1/24 interface pvc1
[admin@hq] interface pvc> /ip address print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.112/24      10.0.0.0        10.0.0.255      ether1
  1   192.168.0.1/24     192.168.0.0     192.168.0.255   ether2
  2   2.2.2.1/24         2.2.2.0         2.2.2.255       pvc1
[admin@hq] interface pvc> enable 0
[admin@hq] interface pvc>
```

and on the **office** router:

```
[admin@office] interface pvc> /ip addr add address 2.2.2.2/24 interface pvc1
[admin@office] interface pvc> /ip address print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.112/24      10.0.0.0        10.0.0.255      ether1
  1   2.2.2.2/24         2.2.2.0         2.2.2.255       pvc1
[admin@office] interface pvc> enable 0
[admin@office] interface pvc>
```

Now we can monitor the synchronous link status:

```
[admin@hq] interface pvc> /ping 2.2.2.2
2.2.2.2 64 byte ping: ttl=64 time=20 ms
2.2.2.2 64 byte ping: ttl=64 time=20 ms
2.2.2.2 64 byte ping: ttl=64 time=21 ms
2.2.2.2 64 byte ping: ttl=64 time=21 ms
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 20/20.5/21 ms
[admin@hq] interface pvc> /interface farsync monitor 0
           card-type: T2P FarSync T-Series
               state: running-normally
         firmware-id: 2
    firmware-version: 1.0.1
            physical: X.21
               cable: detected
               clock: detected
       input-signals: CTS
      output-signals: RTS,DTR
[admin@hq] interface pvc>
```

FrameRelay (PVC, Private Virtual Circuit) Interface

Document revision 1.2 (February 6, 2008, 2:56 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Frame Relay is a multiplexed interface to packet switched network and is a simplified form of Packet Switching similar in principle to X.25 in which synchronous frames of data are routed to different destinations depending on header information. Frame Relay uses the synchronous HDLC frame format.

Specifications

Packages required: ***synchronous***
License required: ***level4***
Home menu level: ***/interface pvc***
Standards and Technologies: ***Frame Relay (RFC1490)***
Hardware usage: ***Not significant***

Description

To use Frame Relay interface you must have already working synchronous interface. Please read how to set up your synchronous boards with MikroTik RouterOS first.

Additional Documents

- Frame Relay Forum
- http://www2.rad.com/networks/1994/fram_rel/frame.htm

Configuring Frame Relay Interface

Home menu level: ***/interface pvc***

Description

To configure frame relay, at first you should set up the synchronous interface, and then the PVC interface.

Property Description

dlci (*integer*; default: **16**) - Data Link Connection Identifier assigned to the PVC interface
interface (*name*) - Frame Relay interface
mtu (*integer*; default: **1500**) - Maximum Transmission Unit of an interface
name (*name*; default: **pvcN**) - assigned name of the interface

Notes

A DLCI is a channel number (Data Link Connection Identifier) which is attached to data frames to tell the network how to route the data. Frame Relay is "statistically multiplexed", which means that only one frame can be transmitted at a time but many logical connections can co-exist on a single physical line. The DLCI allows the data to be logically tied to one of the connections so that once it gets to the network, it knows where to send it.

Frame Relay Configuration

Example with Cyclades Interface

Let us consider the following network setup with MikroTik router with Cyclades PC300 interface connected to a leased line with baseband modems and a Cisco router at the other end.

```
[admin@MikroTik] ip address> add interface=pvc1 address=1.1.1.1/24
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   1.1.1.1/24         1.1.1.0         1.1.1.255       pvc1
[admin@MikroTik] ip address>
```

PVC and Cyclades interface configuration

- Cyclades

```
[admin@MikroTik] interface cyclades> print
Flags: X - disabled, R - running
  0  R name="cyclades1" mtu=1500 line-protocol=frame-relay media-type=V35
       clock-rate=64000 clock-source=external line-code=B8ZS framing-mode=ESF
       line-build-out=0dB rx-sensitivity=short-haul frame-relay-lmi-type=ansi
       frame-relay-dce=no chdlc-keepalive=10s
[admin@MikroTik] interface cyclades>
```

- PVC

```
[admin@MikroTik] interface pvc> print
Flags: X - disabled, R - running
  #    NAME                 MTU  DLCI INTERFACE
  0  R pvc1                 1500 42   cyclades1
[admin@MikroTik] interface pvc>
```

- Cisco router setup

```
CISCO# show running-config

Building configuration...

Current configuration...

...
!
ip subnet-zero
no ip domain-lookup
frame-relay switching
!
interface Ethernet0
 description connected to EthernetLAN
 ip address 10.0.0.254 255.255.255.0
!
interface Serial0
 description connected to Internet
 no ip address
 encapsulation frame-relay IETF
 serial restart-delay 1
 frame-relay lmi-type ansi
 frame-relay intf-type dce
!
interface Serial0.1 point-to-point
 ip address 1.1.1.2 255.255.255.0
 no arp frame-relay
 frame-relay interface-dlci 42
!
...
end.
```

Send ping to MikroTik router

```
CISCO#ping 1.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/31/32 ms
CISCO#
```

Example with MOXA Interface

Let us consider the following network setup with MikroTik router with MOXA C502 synchronous interface connected to a leased line with baseband modems and a Cisco router at the other end.

```
[admin@MikroTik] ip address> add interface=pvc1 address=1.1.1.1/24
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   1.1.1.1/24         1.1.1.0         1.1.1.255       pvc1
[admin@MikroTik] ip address>
```

PVC and Moxa interface configuration

- Moxa

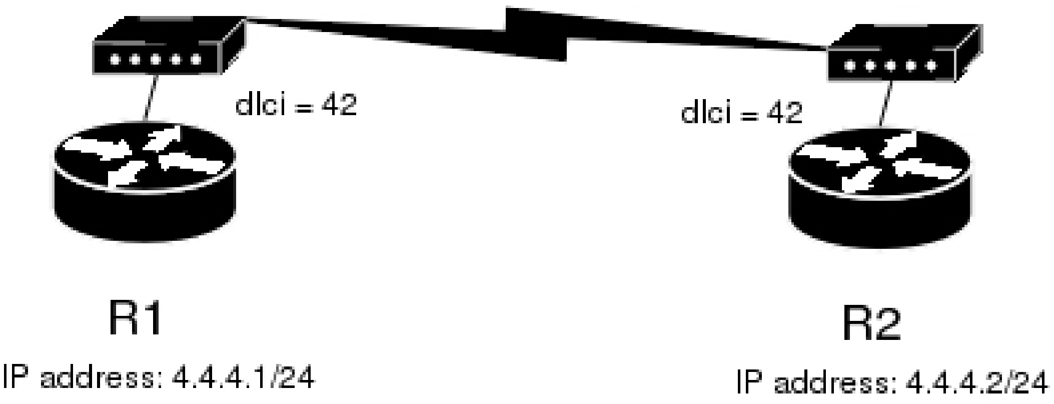
```
[admin@MikroTik] interface moxa-c502> print
Flags: X - disabled, R - running
  0  R name="moxa1" mtu=1500 line-protocol=frame-relay clock-rate=64000
       clock-source=external frame-relay-lmi-type=ansi frame-relay-dce=no
       cisco-hdlc-keepalive-interval=10s

  1 X  name="moxa-c502-2" mtu=1500 line-protocol=sync-ppp clock-rate=64000
       clock-source=external frame-relay-lmi-type=ansi frame-relay-dce=no
       cisco-hdlc-keepalive-interval=10s
[admin@MikroTik] interface moxa-c502>
```

- PVC

```
[admin@MikroTik] interface pvc> print
Flags: X - disabled, R - running
  #    NAME                 MTU  DLCI INTERFACE
  0  R pvc1                 1500 42   moxa1
[admin@MikroTik] interface pvc>

CISCO router setup

CISCO# show running-config

Building configuration...

Current configuration...

...
!
ip subnet-zero
no ip domain-lookup
frame-relay switching
!
interface Ethernet0
 description connected to EthernetLAN
 ip address 10.0.0.254 255.255.255.0
!
interface Serial0
 description connected to Internet
 no ip address
 encapsulation frame-relay IETF
 serial restart-delay 1
 frame-relay lmi-type ansi
 frame-relay intf-type dce
!
interface Serial0.1 point-to-point
 ip address 1.1.1.2 255.255.255.0
 no arp frame-relay
 frame-relay interface-dlci 42
!
...
end.

Send ping to MikroTik router

CISCO#ping 1.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/31/32 ms
CISCO#
```

Example with MikroTik Router to MikroTik Router

Let us consider the following example:



In this example we will use two Moxa C101 synchronous cards.

Do not forget to set **line-protocol** for synchronous interfaces to **frame-relay**. To achieve proper result, one of the synchronous interfaces must operate in DCE mode:

```
[admin@r1] interface moxa-c101> set 0 frame-relay-dce=yes
[admin@r1] interface moxa-c101> print
Flags: X - disabled, R - running
  0  R name="moxa-c101-1" mtu=1500 line-protocol=frame-relay clock-rate=64000
       clock-source=external frame-relay-lmi-type=ansi frame-relay-dce=yes
       cisco-hdlc-keepalive-interval=10s ignore-dcd=no
[admin@r1] interface moxa-c101>
```

Then we need to add PVC interfaces and **IP addresses**.

On the **R1**:

```
[admin@r1] interface pvc> add dlci=42 interface=moxa-c101-1
[admin@r1] interface pvc> print
Flags: X - disabled, R - running
  #    NAME                                               MTU  DLCI INTERFACE
  0 X  pvc1                                               1500 42   moxa-c101-1
[admin@r1] interface pvc> /ip address add address=4.4.4.1/24 interface=pvc1
```

on the **R2**:

```
[admin@r2] interface pvc> add dlci=42 interface=moxa-c101-1
[admin@r2] interface pvc> print
Flags: X - disabled, R - running
  #    NAME                                               MTU  DLCI INTERFACE
  0 X  pvc1                                               1500 42   moxa-c101-1
[admin@r2] interface pvc> /ip address add address 4.4.4.2/24 interface=pvc1
```

Finally, we must enable PVC interfaces:

```
[admin@r1] interface pvc> enable pvc1
[admin@r1] interface pvc>

[admin@r2] interface pvc> enable pvc1
[admin@r2] interface pvc>
```

Troubleshooting

Description

- **I cannot ping through the synchronous frame relay interface between MikroTik router and**

a Cisco router

Frame Relay does not support address resolving and IETF encapsulation should be used. Please check the configuration on the Cisco router

General Interface Settings

Document revision 1.3 (September 10, 2007, 12:57 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

MikroTik RouterOS supports a variety of Network Interface Cards as well as some virtual interfaces (like Bonding, Bridge, VLAN etc.). Each of them has its own submenu, but there is also a list of all interfaces where some common properties can be configured.

Description

The Manual describes general settings of MikroTik RouterOS interfaces.

Interface Status

Home menu level: ***/interface***

Property Description

mtu (*integer*) - maximum transmission unit for the interface (in bytes)

name (*text*) - the name of the interface

type (*read-only: arlan* | *bonding* | *bridge* | *cyclades* | *eoip* | *ethernet* | *farsync* | *ipip* | *isdn-client* | *isdn-server* | *l2tp-client* | *l2tp-server* | *moxa-c101* | *moxa-c502* | *mtsync* | *pc* | *ppp-client* | *ppp-server* | *pppoe-client* | *pppoe-server* | *pptp-client* | *pptp-server* | *pvc* | *radiolan* | *sbe* | *vlan* | *wavelan* | *wireless* | *xpeed*) - interface type

Example

To see the list of all available interfaces:

```
[admin@MikroTik] interface> print
Flags: X - disabled, D - dynamic, R - running
 #    NAME                                              TYPE             MTU
 0  R ether1                                            ether            1500
 1  R bridge1                                           bridge           1500
 2  R ether2                                            ether            1500
 3  R wlan1                                             wlan             1500
[admin@MikroTik] interface>
```

Traffic Monitoring

Command name: ***/interface monitor-traffic***

Description

The traffic passing through any interface can be monitored.

Property Description

received-bits-per-second (*read-only: integer*) - number of bits that interface has received in one second

received-packets-per-second (*read-only: integer*) - number of packets that interface has received in one second

sent-bits-per-second (*read-only: integer*) - number of bits that interface has sent in one second

sent-packets-per-second (*read-only: integer*) - number of packets that interface has sent in one second

Notes

One or more interfaces can be monitored at the same time.

To see overall traffic passing through all interfaces at time, use **aggregate** instead of interface name.

Example

Multiple interface monitoring:

```
/interface monitor-traffic ether1,aggregate
    received-packets-per-second: 9        11
       received-bits-per-second: 4.39kbps 6.19kbps
        sent-packets-per-second: 16       17
           sent-bits-per-second: 101kbps  101kbps
-- [Q quit|D dump|C-z pause]
```

GPRS PCMCIA

Document revision 1.1 (February 6, 2008, 2:56 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

How to make a GPRS connection

Description

Let us consider a situation that you are in a place where no internet connection is available, but you have access to your mobile network provider. In this case you can connect MikroTik router to your mobile phone provider using GPRS (General Packet Radio Service) and so establish an internet connection.

Example

- Plug the GPRS PCMCIA card (with your SIM card) into the router, turn on the router and after it has started, see if a new port has appeared. In this case it is the **serial1** port which is our GPRS device:

```
[admin@MikroTik] port> print
 # NAME                                USED-BY                          BAUD-RATE
 0 serial0                             Serial Console                   115200
 1 serial1                                                              9600
[admin@MikroTik] port>
```

- Enter the pin code from serial-terminal (in this case, PIN code is **3663**) :

```
/system serial-terminal serial1

AT+CPIN=”3663”
```

  Now you should see **OK** on your screen. Wait for about 5 seconds and see if the green led started to blink. Press Ctrl+Q to quit the serial-terminal.

- Change remote-address in **/ppp profile**, in this case to 212.93.96.65 (you should obtain it from your mobile network operator):

```
/ppp profile set default remote-address=212.93.96.65
```

- Add a ppp client:

```
/interface ppp-client add dial-command=ATD phone=*99***1# \
\... modem-init="AT+CGDCONT=1,\"IP\",\"internet\"" port=serial1
```

- Now enable the interface and see if it is connected:

```
[admin@MikroTik] interface ppp-client> enable 0
[admin@MikroTik] interface ppp-client> mo 0
    status: dialing...
    status: link established

       status: authenticated
       uptime: 0s
    idle-time: 0s

       status: authenticated
       uptime: 1s
    idle-time: 1s

       status: connected
       uptime: 2s
    idle-time: 2s
[admin@MikroTik] interface ppp-client>
```

Check the IP addresses:

```
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   192.168.0.5/24     192.168.0.0     192.168.0.255   ether1
 1 D 10.40.205.168/32   212.93.96.65    0.0.0.0         ppp-out1
[admin@MikroTik] ip address>
```

ISDN (Integrated Services Digital Network) Interface

Document revision 1.2 (February 6, 2008, 2:56 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik router can act as an ISDN client for dialing out, or as an ISDN server for accepting incoming calls. The dial-out connections may be set as dial-on-demand or as permanent connections (simulating a leased line). The remote **IP address** (provided by the ISP) can be used as the default gateway for the router.

Specifications

Packages required: ***isdn, ppp***
License required: ***level1***
Home menu level: ***/interface isdn-server, /interface isdn-client***
Standards and Technologies: ***PPP (RFC 1661)***
Hardware usage: ***Not significant***

Additional Documents

- PPP over ISDN
- RFC3057 - ISDN Q.921-User Adaptation Layer

ISDN Hardware and Software Installation

Command name: ***/driver add***

Description

Please install the ISDN adapter into the PC accordingly the instructions provided by the adapter manufacturer.

Appropriate packages have to be downloaded from MikroTik web page http://www.mikrotik.com. After all, the ISDN driver should be loaded using the **/driver add** command.

MikroTik RouterOS supports passive PCI adapters with Siemens chipset:

- Eicon. Diehl Diva - **diva**
- Sedlbauer Speed - **sedlbauer**
- ELSA Quickstep 1000 - **quickstep**
- NETjet - **netjet**
- Teles - **teles**
- Dr. Neuhaus Niccy - **niccy**
- AVM - **avm**
- Gazel - **gazel**
- HFC 2BDS0 based adapters - **hfc**
- W6692 based adapters - **w6692**

For example, for the HFC based PCI card, it is enough to use **/driver add name=hfc** command to get the driver loaded.

Note! ISDN **ISA** adapters are **not** supported!

Property Description

isdn-protocol (*euro* | *german*; default: **euro**) - data channel protocol
name (*name*) - name of the driver

ISDN Channels

ISDN channels are added to the system automatically when the ISDN card driver is loaded. Each channel corresponds to one physical 64K ISDN data channel.

The list of available ISDN channels can be viewed using the **/isdn-channels print** command. The channels are named **channel1**, **channel2**, and so on. E.g., if you have two ISDN channels, and one of them currently used by an ISDN interface, but the other available, the output should look like this:

```
[admin@MikroTik] isdn-channels> print
Flags: X - disabled, E - exclusive
  #    NAME                     CHANNEL    DIR.. TYPE  PHONE
  0    channel1                 0
  1    channel2                 1
[admin@MikroTik] isdn-channels>
```

ISDN channels are very similar to PPP serial ports. Any number of ISDN interfaces can be configured on a single channel, but only one interface can be enabled for that channel at a time. It means that every ISDN channel is either available or used by an ISDN interface.

MSN and EAZ numbers

In Euro-ISDN a subscriber can assign more than one ISDN number to an ISDN line. For example, an ISDN line could have the numbers 1234067 and 1234068. Each of these numbers can be used to dial the ISDN line. These numbers are referred to as Multiple Subscriber Numbers (MSN).

A similar, but separate concept is EAZ numbering, which is used in German ISDN networking. EAZ number can be used in addition to dialed phone number to specify the required service.

For dial-out ISDN interfaces, MSN/EAZ number specifies the outgoing phone number (the calling end). For dial-in ISDN interfaces, MSN/EAZ number specifies the phone number that will be answered. If you are unsure about your MSN/EAZ numbers, leave them blank (it is the default).

For example, if your ISDN line has numbers 1234067 and 1234068, you could configure your dial-in server to answer only calls to 1234068 by specifying **1234068** as your MSN number. In a sense, MSN is just your phone number.

ISDN Client Interface Configuration

Home menu level: ***/interface isdn-client***

Description

The ISDN client is used to connect to remote dial-in server (probably ISP) via ISDN. To set up an ISDN dial-out connection, use the ISDN dial-out configuration menu under the submenu.

Property Description

add-default-route (*yes* | *no*; default: **no**) - add default route to remote host on connect

allow (*multiple choice: mschap2, mschap1, chap, pap*; default: **mschap2, mschap1, chap, pap**) - the protocol to allow the client to use for authentication

bundle-128K (*yes* | *no*; default: **yes**) - use both channels instead of just one

dial-on-demand (*yes* | *no*; default: **no**) - use dialing on demand

l2-protocol (*hdlc* | *x75i* | *x75ui* | *x75bui*; default: **hdlc**) - level 2 protocol to be used

mru (*integer*; default: **1500**) - Maximum Receive Unit

msn (*integer*; default: **""**) - MSN/EAZ of ISDN line provided by the line operator

mtu (*integer*; default: **1500**) - Maximum Transmission Unit

name (*name*; default: **isdn-outN**) - interface name

password (*text*) - password that will be provided to the remote server

phone (*integer*; default: **""**) - phone number to dial

profile (*name*; default: **default**) - profile to use when connecting to the remote server

use-peer-dns (*yes* | *no*; default: **no**) - use or not peer DNS

user (*text*) - user name that will be provided to the remote server

Example

ISDN client interfaces can be added using the **add** command:

```
[admin@MikroTik] interface isdn-client> add msn="142" user="test" \
\... password="test" phone="144" bundle-128K=no
[admin@MikroTik] interface isdn-client> print
Flags: X - disabled, R - running
  0 X  name="isdn-out1" mtu=1500 mru=1500 msn="142" user="test"
       password="test" profile=default phone="144" l2-protocol=hdlc
       bundle-128K=no dial-on-demand=no add-default-route=no use-peer-dns=no
[admin@MikroTik] interface isdn-client>
```

ISDN Server Interface Configuration

Home menu level: ***/interface isdn-client***

Description

ISDN server is used to accept remote dial-in connections form ISDN clients.

Property Description

authentication (*pap* | *chap* | *mschap1* | *mschap2*; default: **mschap2, mschap1, chap, pap**) - used authentication

bundle-128K (*yes* | *no*; default: **yes**) - use both channels instead of just one

l2-protocol (*hdlc* | *x75i* | *x75ui* | *x75bui*; default: **hdlc**) - level 2 protocol to be used

mru (*integer*; default: **1500**) - Maximum Receive Unit

msn (*integer*; default: **""**) - MSN/EAZ of ISDN line provided by the line operator

mtu (*integer*; default: **1500**) - Maximum Transmission Unit

name (*name*; default: **isdn-inN**) - interface name

phone (*integer*; default: **""**) - phone number to dial

profile (*name*; default: **default**) - profile to use when connecting to the remote server

Example

ISDN server interfaces can be added using the **add** command:

```
[admin@MikroTik] interface isdn-server> add msn="142" bundle-128K=no
```

```
[admin@MikroTik] interface isdn-server> print
Flags: X - disabled, R - running
  0 X  name="isdn-in1" mtu=1500 mru=1500 msn="142"
       authentication=mschap2,chap,pap profile=default l2-protocol=x75bui
       bundle-128K=no
[admin@MikroTik] interface isdn-server>
```

ISDN Examples

ISDN Dial-out

Dial-out ISDN connections allow a local router to connect to a remote dial-in server (ISP's) via ISDN.

Let's assume you would like to set up a router that connects your local LAN with your ISP via ISDN line. First you should load the corresponding ISDN card driver. Supposing you have an ISDN card with a **W6692**-based chip:

```
[admin@MikroTik]> /driver add name=w6692
```

Now additional channels should appear. Assuming you have only one ISDN card driver loaded, you should get following:

```
[admin@MikroTik] isdn-channels> print
Flags: X - disabled, E - exclusive
  #    NAME                        CHANNEL    DIR.. TYPE  PHONE
  0    channel1                    0
  1    channel2                    1
[admin@MikroTik] isdn-channels>
```

Suppose you would like to use dial-on-demand to dial your ISP and automatically add a default route to it. Also, you would like to disconnect when there is more than 30s of network inactivity. Your ISP's phone number is 12345678 and the user name for authentication is 'john'. Your ISP assigns IP addresses automatically. Add an outgoing ISDN interface and configure it in the following way:

```
[admin@mikrotik]> /interface isdn-client add name="isdn-isp" phone="12345678"
user="john" password="31337!)" add-default-route=yes dial-on-demand=yes
[admin@MikroTik] > /interface isdn-client print
Flags: X - disabled, R - running
  0 X  name="isdn-isp" mtu=1500 mru=1500 msn="" user="john" password="31337!)"
       profile=default phone="12345678" l2-protocol=hdlc bundle-128K=no
       dial-on-demand=yes add-default-route=yes use-peer-dns=no
```

Configure PPP profile.

```
[admin@MikroTik] ppp profile> print
Flags: * - default
 0 * name="default" use-compression=default use-vj-compression=default
     use-encryption=default only-one=default change-tcp-mss=yes

 1 * name="default-encryption" use-compression=default
     use-vj-compression=default use-encryption=yes only-one=default
     change-tcp-mss=yes
[admin@Mikrotik] ppp profile> set default idle-timeout=30s
```

If you would like to remain connected all the time, i.e., as a leased line, then set the **idle-timeout** to 0s.

All that remains is to enable the interface:

```
[admin@MikroTik] /interface set isdn-isp disabled=no
```

You can monitor the connection status with the following command:

```
[admin@MikroTik] /interface isdn-client monitor isdn-isp
```

ISDN Dial-in

Dial-in ISDN connections allow remote clients to connect to your router via ISDN.

Let us assume you would like to configure a router for accepting incoming ISDN calls from remote clients. You have an Ethernet card connected to the LAN, and an ISDN card connected to the ISDN line. First you should load the corresponding ISDN card driver. Supposing you have an ISDN card with an HFC chip:

```
[admin@MikroTik] /driver add name=hfc
```

Now additional channels should appear. Assuming you have only one ISDN card driver loaded, you should get the following:

```
[admin@MikroTik] isdn-channels> print
Flags: X - disabled, E - exclusive
  #    NAME                       CHANNEL    DIR.. TYPE  PHONE
  0    channel1                   0
  1    channel2                   1
[admin@MikroTik] isdn-channels>
```

Add an incoming ISDN interface and configure it in the following way:

```
[admin@MikroTik] interface isdn-server> add msn="7542159" \
\... authentication=chap,pap bundle-128K=no
[admin@MikroTik] interface isdn-server> print
Flags: X - disabled
  0 X  name="isdn-in1" mtu=1500 mru=1500 msn="7542159" authentication=chap,pap
       profile=default l2-protocol=hldc bundle-128K=no
```

Configure PPP settings and add users to router's database.

```
[admin@MikroTik] ppp profile> print
Flags: * - default
 0 * name="default" use-compression=default use-vj-compression=default
     use-encryption=default only-one=default change-tcp-mss=yes

 1 * name="default-encryption" use-compression=default
     use-vj-compression=default use-encryption=yes only-one=default
     change-tcp-mss=yes
[admin@Mikrotik] ppp profile> set default idle-timeout=5s local-address=10.99.8.1 \
\... remote-address=10.9.88.1
```

Add user 'john' to the router's user database. Assuming that the password is '31337!)':

```
[admin@MikroTik] ppp secret> add name=john password="31337!)" service=isdn
[admin@MikroTik] ppp secret> print
Flags: X - disabled
 #   NAME                    SERVICE CALLER-ID PASSWORD PROFILE REMOTE-ADDRESS
 0   john                    isdn              31337!)  default
[admin@MikroTik] ppp secret>
```

Check the status of the ISDN server interface and wait for the call:

```
[admin@MikroTik] interface isdn-server> monitor isdn-in1

    status: Waiting for call...
```

ISDN Backup

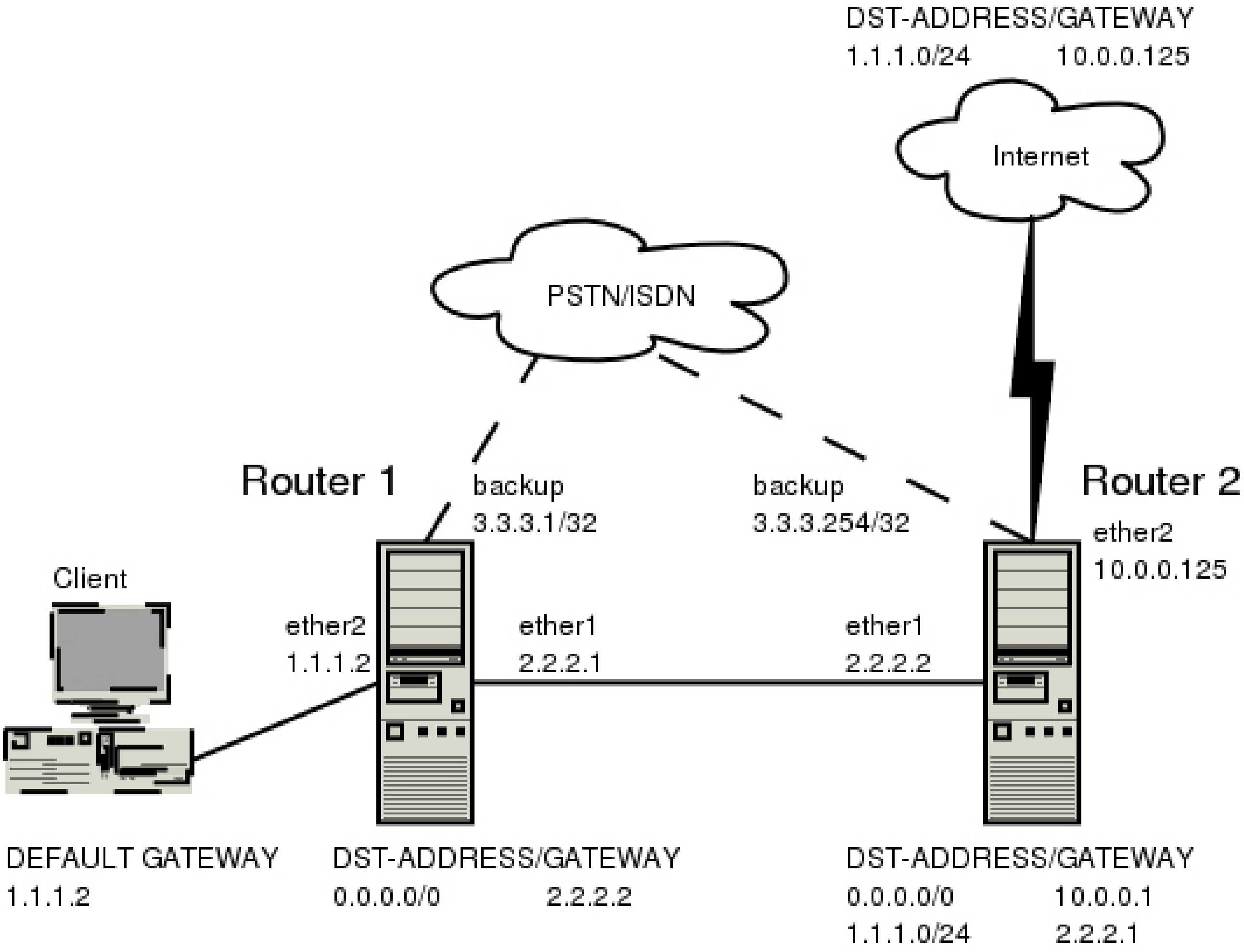
Backup systems are used in specific cases, when you need to maintain a connection, even if a fault occurs.

For example, if someone cuts the wires, the router can automatically connect to a different interface to continue its work. Such a backup is based on an utility that monitors the status of the connection - netwatch, and a script, which runs the netwatch.

This is an example of how to make simple router backup system. In this example we'll use an ISDN connection for purpose to backup a standard Ethernet connection. You can, however, use instead of the ISDN connection anything you need - PPP, for example. When the Ethernet fail (the router nr.1 cannot ping the router nr.2 to 2.2.2.2 (see picture) the router nr.1 will establish an ISDN connection, so-called backup link, to continue communicating with the nr. 2.

You must keep in mind, that in our case there are just two routers, but this system can be extended to support more different networks.

The backup system example is shown in the following picture:



In this case the **backup** interface is an ISDN connection, but in real applications it can be substituted by a particular connection. Follow the instructions below on how to set up the backup link:

- At first, you need to set up ISDN connection. To use ISDN, the ISDN card driver must be loaded:

```
[admin@MikroTik] driver> add name=hfc
```

The PPP connection must have a new user added to the routers one and two:

```
[admin@Mikrotik] ppp secret> add name=backup password=backup service=isdn
```

An ISDN server and PPP profile must be set up on the second router:

```
[admin@MikroTik] ppp profile> set default local-address=3.3.3.254
remote-address=3.3.3.1
[admin@MikroTik] interface isdn-server> add name=backup msn=7801032
```

An ISDN client must be added to the first router:

```
[admin@MikroTik] interface isdn-client>
add name=backup user="backup" password="backup" phone=7801032 msn=7542159
```

- Then, you have to set up static routes
  Use the **/ip route add** command to add the required static routes and comments to them. Comments are required for references in scripts.
  The **first** router:

```
[admin@Mikrotik] ip route> add gateway=2.2.2.2 comment="route1"
```

The **second** router:

```
[admin@Mikrotik] ip route> add gateway=2.2.2.1 comment="route1" dst-address=1.1.1.0/24
```

- And finally, you have to add scripts.
  Add scripts in the submenu **/system script** using the following commands:
  The **first** router:

```
[admin@Mikrotik] system script> add name=connection_down \
\... source={/interface enable backup; /ip route set route1 gateway=3.3.3.254}
[admin@Mikrotik] system script> add name=connection_up \
\... source={/interface disable backup; /ip route set route1 gateway=2.2.2.2}
```

The **second** router:

```
[admin@Mikrotik] system script> add name=connection_down \
\... source={/ip route set route1 gateway=3.3.3.1}
[admin@Mikrotik] system script> add name=connection_up \
\... source={/ip route set route1 gateway=2.2.2.1}
```

- To get all above listed to work, set up Netwatch utility. To use netwatch, you need the advanced tools feature package installed. Please upload it to the router and reboot. When installed, the advanced-tools package should be listed under the **/system package print** list.
  Add the following settings to the first router:

```
[admin@Mikrotik] tool netwatch> add host=2.2.2.1 interval=5s \
\... up-script=connection_up down-script=connection_down
```

Add the following settings to the second router:

```
[admin@Mikrotik] tool netwatch> add host=2.2.2.2 interval=5s \
\... up-script=connection_up down-script=connection_down
```

M3P

Document revision 0.4 (February 6, 2008, 4:21 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik Packet Packer Protocol (M3P) optimizes the data rate usage of links using protocols that have a high overhead per packet transmitted. The basic purpose of this protocol is to better enable wireless networks to transport VoIP traffic and other traffic that uses small packet sizes of around 100 bytes.

M3P features:

- enabled by a per interface setting
- other routers with MikroTik Discovery Protocol enabled will broadcast M3P settings
- significantly increases bandwidth availability over some wireless links by approximately four times
- offer configuration settings to customize this feature

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip packing***
Standards and Technologies: ***M3P***
Hardware usage: ***Not significant***

Description

The wireless protocol IEEE 802.11 and, to a lesser extent, Ethernet protocol have a high overhead per packet as for each packet it is necessary to access the media, check for errors, resend in case of errors occured, and send network maintenance messages (network maintenance is applicable only for wireless). The MikroTik Packet Packer Protocol improves network performance by aggregating many small packets into a big packet, thereby minimizing the network per packet overhead cost. The M3P is very effective when the average packet size is 50-300 bytes the common size of VoIP packets.

Features:

- may work on any Ethernet-like media
- is disabled by default for all interfaces
- when older version on the RouterOS are upgraded from a version without M3P to a version with discovery, current wireless interfaces will not be automatically enabled for M3P
- small packets going to the same MAC level destination (regardless of IP destination) are collected according to the set configuration and aggregated into a large packet according to the set size
- the packet is sent as soon as the maximum aggregated-packet packet size is reached or a maximum time of 15ms (+/-5ms)

Setup

Home menu level: ***/ip packing***

Description

M3P is working only between MikroTik routers, which are discovered with MikroTik Neighbor Discovery Protocol (MNDP). When M3P is enabled router needs to know which of its neighbouring hosts have enabled M3P. MNDP is used to negotiate unpacking settings of neighbours, therefore it has to be enabled on interfaces you wish to enable M3P. Consult MNDP manual on how to do it.

Property Description

aggregated-size (*integer*; default: **1500**) - the maximum aggregated packet's size

interface (*name*) - interface to enable M3P on

packing (*none* | *simple* | *compress-all* | *compress-headers*; default: **simple**) - specifies the packing mode

- **none** - no packing is applied to packets
- **simple** - aggregate many small packets into one large packet, minimizing network overhead per packet
- **compress-headers** - further increase network performance by compressing IP packet header (consumes more CPU resources)
- **compress-all** - increase network performance even more by using header and data compression (extensive CPU usage)

unpacking (*none* | *simple* | *compress-all* | *compress-headers*; default: **simple**) - specifies the unpacking mode

- **none** - accept only usual packets
- **simple** - accept usual packets and aggregated packets without compression
- **compress-headers** - accept all packets except those with payload compression
- **compress-all** - accept all packets

Notes

Level of packet compression increases like this: **none -> simple -> compress-headers -> compress-all**.

When router has to send a packet it choses minimum level of packet compression from what its own **packing** type is set and what other router's **unpacking** type is set. Same is with **aggregated-size** setting - minimum value of both ends is actual maximum size of aggregated packet used.

aggregated-size can be bigger than interface MTU if network device allows it to be (i.e., it supports sending and receiving frames bigger than 1514 bytes)

Example

To enable maximal compression on the **ether1** interface:

```
[admin@MikroTik] ip packing> add interface=ether1 packing=compress-all \
\... unpacking=compress-all
[admin@MikroTik] ip packing> print
Flags: X - disabled
  #   INTERFACE PACKING          UNPACKING        AGGREGATED-SIZE
  0   ether1    compress-all     compress-all     1500

[admin@MikroTik] ip packing>
```

MOXA C101 Synchronous Interface

Document revision 1.3 (February 6, 2008, 2:58 GMT)

This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS supports MOXA C101 Synchronous 4Mb/s Adapter hardware. The V.35 synchronous interface is the standard for VSAT and other satellite modems. However, you must check with the satellite system supplier for the modem interface type.

Specifications

Packages required: ***synchronous***
License required: ***level4***
Home menu level: ***/interface moxa-c101***
Standards and Technologies: ***Cisco/HDLC-X.25 (RFC 1356)***, ***Frame Relay (RFC1490)***, ***PPP (RFC-1661)***, ***PPP (RFC-1662)***
Hardware usage: ***Not significant***

Description

You can install up to four MOXA C101 synchronous cards in one PC box, if you have so many slots and IRQs available. Assuming you have all necessary packages and licenses installed, in most cases it should to be done nothing at that point (all drivers are loaded automatically). However, if you have a non Plug-and-Play ISA card, the corresponding driver requires to be loaded.

MOXA C101 PCI variant cabling

The MOXA C101 PCI requires different from MOXA C101 ISA cable. It can be made using the following table:

DB25f	Signal	Direction	V.35m
4	RTS	OUT	C
5	CTS	IN	D
6	DSR	IN	E
7	GND	-	B
8	DCD	IN	F
10	TxDB	OUT	S
11	TxDA	OUT	P
12	RxDB	IN	T
13	RxDA	IN	R
14	TxCB	IN	AA
16	TxCA	IN	Y
20	DTR	OUT	H
22	RxCB	IN	X
23	RxCA	IN	V
short 9 and 25 pin			

Additional Documents

For more information about the MOXA C101 synchronous 4Mb/s adapter hardware please see:

- http://www.moxa.com/product/sync/C101.htm - the product on-line documentation
- C101 SuperSync Board User's Manual the user's manual in PDF format

Synchronous Interface Configuration

Home menu level: ***/interface moxa-c101***

Description

Moxa c101 synchronous interface is shown under the interfaces list with the name moxa-c101-N

Property Description

cisco-hdlc-keepalive-interval (*time*; default: **10s**) - keepalive period in seconds
clock-rate (*integer*; default: **64000**) - speed of internal clock

clock-source (*external* | *internal* | *tx-from-rx* | *tx-internal*; default: **external**) - clock source

frame-relay-dce (*yes* | *no*; default: **no**) - operate or not in DCE mode

frame-relay-lmi-type (*ansi* | *ccitt*; default: **ansi**) - Frame-relay Local Management Interface type:

- **ansi** - set LMI type to ANSI-617d (also known as Annex A)
- **ccitt** - set LMI type to CCITT Q933a (also known as Annex A)

ignore-dcd (*yes* | *no*; default: **no**) - ignore or not DCD

line-protocol (*cisco-hdlc* | *frame-relay* | *sync-ppp*; default: **sync-ppp**) - line protocol name

mtu (*integer*; default: **1500**) - Maximum Transmit Unit

name (*name*; default: **moxa-c101-N**) - interface name

Notes

If you purchased the MOXA C101 Synchronous card from MikroTik, you have received a V.35 cable with it. This cable should work for all standard modems, which have V.35 connections. For synchronous modems, which have a DB-25 connection, you should use a standard DB-25 cable.

The MikroTik driver for the MOXA C101 Synchronous adapter allows you to unplug the V.35 cable from one modem and plug it into another modem with a different clock speed, and you do not need to restart the interface or router.

Example

```
[admin@MikroTik] interface> moxa-c101
[admin@MikroTik] interface moxa-c101> print
Flags: X - disabled, R - running
  0  R name="moxa-c101-1" mtu=1500 line-protocol=sync-ppp clock-rate=64000
       clock-source=external frame-relay-lmi-type=ansi frame-relay-dce=no
       cisco-hdlc-keepalive-interval=10s ignore-dcd=no
[admin@MikroTik] interface moxa-c101>
```

You can monitor the status of the synchronous interface:

```
[admin@MikroTik] interface moxa-c101> monitor 0
    dtr: yes
    rts: yes
    cts: no
    dsr: no
    dcd: no
[admin@MikroTik] interface moxa-c101>
```

Connect a communication device, e.g., a baseband modem, to the V.35 port and turn it on. If the link is working properly the status of the interface is:

```
[admin@MikroTik] interface moxa-c101> monitor 0
    dtr: yes
    rts: yes
    cts: yes
    dsr: yes
    dcd: yes
[admin@MikroTik] interface moxa-c101>
```

Troubleshooting

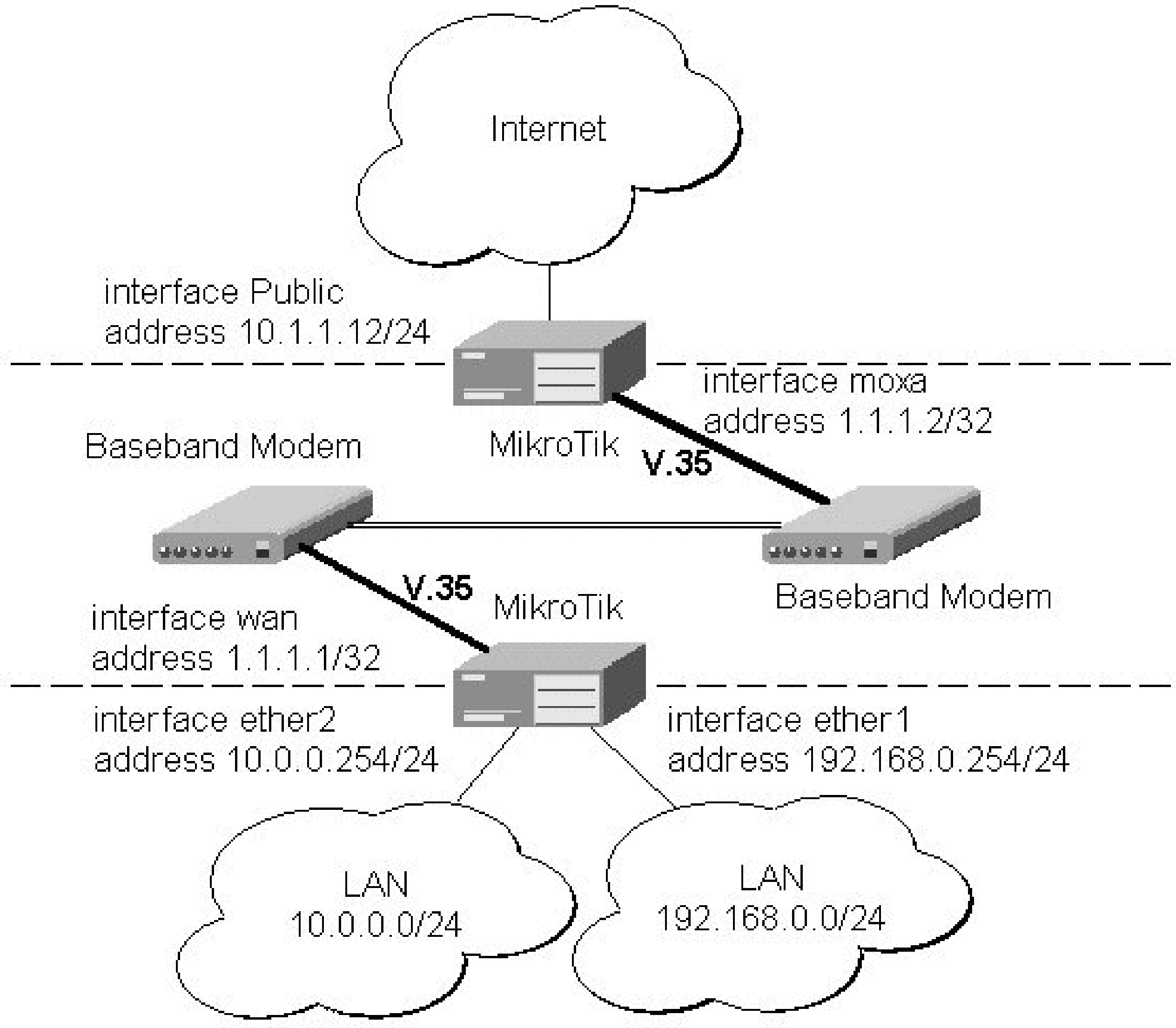
Description

- **The synchronous interface does not show up under the interfaces list**
  Obtain the required license for synchronous feature
- **The synchronous link does not work**
  Check the V.35 cabling and the line between the modems. Read the modem manual

Synchronous Link Application Examples

MikroTik Router to MikroTik Router

Let us consider the following network setup with two MikroTik Routers connected to a leased line with baseband modems:



The driver for MOXA C101 card should be loaded and the interface should be enabled according to the instructions given above. The IP addresses assigned to the synchronous interface should be as follows:

```
[admin@MikroTik] ip address> add address=1.1.1.1/32 interface=wan \
\... network=1.1.1.2 broadcast=255.255.255.255

[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.254/24      10.0.0.254      10.0.0.255      ether2
  1   192.168.0.254/24   192.168.0.254   192.168.0.255   ether1
  2   1.1.1.1/32         1.1.1.2         255.255.255.255 wan
[admin@MikroTik] ip address> /ping 1.1.1.2
1.1.1.2 64 byte ping: ttl=255 time=31 ms
1.1.1.2 64 byte ping: ttl=255 time=26 ms
1.1.1.2 64 byte ping: ttl=255 time=26 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 26/27.6/31 ms
[admin@MikroTik] ip address>
```

The default route should be set to the gateway router 1.1.1.2:

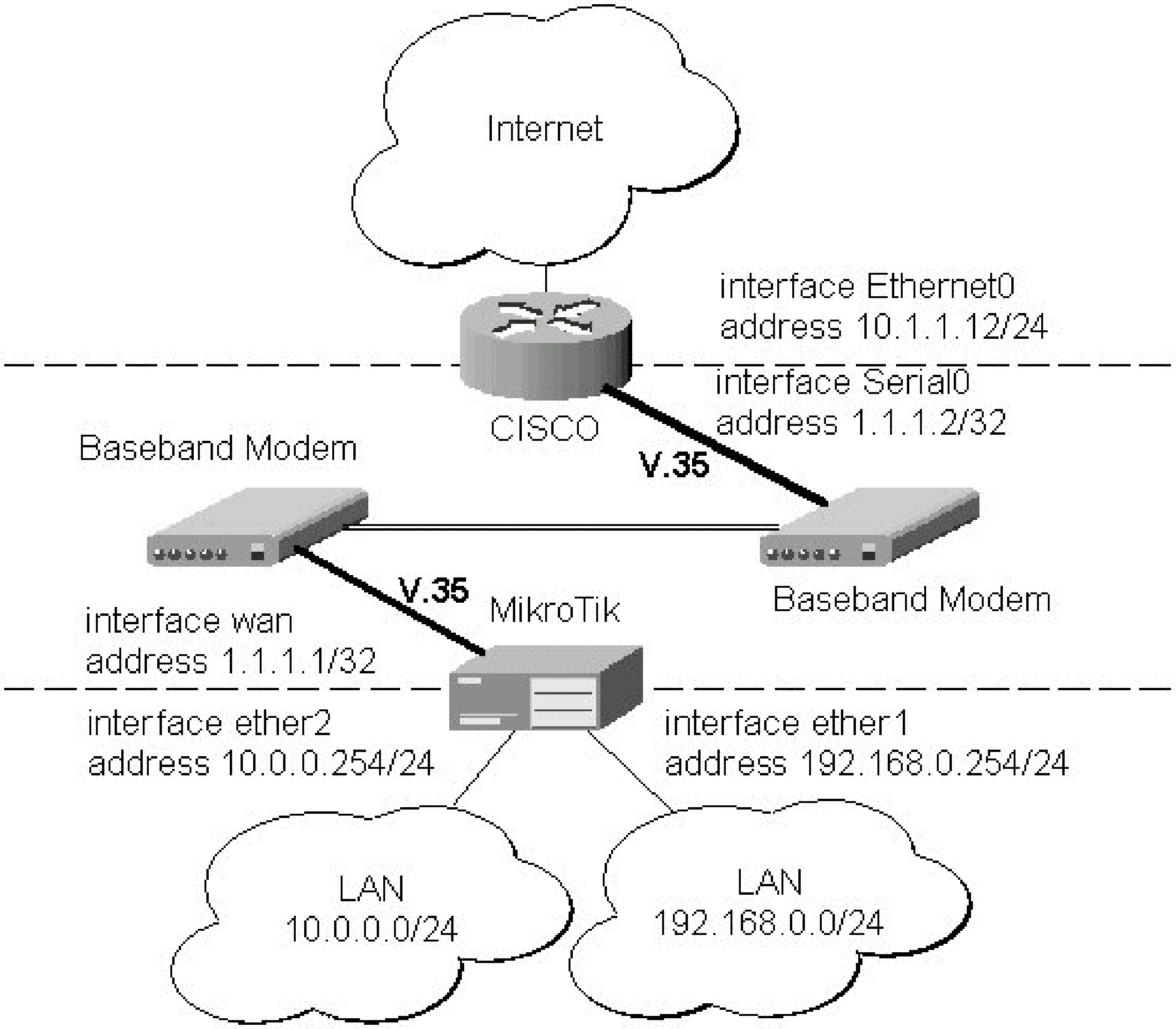
```
[admin@MikroTik] ip route> add gateway 1.1.1.2
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DISTANCE INTER...
 0 A S  0.0.0.0/0                          r 1.1.1.2         1        wan
 1 ADC  10.0.0.0/24        10.0.0.254      r                 0        ether2
 2 ADC  192.168.0.0/24     192.168.0.254   r                 0        ether1
 3 ADC  1.1.1.2/32         1.1.1.1         r                 0        wan
[admin@MikroTik] ip route>
```

The configuration of the MikroTik router at the other end is similar:

```
[admin@MikroTik] ip address> add address=1.1.1.2/32 interface=moxa \
\... network=1.1.1.1 broadcast=255.255.255.255
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.1.1.12/24       10.1.1.12       10.1.1.255      Public
  1   1.1.1.2/32         1.1.1.1         255.255.255.255 moxa
[admin@MikroTik] ip address> /ping 1.1.1.1
1.1.1.1 64 byte ping: ttl=255 time=31 ms
1.1.1.1 64 byte ping: ttl=255 time=26 ms
1.1.1.1 64 byte ping: ttl=255 time=26 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 26/27.6/31 ms
[admin@MikroTik] ip address>
```

MikroTik Router to Cisco Router

Let us consider the following network setup with MikroTik Router connected to a leased line with baseband modems and a CISCO router at the other end:



The driver for MOXA C101 card should be loaded and the interface should be enabled according to the instructions given above. The IP addresses assigned to the synchronous interface should be as follows:

```
[admin@MikroTik] ip address> add address 1.1.1.1/32 interface wan \
\... network 1.1.1.2 broadcast 255.255.255.255
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.254/24      10.0.0.254      10.0.0.255      ether2
  1   192.168.0.254/24   192.168.0.254   192.168.0.255   ether1
  2   1.1.1.1/32         1.1.1.2         255.255.255.255 wan
[admin@MikroTik] ip address> /ping 1.1.1.2
1.1.1.2 64 byte ping: ttl=255 time=31 ms
1.1.1.2 64 byte ping: ttl=255 time=26 ms
1.1.1.2 64 byte ping: ttl=255 time=26 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 26/27.6/31 ms
[admin@MikroTik] ip address>
```

The default route should be set to the gateway router 1.1.1.2:

```
[admin@MikroTik] ip route> add gateway 1.1.1.2
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DISTANCE INTER...
 0 A S  0.0.0.0/0                          r 1.1.1.2         1        wan
 1 ADC  10.0.0.0/24        10.0.0.254      r                 0        ether2
 2 ADC  192.168.0.0/24     192.168.0.254   r                 0        ether1
 3 ADC  1.1.1.2/32         1.1.1.1         r                 0        wan
[admin@MikroTik] ip route>
```

The configuration of the Cisco router at the other end (part of the configuration) is:

```
CISCO#show running-config
Building configuration...

Current configuration:
...
!
interface Ethernet0
 description connected to EthernetLAN
 ip address 10.1.1.12 255.255.255.0
!
interface Serial0
 description connected to MikroTik
 ip address 1.1.1.2 255.255.255.252
 serial restart-delay 1
!
ip classless
ip route 0.0.0.0 0.0.0.0 10.1.1.254
!
...
end

CISCO#
```

Send ping packets to the MikroTik router:

```
CISCO#ping 1.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/32/40 ms
CISCO#
```

Note! Keep in mind that for the point-to-point link the network mask is set to **32** bits, the argument **network** is set to the IP address of the other end, and the broadcast address is set to **255.255.255.255**.

MOXA C502 Dual-port Synchronous Interface

Document revision 1.3 (February 6, 2008, 2:58 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS supports the MOXA C502 PCI Dual-port Synchronous 8Mb/s Adapter hardware. The V.35 synchronous interface is the standard for VSAT and other satellite modems. However, you must check with the satellite system supplier for the modem interface type.

Specifications

Packages required: ***synchronous***
License required: ***level4***
Home menu level: ***/interface moxa-c502***
Standards and Technologies: ***Cisco/HDLC-X.25 (RFC 1356)***, ***Frame Relay (RFC1490)***, ***PPP (RFC-1661)***, ***PPP (RFC-1662)***
Hardware usage: ***Not significant***

Description

You can install up to four MOXA C502 synchronous cards in one PC box, if you have so many PCI slots available. Assuming you have all necessary packages and licences installed, in most cases it should to be done nothing at that point (all drivers are loaded automatically).

Additional Documents

For more information about the MOXA C502 Dual-port Synchronous 8Mb/s Adapter hardware please see:

- http://www.moxa.com/product/sync/C502.htm - the product on-line documentation
- C502 Dual Port Sync Board User's Manuall the user's manual in PDF format

Synchronous Interface Configuration

Home menu level: ***/interface moxa-c502***

Description

Moxa c502 synchronous interface is shown under the interfaces list with the name moxa-c502-N

Property Description

cisco-hdlc-keepalive-interval (*time*; default: **10s**) - keepalive period in seconds

clock-rate (*integer*; default: **64000**) - speed of internal clock

clock-source (*external* | *internal* | *tx-from-rx* | *tx-internal*; default: **external**) - clock source

frame-relay-dce (*yes* | *no*; default: **no**) - operate or not in DCE mode

frame-relay-lmi-type (*ansi* | *ccitt*; default: **ansi**) - Frame-relay Local Management Interface type:

- **ansi** - set LMI type to ANSI-617d (also known as Annex A)
- **ccitt** - set LMI type to CCITT Q933a (also known as Annex A)

ignore-dcd (*yes* | *no*; default: **no**) - ignore or not DCD

line-protocol (*cisco-hdlc* | *frame-relay* | *sync-ppp*; default: **sync-ppp**) - line protocol name

mtu (*integer*; default: **1500**) - Maximum Transmit Unit

name (*name*; default: **moxa-c502-N**) - interface name

Notes

There will be TWO interfaces for each MOXA C502 card since the card has TWO ports.

The MikroTik driver for the MOXA C502 Dual Synchronous adapter allows you to unplug the V.35 cable from one modem and plug it into another modem with a different clock speed, and you do not need to restart the interface or router.

Example

```
[admin@MikroTik] interface> moxa-c502
[admin@MikroTik] interface moxa-c502> print
Flags: X - disabled, R - running
  0  R name="moxa-c502-1" mtu=1500 line-protocol=sync-ppp clock-rate=64000
       clock-source=external frame-relay-lmi-type=ansi frame-relay-dce=no
       cisco-hdlc-keepalive-interval=10s
  1  R name="moxa-c502-2" mtu=1500 line-protocol=sync-ppp clock-rate=64000
       clock-source=external frame-relay-lmi-type=ansi frame-relay-dce=no
       cisco-hdlc-keepalive-interval=10s
```

```
[admin@MikroTik] interface moxa-c502>
```

You can monitor the status of the synchronous interface:

```
[admin@MikroTik] interface moxa-c502> monitor 0
    dtr: yes
    rts: yes
    cts: no
    dsr: no
    dcd: no
[admin@MikroTik] interface moxa-c502>
```

Connect a communication device, e.g., a baseband modem, to the V.35 port and turn it on. If the link is working properly the status of the interface is:

```
[admin@MikroTik] interface moxa-c502> monitor 0
    dtr: yes
    rts: yes
    cts: yes
    dsr: yes
    dcd: yes
[admin@MikroTik] interface moxa-c502>
```

Troubleshooting

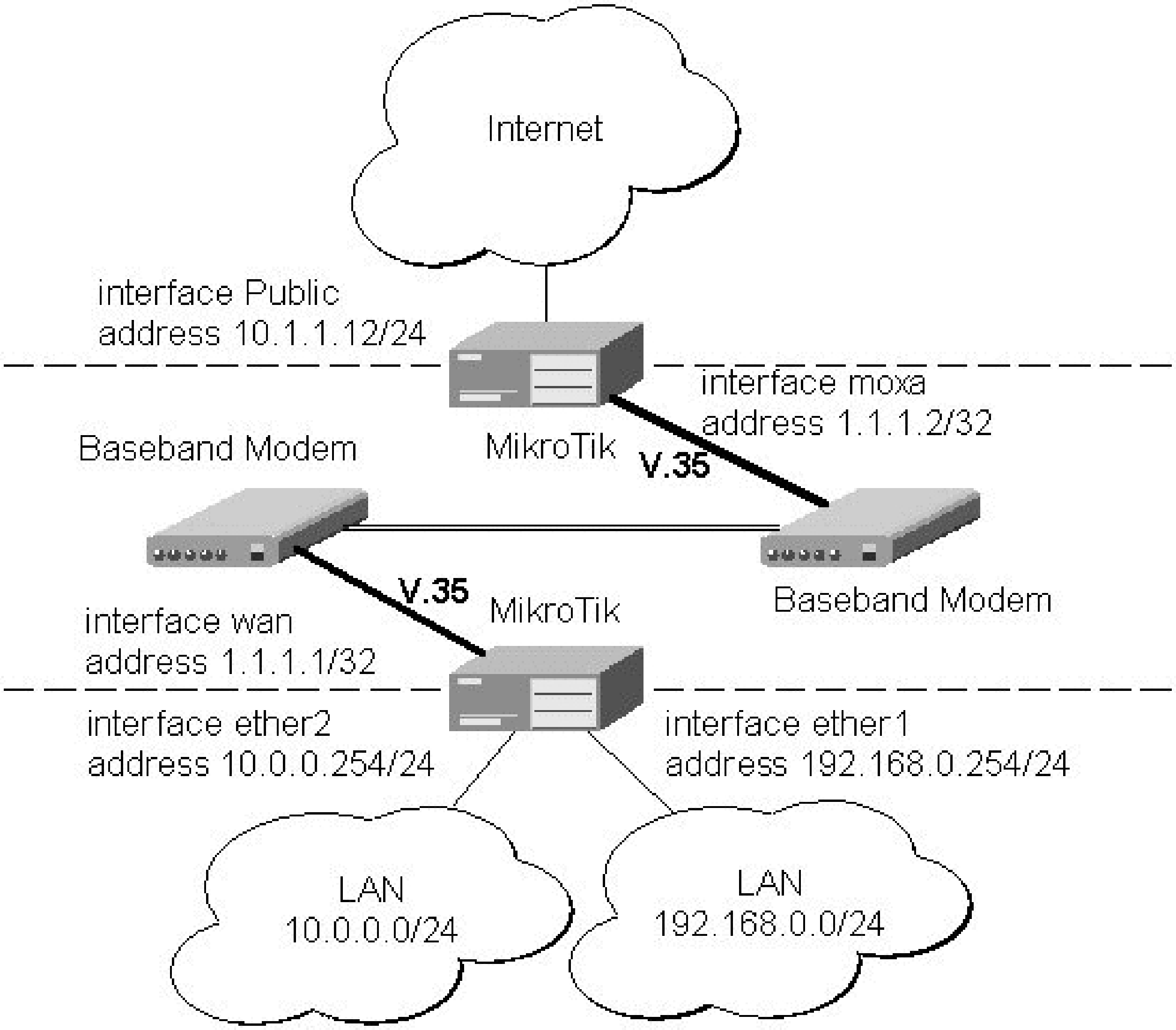
Description

- **The synchronous interface does not show up under the interfaces list**
  Obtain the required license for synchronous feature
- **The synchronous link does not work**
  Check the V.35 cabling and the line between the modems. Read the modem manual

Synchronous Link Application Examples

MikroTik Router to MikroTik Router

Let us consider the following network setup with two MikroTik Routers connected to a leased line with baseband modems:



The driver for MOXA C502 card should be loaded and the interface should be enabled according to the instructions given above. The IP addresses assigned to the synchronous interface should be as follows:

```
[admin@MikroTik] ip address> add address=1.1.1.1/32 interface=wan \
\... network=1.1.1.2 broadcast=255.255.255.255
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.254/24      10.0.0.254      10.0.0.255      ether2
  1   192.168.0.254/24   192.168.0.254   192.168.0.255   ether1
  2   1.1.1.1/32         1.1.1.2         255.255.255.255 wan
[admin@MikroTik] ip address> /ping 1.1.1.2
1.1.1.2 64 byte ping: ttl=255 time=31 ms
1.1.1.2 64 byte ping: ttl=255 time=26 ms
1.1.1.2 64 byte ping: ttl=255 time=26 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 26/27.6/31 ms
[admin@MikroTik] ip address>
```

The default route should be set to the gateway router 1.1.1.2:

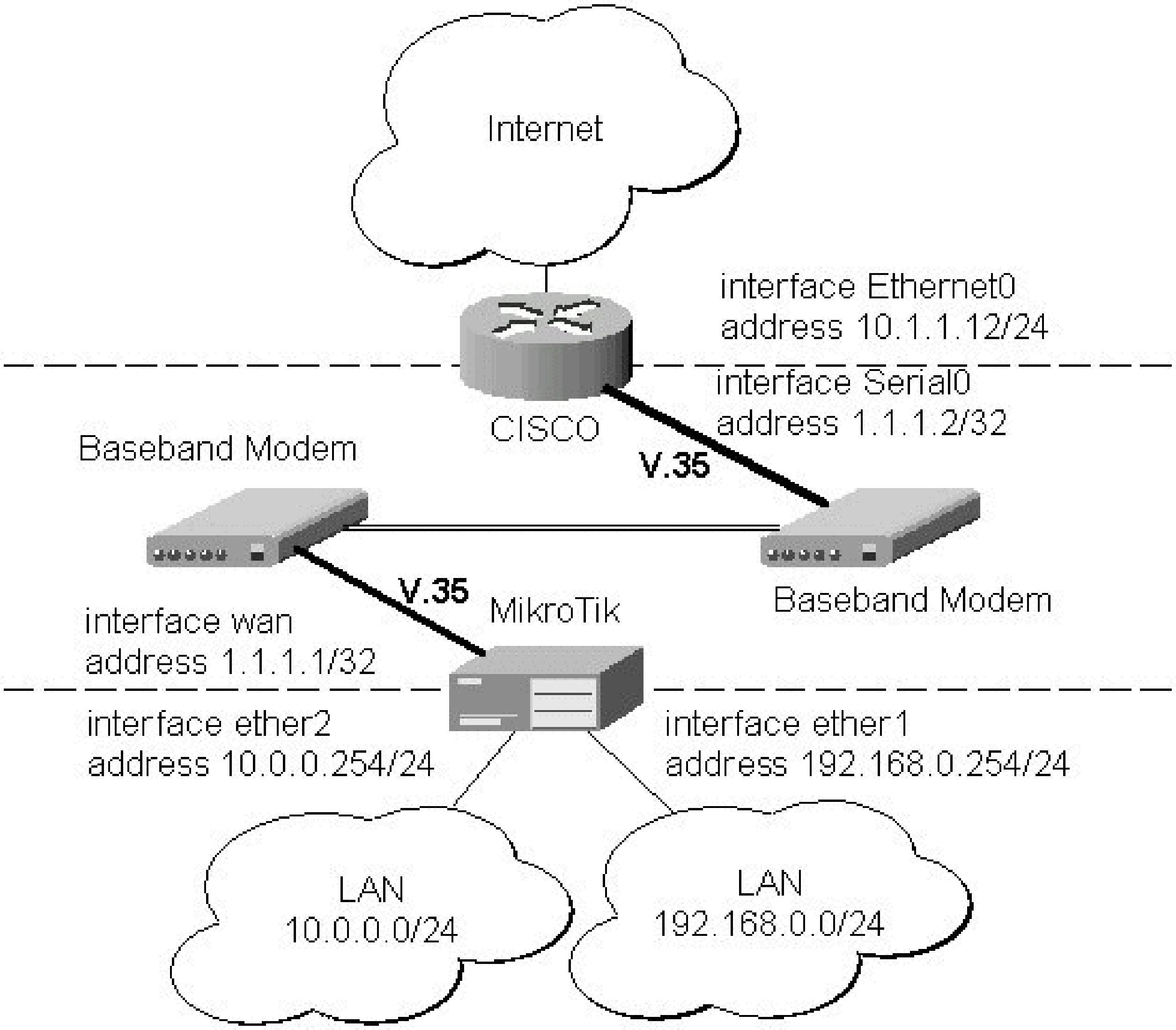
```
[admin@MikroTik] ip route> add gateway=1.1.1.2 interface=wan
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DISTANCE INTER...
 0 A S  0.0.0.0/0                          r 1.1.1.2         1        wan
 1 ADC  10.0.0.0/24        10.0.0.254      r                 0        ether2
 2 ADC  192.168.0.0/24     192.168.0.254   r                 0        ether1
 3 ADC  1.1.1.2/32         1.1.1.1         r                 0        wan
[admin@MikroTik] ip route>
```

The configuration of the MikroTik router at the other end is similar:

```
[admin@MikroTik] ip address> add address=1.1.1.2/32 interface=moxa \
\... network=1.1.1.1 broadcast=255.255.255.255
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.1.1.12/24       10.1.1.12       10.1.1.255      Public
  1   1.1.1.2/32         1.1.1.1         255.255.255.255 moxa
[admin@MikroTik] ip address> /ping 1.1.1.1
1.1.1.1 64 byte ping: ttl=255 time=31 ms
1.1.1.1 64 byte ping: ttl=255 time=26 ms
1.1.1.1 64 byte ping: ttl=255 time=26 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 26/27.6/31 ms
[admin@MikroTik] ip address>
```

MikroTik Router to Cisco Router

Let us consider the following network setup with MikroTik Router connected to a leased line with baseband modems and a CISCO router at the other end:



The driver for MOXA C502 card should be loaded and the interface should be enabled according to the instructions given above. The IP addresses assigned to the synchronous interface should be as follows:

```
[admin@MikroTik] ip address> add address=1.1.1.1/32 interface=wan \
\... network=1.1.1.2 broadcast=255.255.255.255
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.254/24      10.0.0.254      10.0.0.255      ether2
  1   192.168.0.254/24   192.168.0.254   192.168.0.255   ether1
  2   1.1.1.1/32         1.1.1.2         255.255.255.255 wan
[admin@MikroTik] ip address> /ping 1.1.1.2
1.1.1.2 64 byte ping: ttl=255 time=31 ms
1.1.1.2 64 byte ping: ttl=255 time=26 ms
1.1.1.2 64 byte ping: ttl=255 time=26 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 26/27.6/31 ms
[admin@MikroTik] ip address>
```

The default route should be set to the gateway router 1.1.1.2:

```
[admin@MikroTik] ip route> add gateway 1.1.1.2
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DISTANCE INTER...
 0 A S  0.0.0.0/0                          r 1.1.1.2         1        wan
 1 ADC  10.0.0.0/24        10.0.0.254      r                 0        ether2
 2 ADC  192.168.0.0/24     192.168.0.254   r                 0        ether1
 3 ADC  1.1.1.2/32         1.1.1.1         r                 0        wan
[admin@MikroTik] ip route>
```

The configuration of the Cisco router at the other end (part of the configuration) is:

```
CISCO#show running-config
Building configuration...

Current configuration:
...
!
interface Ethernet0
 description connected to EthernetLAN
 ip address 10.1.1.12 255.255.255.0
!
interface Serial0
 description connected to MikroTik
 ip address 1.1.1.2 255.255.255.252
 serial restart-delay 1
!
ip classless
ip route 0.0.0.0 0.0.0.0 10.1.1.254
!
...
end

CISCO#
```

Send ping packets to the MikroTik router:

```
CISCO#ping 1.1.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/32/40 ms
CISCO#
```

Note! Keep in mind that for the point-to-point link the network mask is set to **32** bits, the argument **network** is set to the IP address of the other end, and the broadcast address is set to **255.255.255.255**.

PPP and Asynchronous Interfaces

Document revision 1.3 (October 31, 2007, 13:15 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

PPP (Point-to-Point Protocol) provides a method for transmitting datagrams over serial point-to-point links. Physically it relies on **COM1** and **COM2** ports from standard PC hardware configurations. These appear as **serial0** and **serial1** automatically. You can add more serial ports to use the router for a modem pool using these adapters:

- MOXA (http://www.moxa.com) Smartio CP-132 2-port PCI multiport asynchronous board with maximum of 8 ports (4 cards)
- MOXA (http://www.moxa.com) Smartio C104H, CP-114 or CT-114 4-port PCI multiport asynchronous board with maximum of 16 ports (4 cards)
- MOXA (http://www.moxa.com) Smartio C168H, CP-168H or CP-168U 8-port PCI multiport asynchronous board with maximum of 32 ports (4 cards)
- Cyclades (http://www.cyclades.com) Cyclom-Y Series 4 to 32 port PCI multiport asynchronous board with maximum of 128 ports (4 cards)

- Cyclades (http://www.cyclades.com) Cyclades-Z Series 16 to 64 port PCI multiport asynchronous board with maximum of 256 ports (4 cards)
- TCL (http://www.thetcl.com) DataBooster 4 or 8 port High Speed Buffered PCI Communication Controllers

Specifications

Packages required: ***ppp***
License required: ***level1***
Home menu level: ***/interface ppp-client, /interface ppp-server***
Standards and Technologies: ***PPP (RFC 1661)***
Hardware usage: ***Not significant***

Additional Documents

- http://www.ietf.org/rfc/rfc2138.txt?number=2138
- http://www.ietf.org/rfc/rfc2138.txt?number=2139

Serial Port Configuration

Home menu level: ***/port***

Property Description

baud-rate (*integer*; default: **9600**) - data rate of the port

data-bits (*7* | *8*; default: **8**) - number of bits per character transmitted

flow-control (*none* | *hardware* | *xon-xoff*; default: **hardware**) - flow control method

name (*name*; default: **serialN**) - port name

parity (*none* | *even* | *odd*; default: **none**) - character parity check method

stop-bits (*1* | *2*; default: **1**) - number of stop bits after each character transmitted

used-by (*read-only: text*) - shows the user (if any) of the port. Only unused ports can be used in PPP setup

Notes

Keep in mind that **baud-rate**, **data-bits**, **parity**, **stop-bits** and **flow control** parameters must be the same for both communicating parties.

Example

```
[admin@MikroTik] > /port print
  # NAME                                  USED-BY                           BAUD-RATE
  0 serial0                               Serial Console                    9600
  1 databooster1                                                            9600
  2 databooster2                                                            9600
  3 databooster3                                                            9600
  4 databooster4                                                            9600
  5 databooster5                                                            9600
```

```
   6 databooster6                                                     9600
   7 databooster7                                                     9600
   8 databooster8                                                     9600
   9 cycladesA1                                                       9600
  10 cycladesA2                                                       9600
  11 cycladesA3                                                       9600
  12 cycladesA4                                                       9600
  13 cycladesA5                                                       9600
  14 cycladesA6                                                       9600
  15 cycladesA7                                                       9600
  16 cycladesA8                                                       9600
 [admin@MikroTik] > set 9 baud-rate=38400
 [admin@MikroTik] >
```

PPP Server Setup

Home menu level: ***/interface ppp-server***

Description

PPP server provides a remode connection service for users. When dialing in, the users can be authenticated locally using the local user database in the **/user** menu, or at the RADIUS server specified in the **/ip ppp** settings.

Property Description

authentication (*multiple choice: mschap2, mschap1, chap, pap*; default: **mschap2, mschap1, chap, pap**) - authentication protocol(s)

max-mru (*integer*; default: **1500**) - maximum value of MRU (Maximum Receive Unit) allowed on this link. Largest packet that can be received

max-mtu (*integer*; default: **1500**) - maximum value of MTU (Maximum Transmission Unit) allowed on this link. Maximum packet size to be transmitted

modem-init (*text*; default: **""**) - modem initialization string. For example, you may use "s11=40" to improve dialing speed

mrru (*integer*) - maximum packet size that can be received on the link. If packet is bigger than tunnel MTU, it will be split into multiple packets. That way it is possible to send full size (1500 or even 1514) packets over PPTP or L2TP tunnels.

- **disabled** - disable MRRU on this link

name (*name*; default: **ppp-inN**) - interface name for reference

null-modem (*no* | *yes*; default: **no**) - enable/disable null-modem mode (when enabled, no modem initialization strings are sent)

port (*name*) - serial port name

profile (*name*; default: **default**) - default (fall-back) profile name used for the link

ring-count (*integer*; default: **1**) - number of rings to wait before answering phone

Notes

Specifying MRRU means enabling MP (Multilink PPP) over single link. This protocol is used to split big packets into smaller ones. Under Windows it can be enabled in Networking tag, Settings button, "Negotiate multi-link for single link connections". Their MRRU is hardcoded to 1614. This setting is usefull to overcome PathMTU discovery failures. The MP should be enabled on both peers.

Example

You can add a PPP server using the **add** command:

```
[admin@MikroTik] interface ppp-server> add name=test port=serial1
[admin@MikroTik] interface ppp-server> print
Flags: X - disabled, R - running
  0 X  name="test" max-mtu=1500 max-mru=1500 mrru=disabled port=serial1
       authentication=pap,chap,mschap1,mschap2 profile=default modem-init=""
       ring-count=1 null-modem=no
[admin@MikroTik] interface ppp-server> enable 0
[admin@MikroTik] interface ppp-server> monitor test
            status: "waiting for call..."

[admin@MikroTik] interface ppp-server>
```

PPP Client Setup

Home menu level: ***/interface ppp-client***

Description

The section describes PPP clients configuration routines.

Property Description

add-default-route (*yes* | *no*; default: **no**) - add PPP remote address as a default route

allow (*multiple choice: mschap2*, *mschap1*, *chap*, *pap*; default: **mschap2, mschap1, chap, pap**) - the protocol to allow the client to use for authentication

dial-command (*text*; default: **"ATDT"**) - AT dial command to use. The default one sets tone diling mode

dial-on-demand (*yes* | *no*; default: **no**) - enable/disable dial on demand

max-mru (*integer*; default: **1500**) - maximum value of MRU (Maximum Receive Unit) allowed on this link. Largest packet that can be received

max-mtu (*integer*; default: **1500**) - maximum value of MTU (Maximum Transmission Unit) allowed on this link. Maximum packet size to be transmitted

modem-init (*text*; default: **""**) - modem initialization strings. You may use "s11=40" to improve dialing speed

mrru (*integer*) - maximum packet size that can be received on the link. If packet is bigger than tunnel MTU, it will be split into multiple packets. That way it is possible to send full size (1500 or even 1514) packets over PPTP or L2TP tunnels.

- **disabled** - disable MRRU on this link

name (*name*; default: **ppp-inN**) - interface name for reference

null-modem (*no* | *yes*; default: **no**) - enable/disable null-modem mode (when enabled, no modem initialization strings are sent)

password (*text*; default: **""**) - P2P user password on the remote server to use for dialout

phone (*integer*; default: **""**) - phone number for dialout

port (*name*) - serial port

profile (*name*; default: **default**) - local profile to use for dialout

use-peer-dns (*yes* | *no*; default: **no**) - use DNS server settings from the remote server

user (*text*; default: "") - P2P user name on the remote server to use for dialout

Notes

Additional client profiles must be configured on the server side for clients to accomplish logon procedure. For more information see **Related Documents** section.

PPP client profiles must match at least partially (**local-address** and values related to encryption should match) with corresponding remote server values.

Example

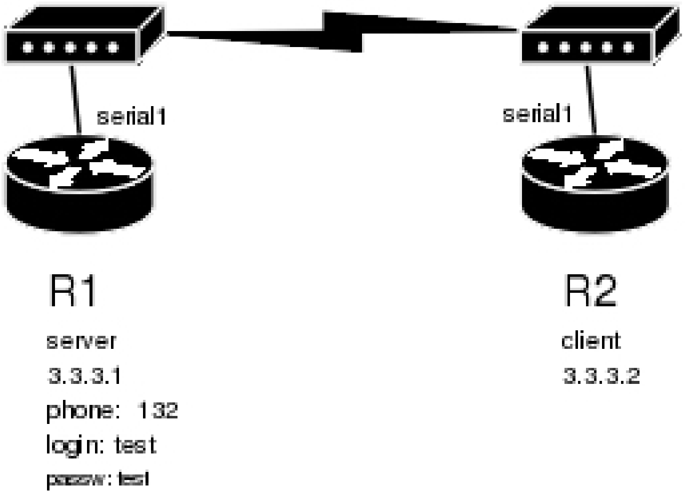
You can add a PPP client using the **add** command:

```
[admin@MikroTik] interface ppp-client> add name=test user=test port=serial1 \
\... add-default-route=yes
[admin@MikroTik] interface ppp-client> print
Flags: X - disabled, R - running
  0 X  name="test" mtu=1500 mru=1500 port=serial1 user="test" password=""
       profile=default phone="" tone-dial=yes modem-init="" null-modem=no
       dial-on-demand=no add-default-route=yes use-peer-dns=no

[admin@MikroTik] interface ppp-client> enable 0
[admin@MikroTik] interface ppp-client> monitor test
[admin@MikroTik] interface ppp-client> monitor 0
           status: "dialing out..."

[admin@MikroTik] interface ppp-client>
```

PPP Application Example

Client - Server Setup

In this example we will consider the following network setup:



For a typical server setup we need to add one user to the **R1** and configure the PPP server.

```
[admin@MikroTik] ppp secret> add name=test password=test local-address=3.3.3.1 \
\... remote-address=3.3.3.2
[admin@MikroTik] ppp secret> print
Flags: X - disabled
  0   name="test" service=any caller-id="" password="test" profile=default
      local-address=3.3.3.1 remote-address=3.3.3.2 routes=""

[admin@MikroTik] ppp secret> /int ppp-server
[admin@MikroTik] interface ppp-server> add port=serial1 disabled=no
[admin@MikroTik] interface ppp-server> print
Flags: X - disabled, R - running
  0    name="ppp-in1" mtu=1500 mru=1500 port=serial1
       authentication=mschap2,mschap1,chap,pap profile=default modem-init=""
       ring-count=1 null-modem=no

[admin@MikroTik] interface ppp-server>
```

Now we need to setup the client to connect to the server:

```
[admin@MikroTik] interface ppp-client> add port=serial1 user=test password=test \
\... phone=132
[admin@MikroTik] interface ppp-client> print
Flags: X - disabled, R - running
  0 X  name="ppp-out1" mtu=1500 mru=1500 port=serial1 user="test"
       password="test" profile=default phone="132" tone-dial=yes
       modem-init="" null-modem=no dial-on-demand=no add-default-route=no
       use-peer-dns=no

[admin@MikroTik] interface ppp-client> enable 0

After a short duration of time the routers will be able to ping each other:
[admin@MikroTik] interface ppp-client> /ping 3.3.3.1
3.3.3.1 64 byte ping: ttl=64 time=43 ms
3.3.3.1 64 byte ping: ttl=64 time=11 ms
3.3.3.1 64 byte ping: ttl=64 time=12 ms
3.3.3.1 64 byte ping: ttl=64 time=11 ms
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 11/19.2/43 ms
[admin@MikroTik] interface ppp-client>
```

RadioLAN 5.8GHz Wireless Interface

Document revision 1.2 (February 6, 2008, 2:56 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS supports the following RadioLAN 5.8GHz Wireless Adapter hardware:

- RadioLAN ISA card (Model 101)
- RadioLAN PCMCIA card

For more information about the RadioLAN adapter hardware please see the relevant User???s Guides and Technical Reference Manuals.

Specifications

Packages required: ***radiolan***
License required: ***level4***
Home menu level: ***/interface radiolan***
Hardware usage: ***Not significant***

Description

Installing the Wireless Adapter

These installation instructions apply to non-Plug-and-Play ISA cards. If You have a Plug-and-Play compliant system AND **PnP OS Installed** option in system BIOS is set to **Yes** AND you have a Plug-and-Play compliant ISA or PCI card (using PCMCIA or CardBus card with Plug-and-Play compliant adapter), the driver should be loaded automatically. If it is not, these instructions may also apply to your system.

The basic installation steps of the wireless adapter should be as follows:

1. Check the system BIOS settings for peripheral devices, like, Parallel or Serial communication ports. Disable them, if you plan to use IRQ's assigned to them by the BIOS.
2. Use the RLProg.exe to set the IRQ and Base Port address of the RadioLAN ISA card (Model 101). RLProg must not be run from a DOS window. Use a separate computer or a bootable floppy to run the RLProg utility and set the hardware parameters. The factory default values of I/O 0x300 and IRQ 10 might conflict with other devices.

Please note, that not all combinations of I/O base addresses and IRQs may work on your motherboard. As it has been observed, the IRQ 5 and I/O 0x300 work in most cases.

Wireless Interface Configuration

Home menu level: ***/interface ratiolan***

Description

To set the wireless interface for working with another wireless card in a point-to-point link, you should set the following parameters:

- The **Service Set Identifier.** It should match the sid of the other card.
- The **Distance** should be set to that of the link. For example, if you have 6 km link, use distance 4.7 km - 6.6 km.

All other parameters can be left as default. You can monitor the list of neighbors having the same sid and being within the radio range.

Property Description

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; default: **enabled**) - Address Resolution Protocol, one of the:

- **disabled** - the interface will not use ARP protocol
- **enabled** - the interface will use ARP protocol
- **proxy-arp** - the interface will be an ARP proxy (see corresponding manual)
- **reply-only** - the interface will only reply to the requests originated to its own IP addresses, but neighbor MAC addresses will be gathered from /ip arp statically set table only.

card-name (*text*) - card name

default-address (*MAC address*; default: **00:00:00:00:00:00**) - MAC address of a host in the radio network where to send the packet, if it is for none of the radio clients

default-destination (*ap* | *as-specified* | *first-ap* | *first-client* | *no-destination*; default: **first-client**) - default destination. It sets the destination where to send the packet if it is not for a client in the radio network

distance (*0-150m* | *10.2km-13.0km* | *2.0km-2.9km* | *4.7km-6.6km* | *1.1km-2.0km* | *150m-1.1km* | *2.9km-4.7km* | *6.6km-10.2km*; default: **0-150m**) - distance setting for the link

mac-address (*read-only: MAC address*) - MAC address

max-retries (*integer*; default: **1500**) - maximum retries before dropping the packet

mtu (*integer*; default: **1500**) - Maximum Transmission Unit

name (*name*; default: **radiolanN**) - assigned interface name

rx-diversity (*enabled | disabled*; default: **disabled**) - receive diversity

sid (*text*) - Service Identifier

tx-diversity (*enabled | disabled*; default: **disabled**) - transmit diversity

Example

```
[admin@MikroTik] interface radiolan> print
Flags: X - disabled, R - running
  0  R name="radiolan1" mtu=1500 mac-address=00:A0:D4:20:4B:E7 arp=enabled
       card-name="00A0D4204BE7" sid="bbbb" default-destination=first-client
       default-address=00:00:00:00:00:00 distance=0-150m max-retries=15
       tx-diversity=disabled rx-diversity=disabled
[admin@MikroTik] interface radiolan>
```

You can monitor the status of the wireless interface:

```
[admin@MikroTik] interface radiolan> monitor radiolan1
    default: 00:00:00:00:00:00
      valid: no
[admin@MikroTik] interface radiolan>
```

Here, the wireless interface card has not found any neighbor.

```
[admin@MikroTik] interface radiolan> set 0 sid ba72 distance 4.7km-6.6km
[admin@MikroTik] interface radiolan> print
Flags: X - disabled, R - running
  0  R name="radiolan1" mtu=1500 mac-address=00:A0:D4:20:4B:E7 arp=enabled
       card-name="00A0D4204BE7" sid="ba72" default-destination=first-client
       default-address=00:00:00:00:00:00 distance=4.7km-6.6km max-retries=15
       tx-diversity=disabled rx-diversity=disabled
[admin@MikroTik] interface radiolan> monitor 0
    default: 00:A0:D4:20:3B:7F
      valid: yes
[admin@MikroTik] interface radiolan>
```

Now we'll monitor other cards with the same **sid** within range:

```
[admin@MikroTik] interface radiolan> neighbor radiolan1 print
Flags: A - access-point, R - registered, U - registered-to-us,
D - our-default-destination
      NAME                 ADDRESS            ACCESS-POINT
    D 00A0D4203B7F         00:A0:D4:20:3B:7F
[admin@MikroTik] interface radiolan>
```

You can test the link by pinging the neighbor by its MAC address:

```
[admin@MikroTik] interface radiolan> ping 00:a0:d4:20:3b:7f radiolan1 \
\... size=1500 count=50
                  sent: 1
    successfully-sent: 1
          max-retries: 0
      average-retries: 0
          min-retries: 0

                  sent: 11
    successfully-sent: 11
          max-retries: 0
      average-retries: 0
          min-retries: 0
```

```
                 sent: 21
    successfully-sent: 21
          max-retries: 0
      average-retries: 0
          min-retries: 0

                 sent: 31
    successfully-sent: 31
          max-retries: 0
      average-retries: 0
          min-retries: 0

                 sent: 41
    successfully-sent: 41
          max-retries: 0
      average-retries: 0
          min-retries: 0

                 sent: 50
    successfully-sent: 50
          max-retries: 0
      average-retries: 0
          min-retries: 0

[admin@MikroTik] interface radiolan>
```

Troubleshooting

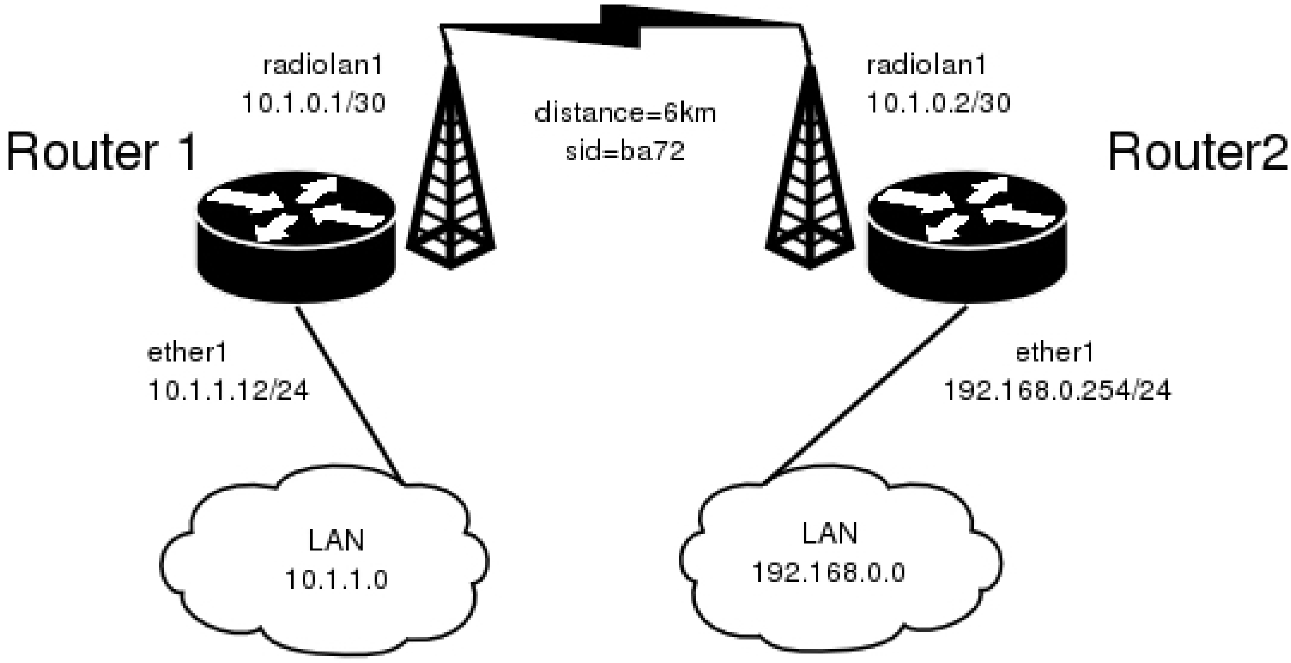
Description

- **The radiolan interface does not show up under the interfaces list**
  Obtain the required license for RadioLAN 5.8GHz wireless feature
- **The wireless card does not obtain the MAC address of the default destination**
  Check the cabling and antenna alignment

Wireless Network Applications

Point-to-Point Setup with Routing

Let us consider the following network setup:



The minimum configuration required for the RadioLAN interfaces of both routers is:

1. Setting the Service Set Identifier (up to alphanumeric characters). In our case we use SSID "ba72"
2. Setting the distance parameter, in our case we have 6km link.

The IP addresses assigned to the wireless interface of Router#1 should be from the network 10.1.0.0/30, e.g.:

```
[admin@MikroTik] ip address> add address=10.1.0.1/30 interface=radiolan1
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.1.1.12/24       10.1.1.0        10.1.1.255      ether1
  1   10.1.0.1/30        10.1.0.0        10.1.0.3        radiolan1
[admin@MikroTik] ip address>
```

The default route should be set to the gateway router 10.1.1.254. A static route should be added for the network 192.168.0.0/24:

```
[admin@MikroTik] ip route> add gateway=10.1.1.254
comment  copy-from  disabled  distance  dst-address  netmask  preferred-source
[admin@MikroTik] ip route> add gateway=10.1.1.254 preferred-source=10.1.0.1
[admin@MikroTik] ip route> add dst-address=192.168.0.0/24 gateway=10.1.0.2 \
\... preferred-source=10.1.0.1
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DISTANCE INTER...
 0 A S  0.0.0.0/0                          u 10.1.1.254      1        radiolan1
 1 A S  192.168.0.0/24                     r 10.1.0.2        1        radiolan1
 2 ADC  10.1.0.0/30        10.1.0.1        r 0.0.0.0         0        radiolan1
 3 ADC  10.1.1.0/24        10.1.1.12       r 0.0.0.0         0        ether1
[admin@MikroTik] ip route>
```

The Router#2 should have addresses 10.1.0.2/30 and 192.168.0.254/24 assigned to the radiolan and

Ethernet interfaces respectively. The default route should be set to 10.1.0.1

Sangoma Synchronous Cards

Document revision 0.5 (February 6, 2008, 2:56 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS supports the following Sangoma Technologies WAN adapters:

- Sangoma S5141 (dual-port) and S5142 (quad-port) PCI RS232/V.35/X.21 (4Mbit/s - primary port and 512Kbit/s - secondary ones)
- Sangoma S5148 (single-port) and S5147 (dual-port) PCI E1/T1

Specifications

Packages required: ***synchronous***
License required: ***level4***
Home menu level: ***/interface sangoma***
Standards and Technologies: ***X.21, V.35, T1/E1/G.703, Frame Relay, PPP, Cisco-HDLC***
Hardware usage: ***Not significant***

Synchronous Interface Configuration

Home menu level: ***/interface sangoma***

Description

With the introduction of 2.8 release, MikroTik RouterOS supports wide range of Sangoma Technologies WANPIPE cards. These cards provide a router with the ability to communicate over T1, E1, RS232, V.35 and X.21 links directly, without the need of external CSU/DSU equipment.

Property Description

active-channels (*all* | *integer*; default: **all**) - for T1/E1 channels only. Specifies active E1/T1 channel set

chdlc-keepalive (*time*; default: **10s**) - Cisco-HDLC keepalive interval in seconds

clock-rate (*integer*; default: **64000**) - internal clock rate in bps

clock-source (*internal* | *external*; default: **external**) - specifies whether the card should rely on supplied clock or generate its own

frame-relay-dce (*yes* | *no*; default: **no**) - specifies whether the device operates in Data Communication Equipment mode. The value yes is suitable only for T1 models

frame-relay-lmi-type (*ansi* | *ccitt*; default: **ansi**) - Frame Relay Line Management Interface Protocol type

framing mode (*CRC4* | *D4* | *ESF* | *ESF-JAPAN* | *Non-CRC4* | *Unframed*; default: **ESF**) - for T1/E1 channels only. The frame mode:

- **CRC4** - Cyclic Redundancy Check 4-bit (E1 Signaling, Europe)
- **D4** - Fourth Generation Channel Bank (48 Voice Channels on 2 T-1s or 1 T-1c)
- **ESF** - Extended Superframe Format
- **Non-CRC4** - plain Cyclic Redundancy Check
- **Unframed** - do not check frame integrity

line-build-out (*0dB* | *7.5dB* | *15dB* | *22.5dB* | *110ft* | *220ft* | *330ft* | *440ft* | *550ft* | *660ft* | *E1-75* | *E1-120*; default: **0dB**) - for T1/E1 channels only. Line Build Out Signal Level.

line-code (*AMI* | *B8ZS* | *HDB3*; default: **B8ZS**) - for T1/E1 channels only. Line modulation method:

- **AMI** - Alternate Mark Inversion
- **B8ZS** - Binary 8-Zero Substitution
- **HDB3** - High Density Bipolar 3 Code (ITU-T)

line-protocol (*cisco-hdlc* | *frame-relay* | *sync-ppp*; default: **sync-ppp**) - line protocol

media-type (*E1* | *T1* | *RS232* | *V35*; default: **V35**) - the hardware media used for this interface

mtu (*integer*; default: **1500**) - Maximum Transmission Unit for the interface

name (*name*; default: **sangomaN**) - descriptive interface name

LMC/SBEI Synchronous Interfaces

Document revision 0.4 (February 6, 2008, 2:56 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS supports the following Lanmedia Corp (LMC)/SBE Inc interfaces:

- LMC/SBEI wanPCI-1T3 PCI T3 (also known as DS3, 44.736Mbps)
- LMC/SBEI wanPCI-1T1E1 PCI T1/E1 (also known as DS1 or LMC1200P, 1.544 Mbps or 2.048 Mbps)

Specifications

Packages required: ***synchronous***
License required: ***level4***
Home menu level: ***/interface sbe***
Standards and Technologies: ***T1/E1/T3/G.703, Frame Relay, PPP, Cisco-HDLC***
Hardware usage: ***Not significant***

Synchronous Interface Configuration

Home menu level: ***/interface sbe***

Description

With the introduction of 2.8 release, MikroTik RouterOS supports popular SBEI wanPCI-1T3 and wanPCI-1T1E1 cards. These cards provide a router with the ability to communicate over T1, E1 and T3 links directly, without the need of external CSU/DSU equipment.

Property Description

chdlc-keepalive (*time*; default: **10s**) - specifies the keepalive interval for Cisco HDLC protocol
circuit-type (*e1* | *e1-cas* | *e1-plain* | *e1-unframed* | *t1* | *t1-unframed*; default: **e1**) - the circuit type

particular interface is connected to

clock-rate (*integer*; default: **64000**) - internal clock rate in bps

clock-source (*internal* | *external*; default: **external**) - specifies whether the card should rely on supplied clock or generate its own

crc32 (yes | no; default: **no**) - Specifies whether to use CRC32 error correction algorithm or not

frame-relay-dce (*yes* | *no*; default: **no**) - specifies whether the device operates in Data Communication Equipment mode. The value yes is suitable only for T1 models

frame-relay-lmi-type (*ansi* | *ccitt*; default: **ansi**) - Frame Relay Line Management Interface Protocol type

line-protocol (*cisco-hdlc* | *frame-relay* | *sync-ppp*; default: **sync-ppp**) - encapsulated line protocol

long-cable (yes | no; default: **no**) - specifies whether to use signal phase shift for very long links

mtu (*integer*: 68..1500; default: **1500**) - IP protocol Maximum Transmission Unit

name (*name*; default: **sbeN**) - unique interface name.

scrambler (yes | no; default: **no**) - when enabled, makes the card unintelligible to anyone without a special receiver

Application Examples

Connecting two MT routers via T1 crossover

In the following example we will configure two routers to talk to each other via T1 link. The routers are named R1 and R2 with the addresses of 10.10.10.1/24 and 10.10.10.2/24, respectively. Cisco HDLC will be used as incapsulation protocol and circuit type will be regular T1.

First, we need to configure synchronous interfaces on both routers. Keep in mind, that one of the interfaces needs to be set to use its internal clock.

- On **R1** router:

```
[admin@MikroTik] > /interface sbe set sbe1 line-protocol=cisco-hdlc \
\... clock-source=internal circuit-type=t1 disabled=no
[admin@R1] > /interface sbe print
Flags: X - disabled, R - running
 0  R name="sbe1" mtu=1500 line-protocol=cisco-hdlc clock-rate=64000
      clock-source=internal crc32=no long-cable=no scrambler=no
      circuit-type=t1 frame-relay-lmi-type=ansi frame-relay-dce=no
      chdlc-keepalive=10s
[admin@R1] >
```

- On **R2** router:

```
[admin@MikroTik] > /interface sbe set sbe1 line-protocol=cisco-hdlc \
\... circuit-type=t1 disabled=no
[admin@R2] > /interface sbe print
Flags: X - disabled, R - running
 0  R name="sbe1" mtu=1500 line-protocol=cisco-hdlc clock-rate=64000
      clock-source=external crc32=no long-cable=no scrambler=no
      circuit-type=t1 frame-relay-lmi-type=ansi frame-relay-dce=no
      chdlc-keepalive=10s
[admin@R2] >
```

Then, we should assign IP addresses to both interfaces.

- On **R1** router:

```
[admin@R1] > /ip address add address=10.10.10.1/24 interface=sbe1
```

- On **R2** router:

```
[admin@R1] > /ip address add address=10.10.10.2/24 interface=sbe1
```

Finally, we could test connection by issuing **ping** command from **R1** router:

```
[admin@R1] > /ping 10.10.10.2
10.10.10.2 64 byte ping: ttl=64 time=7 ms
10.10.10.2 64 byte ping: ttl=64 time=8 ms
10.10.10.2 64 byte ping: ttl=64 time=8 ms
10.10.10.2 64 byte ping: ttl=64 time=8 ms
10.10.10.2 64 byte ping: ttl=64 time=8 ms
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 7/7.8/8 ms
[admin@R2] >
```

Wireless Client and Wireless Access Point Manual

Document revision 2.3 (January 22, 2008, 8:53 GMT)

This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

This manual discusses management of Atheros and Prism chipset based wireless NICs that comply with IEEE 802.11 set of standards. These interfaces use radio waves as a physical signal carrier and are capable of data transmission with speeds up to 108 Mbps (in 5GHz turbo-mode).

MikroTik RouterOS supports the Intersil Prism II PC/PCI, Atheros AR5000, AR5001X, AR5001X+, AR5002X+, AR5004X+ and AR5006 chipset based cards for working as wireless clients (**station** mode), wireless bridges (**bridge** mode), wireless access points (**ap-bridge** mode), and for antenna positioning (**alignment-only** mode). For furher information about supported wireless adapters, see Device Driver List

MikroTik RouterOS provides a complete support for IEEE 802.11a, 802.11b and 802.11g wireless networking standards. There are several additional features implemented for the wireless networking in RouterOS - WPA/WPA2 (Wi-Fi Protected Access), WEP (Wired Equivalent Privacy), software and hardware AES encryption, EAP authenticaton, WDS (Wireless Distribution System), DFS (Dynamic Frequency Selection), Alignment mode (for positioning antennas and monitoring wireless signal), VAP (Virtual Access Point), ability to disable packet forwarding among clients, Nstreme wireless transmission protocol and others. You can see the table of features supported by different cards.

The Nstreme protocol is MikroTik proprietary (i.e., incompatible with other vendors) wireless protocol aimed to improve point-to-point and point-to-multipoint wireless links. Advanced version of Nstreme, called Nstreme2 works with a pair of wireless cards (Atheros AR5210 and newer MAC chips only) - one for transmitting data and one for receiving.

Benefits of Nstreme protocol:

- Client polling. Polling reduces media access times, because the card does not need to ensure the air is "free" each time it needs to transmit data (the polling mechanism takes care of it)
- Very low protocol overhead per frame allowing super-high data rates
- No implied protocol limits on link distance
- No implied protocol speed degradation for long link distances
- Dynamic protocol adjustment depending on traffic type and resource usage

Quick Setup Guide

Let's consider that you have a wireless interface, called **wlan1**.

- To set it as an Access Point, working in 802.11g standard, using frequency **2442 MHz** and Service Set Identifier **test**, do the following configuration:

```
/interface wireless set wlan1 ssid=test frequency=2442 band=2.4ghz-b/g \
   mode=ap-bridge disabled=no
```

Now your router is ready to accept wireless clients.

- To make a point-to-point connection, using 802.11a standard, frequency **5805 MHz** and Service Set Identifier **p2p**, write:

```
/interface wireless set wlan1 ssid="p2p" frequency=5805 band=5ghz \
    mode=bridge disabled=no
```

The remote interface should be configured to station as showed below.

- To make the wireless interface as a wireless station, working in 802.11a standard and Service Set Identifier **p2p**:

```
/interface wireless set wlan1 ssid="p2p" band=5ghz mode=station disabled=no
```

Specifications

Packages required: ***wireless***
License required: ***level4 (station and bridge mode), level5 (station, bridge and AP mode), levelfreq (more frequencies)***
Home menu level: ***/interface wireless***
Standards and Technologies: ***IEEE802.11a***, ***IEEE802.11b***, ***IEEE802.11g***
Hardware usage: ***Not significant***

Description

The Atheros card has been tested for distances up to 20 km providing connection speed up to 17Mbit/s. With appropriate antennas and cabling the maximum distance should be as far as 50 km.

These values of **ack-timeout** were approximated from the tests done by us, as well as by some of our customers:

range	ack-timeout		
	5GHz	5GHz-turbo	2.4GHz-G
0km	default	default	default
5km	52	30	62
10km	85	48	96
15km	121	67	133
20km	160	89	174
25km	203	111	219
30km	249	137	368
35km	298	168	320
40km	350	190	375
45km	405	-	-

Please **note** that these are not the precise values. Depending on hardware used and many other factors they may vary up to +/- 15 microseconds.

You can also use **dynamic** ack-timeout value - the router will determine **ack-timeout** setting automatically by sending periodically packets with a different ack-timeout. Ack-timeout values by which ACK frame was received are saved and used later to determine the real ack-timeout.

The Nstreme protocol may be operated in three modes:

- **Point-to-Point mode** - controlled point-to-point mode with one radio on each side
- **Dual radio Point-to-Point mode (Nstreme2)** - the protocol will use two radios on both sides simultaneously (one for transmitting data and one for receiving), allowing superfast point-to-point connection
- **Point-to-Multipoint** - controlled point-to-multipoint mode with client polling (like AP-controlled TokenRing)

Hardware Notes

The MikroTik RouterOS supports as many Atheros chipset based cards as many free adapter slots are on your system. One license is valid for all cards on your system. **Note** that maximal number of PCMCIA sockets is 8.

Some chipsets are not stable with Atheros cards and cause radio to stop working. MikroTik RouterBoard 200, RouterBoard 500 series, and systems based on Intel i815 and i845 chipsets are tested and work stable with Atheros cards. There might be many other chipsets that are working stable, but it has been reported that some older chipsets, and some systems based on AMD Duron CPU are not stable.

Only AR5212 and newer Atheros MAC chips are stable with RouterBOARD200 connected via RouterBOARD14 four-port MiniPCI-to-PCI adapter. This note applies only to the RouterBOARD200 platform with Atheros-based cards.

Wireless Interface Configuration

Home menu level: ***/interface wireless***

Description

In this section we will discuss the most important part of the configuration.

Property Description

ack-timeout (*integer* | *dynamic* | *indoors*) - acknowledgement code timeout (transmission acceptance timeout) in microseconds for acknowledgement messages. Can be one of these:

- **dynamic** - ack-timeout is chosen automatically
- **indoors** - standard constant for indoor usage

adaptive-noise-immunity (yes | no; default: **yes**) - adjust various receiver parameters dynamically to minimize interference and noise effect on the signal quality. Only AR6001XL and AR6001GL and newer Atheros chips support this feature

allow-sharedkey (yes | no; default: **no**) - allow WEP Shared Key cilents to connect. Note that no authentication is done for these clients (WEP Shared keys are not compared to anything) - they are just accepted at once (if access list allows that)

antenna-gain (*integer*; default: **0**) - antenna gain in dBi. This parameter will be used to calculate whether your system meets regulatory domain's requirements in your country

antenna-mode (*ant-a* | *ant-b* | *rxa-txb* | *txa-rxb*; default: **ant-a**) - which antenna to use for transmit/receive data:

- **ant-a** - use only antenna a
- **ant-b** - use only antenna b
- **rxa-txb** - use antenna a for receiving packets, use antenna b for transmitting packets
- **txa-rxb** - use antenna a for transmitting packets, antenna b for receiving packets

area (*text*; default: **""**) - string value that is used to describe an Access Point. Connect List on the Client's side comparing this string value with area-prefix string value makes decision whether allow a Client connect to the AP. If area-prefix match the entire area string or only the beginning of it the Client is allowed to connect to the AP

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; default: **enabled**) - Address Resolution Protocol setting

band - operating band

- **2.4ghz-b** - IEEE 802.11b
- **2.4ghz-b/g** - IEEE 802.11g (supports also legacy IEEE 802.11b protocol)
- **2.4ghz-g-turbo** - IEEE 802.11g using double channel, providing air rate of up to 108 Mbit
- **2.4ghz-onlyg** - only IEEE 802.11g
- **5ghz** - IEEE 802.11a up to 54 Mbit
- **5ghz-turbo** - IEEE 802.11a using double channel, providing air rate of up to 108Mbit
- **2ghz-10mhz** - variation of IEEE 802.11g with half the band, and, accordingly, twice lower speed (air rate of up to 27Mbit)
- **2ghz-5mhz** - variation of IEEE 802.11g with quarter the band, and, accordingly, four times lower speed (air rate of up to 13.5Mbit)
- **5ghz-10mhz** - variation of IEEE 802.11a with half the band, and, accordingly, twice lower speed (air rate of up to 27Mbit)
- **5ghz-5mhz** - variation of IEEE 802.11a with quarter the band, and, accordingly, four times lower speed (air rate of up to 13.5Mbit)

basic-rates-a/g (*multiple choice: 6Mbps, 9Mbps, 12Mbps, 18Mbps, 24Mbps, 36Mbps, 48Mbps, 54Mbps*; default: **6Mbps**) - basic rates in 802.11a or 802.11g standard. This should be the minimal speed all the wireless network nodes support (they will not be ableto connect otherwise). It is recommended to leave this as default

basic-rates-b (*multiple choice: 1Mbps, 2Mbps, 5.5Mbps, 11Mbps*; default: **1Mbps**) - basic rates in 802.11b mode. This should be the minimal speed all the wireless network nodes support (they will not be ableto connect otherwise). It is recommended to leave this as default

burst-time (*time*; default: **disabled**) - time in microseconds which will be used to send data without stopping. Note that no other wireless cards in that network will be able to transmit data during burst-time microseconds. This setting is available only for AR5000, AR5001X, and AR5001X+ chipset based cards

compression (yes | no; default: **no**) - if enabled on AP (in ap-bridge or bridge mode), it advertizes that it is capable to use hardware data compression. If a client, connected to this AP, also supports and is configured to use the hardware data compression, it requests the AP to use compression. This property does not affect clients, which do not support compression.

country (*albania* | *algeria* | *argentina* | *armenia* | *australia* | *austria* | *azerbaijan* | *bahrain* | *belarus* | *belgium* | *belize* | *bolvia* | *brazil* | *brunei darussalam* | *bulgaria* | *canada* | *chile* | *china* | *colombia* | *costa rica* | *croatia* | *cyprus* | *czech republic* | *denmark* | *dominican republic* | *ecuador* | *egypt* | *el salvador* | *estonia* | *finland* | *france* | *france_res* | *georgia* | *germany* | *greece* | *guatemala* | *honduras* | *hong kong* | *hungary* | *iceland* | *india* | *indonesia* | *iran* | *ireland* | *israel* | *italy* | *japan* | *japan1* | *japan2* | *japan3* | *japan4* | *japan5* | *jordan* | *kazakhstan* | *korea republic* | *korea republic2* | *kuwait* | *latvia* | *lebanon* | *liechtenstein* | *lithuania* | *luxemburg* | *macau* | *macedonia* | *malaysia* | *mexico* | *monaco* | *morocco* | *netherlands* | *new zealand* | *no_country_set* | *north korea* | *norway* | *oman* | *pakistan* | *panama* | *peru* | *philippines* | *poland* | *portugal* | *puerto rico* | *qatar* | *romania* | *russia* | *saudi arabia* | *singapore* | *slovak republic* | *slovenia* | *south africa* | *spain* | *sweden* | *switzerland* | *syria* | *taiwan* | *thailand* | *trinidad & tobago* | *tunisia* | *turkey* | *ukraine* | *united arab emirates* | *united kingdom* | *united states* | *uruguay* | *uzbekistan* | *venezuela* | *viet nam* | *yemen* | *zimbabwe*; default: **no_country_set**) - limits wireless settings (frequency and transmit power) to those which are allowed in the respective country

- **no_country_set** - no regulatory domain limitations

default-ap-tx-limit (*integer*; default: **0**) - limits data rate for each wireless client (in bps)

- **0** - no limits

default-authentication (yes | no; default: **yes**) - specifies the default action on the client's side for APs that are not in connect list or on the AP's side for clients that are not in access list

- **yes** - enables AP to register a client if it is not in access list. In turn for client it allows to associate with AP not listed in client's connect list

default-client-tx-limit (*integer*; default: **0**) - limits each client's transmit data rate (in bps). Works only if the client is also a MikroTik Router

- **0** - no limits

default-forwarding (yes | no; default: **yes**) - whether to use data forwarding by default or not. If set to 'no', the registered clients will not be able to communicate with each other

dfs-mode (*none* | *radar-detect* | *no-radar-detect*; default: **none**) - used for APs to dynamically select frequency at which this AP will operate

- **none** - do not use DFS
- **no-radar-detect** - AP scans channel list from "scan-list" and chooses the frequency which is with the lowest amount of other networks detected
- **radar-detect** - AP scans channel list from "scan-list" and chooses the frequency which is with the lowest amount of other networks detected, if no radar is detected in this channel for 60 seconds, the AP starts to operate at this channel, if radar is detected, the AP continues searching for the next available channel which is with the lowest amount of other networks detected

disable-running-check (*yes* | *no*; default: **no**) - disable running check. If value is set to 'no', the router determines whether the card is up and running - for AP one or more clients have to be registered to it, for station, it should be connected to an AP. This setting affects the records in the routing table in a way that there will be no route for the card that is not running (the same applies to dynamic routing protocols). If set to 'yes', the interface will always be shown as running

disconnect-timeout (*time*; default: **3s**) - time since the third sending failure (3*(hw-retries+1) packets have been lost) at the lowest datarate only (i.e. since the first time on-fail-retry-time has been activated), when the client gets disconnected (logged as "extensive data loss")

frame-lifetime (*integer*; default: **0**) - frame lifetime in centiseconds since the first sending attempt to send the frame. Wireless normally does not drop any packets at all until the client is disconnected. If there is no need to accumulate packets, you can set the time after which the packet will be discarded

- **0** - never drop packets until the client is disconnected (default value)

frequency (*integer*) - operating frequency of the AP (ignored for the client, which always scans through its scan list regardless of the value set in this field)

frequency-mode (*regulatory-domain* | *manual-tx-power* | *superchannel*; default: **regulatory-domain**) - defines which frequency channels to allow

- **regulatory-domain** - use the channels allowed in the selected country at the allowed transmit power (with the configured antenna-gain deducted) only. Also note that in this mode card will never be configured to higher power than allowed by the respective regulatory domain
- **manual-tx-power** - use the channels allowed in the selected country only, but take transmit power from the tx-power settings
- **superchannel** - only possible with the Superchannel license. In this mode all hardware supported channels and transmit power settings are allowed

hide-ssid (*yes* | *no*; default: **no**) - whether to hide ssid or not in the beacon frames:

- **yes** - ssid is not included in the beacon frames. AP replies only to probe-requests with the given ssid
- **no** - ssid is included in beacon frames. AP replies to probe-requests with the given ssid ant to 'broadcast ssid' (empty ssid)

hw-retries (*integer*; default: **15**) - number of frame sending retries until the transmission is considered failed. Data rate is decreased upon failure, but if there is no lower rate, 3 sequential failures activate on-fail-retry-time transmission pause and the counter restarts. The frame is being retransmitted either until success or until client is disconnected

interface-type (*read-only: text*) - adapter type and model

mac-address (*MAC address*) - Media Access Control (MAC) address of the interface

master-interface (*name*) - physical wireless interface name that will be used by Virtual Access Point (VAP) interface

max-station-count (*integer*: 1..2007; default: **2007**) - maximal number of clients allowed to connect to AP. Real life experiments (from our customers) show that 100 clients can work with one AP, using traffic shaping

mode (*alignment-only* | *ap-bridge* | *bridge* | *nstreme-dual-slave* | *station* | *station-pseudobridge* | *station-pseudobridge-clone* | *station-wds* | *wds-slave*; default: **station**) - operating mode:

- **alignment-only** - this mode is used for positioning antennas (to get the best direction)
- **ap-bridge** - the interface is operating as an Access Point
- **bridge** - the interface is operating as a bridge. This mode acts like ap-bridge with the only difference being it allows only one client
- **nstreme-dual-slave** - the interface is used for nstreme-dual mode
- **station** - the interface is operating as a wireless station (client)
- **station-pseudobridge** - wireless station that can be put in bridge. MAC NAT is performed on all traffic sent over the wireless interface, so that it look like coming from the station's MAC address regardless of the actual sender (the standard does not allow station to send packets with different MAC address from its own). Reverse translation (when replies arrive from the AP to the pseudobridge station) is based on the ARP table. Non-IP protocols are being sent to the

default MAC address (the last MAC address, which the station has received a non-IP packet from). That means that if there is more than one client that uses non-IP protocols (for example, PPPoE) behind the station, none of them will be able to work correctly

- **station-pseudobridge-clone** - similar to the station-pseudobridge, but the station will clone MAC address of a particular device (set in the station-bridge-clone-mac property), i.e. it will change itsown address to the one of a different device. In case no address is set in the station-bridge-clone-mac property, the station postpones connecting to an AP until some packet, with the source MAC address different from any of the router itself, needs to be transmitted over that interface. It then connects to an AP with the MAC address of the device that have sent that packet
- **station-wds** - the interface is working as a station, but can communicate with a WDS peer
- **wds-slave** - the interface is working as it would work in ap-bridge mode, but it adapts to its WDS peer's frequency if it is changed

mtu (*integer*: 68..1600; default: **1500**) - Maximum Transmission Unit

name (*name*; default: **wlanN**) - assigned interface name

noise-floor-threshold (*integer | default*: -128..127; default: **default**) - noise strength in dBm below which the card will transmit

on-fail-retry-time (*time*; default: **100ms**) - time, after which we repeat to communicate with a wireless device, if a data transmission has failed 3 times on the lowest rate

periodic-calibration (*default | disabled | enabled*; default: **default**) - to ensure performance of chipset over temperature and environmental changes, the software performs periodic calibration

periodic-calibration-interval (*integer*; default: **60**) - interfal between periodic recalibrations, in seconds

preamble-mode (*both | long | short*; default: **both**) - sets the synchronization field in a wireless packet

- **long** - has a long synchronization field in a wireless packet (128 bits). Is compatible with 802.11 standard
- **short** - has a short synchronization field in a wireless packet (56 bits). Is not compatible with 802.11 standard. With short preamble mode it is possible to get slightly higher data rates
- **both** - supports both - short and long preamble

prism-cardtype (*30mW | 100mW | 200mW*) - specify the output of the Prism chipset based card

proprietary-extensions (*pre-2.9.25 | post-2.9.25*; default: **post-2.9.25**) - the method to insert additional information (MikroTik proprietary extensions) into the wireless frames. This option is needed to workaround incompatibility between the old (pre-2.9.25) method and new Intel Centrino PCI-Express cards

- **pre-2.9.25** - include extensions in the form accepted by older RouterOS versions. This will include the new format as well, so this mode is compatiblewith all RouterOS versions. This mode is incompatible with wireless clients built on the new Centrino wireless chipset and may as well be incompatible with some other stations

radio-name (*text*) - descriptive name of the card. Only for MikroTik devices

rate-set (*default | configured*) - which rate set to use:

- **default** - basic and supported-rates settings are not used, instead default values are used
- **configured** - basic and supported-rates settings are used as configured

scan-list (*multiple choice: integer | default*; default: **default**) - the list of channels to scan

- **default** - represents all frequencies, allowed by the regulatory domain (in the respective country). If no country is set, these frequencies are used - for 2.4GHz mode: 2412, 2417, 2422, 2427, 2432, 2437, 2442, 2447, 2452, 2457, 2462; for 2.4GHz-g-turbo mode: 2437; for 5GHz mode: 5180, 5200, 5220, 5240, 5260, 5280, 5300, 5320, 5745, 5765, 5785, 5805, 5825; for 5GHz-turbo: 5210, 5250, 5290, 5760, 5800

security-profile (*text*; default: **default**) - which security profile to use. Define security profiles under /interface wireless security-profiles where you can setup WPA or WEP wireless security, for further details, see the Security Profiles section of this manual

ssid (*text*; default: **MikroTik**) - Service Set Identifier. Used to separate wireless networks

supported-rates-a/g (*multiple choice: 6Mbps, 9Mbps, 12Mbps, 18Mbps, 24Mbps, 36Mbps, 48Mbps, 54Mbps*) - rates to be supported in 802.11a or 802.11g standard

supported-rates-b (*multiple choice: 1Mbps, 2Mbps, 5.5Mbps, 11Mbps*) - rates to be supported in 802.11b standard

tx-power (*integer*: -30..30; default: **17**) - manually sets the transmit power of the card (in dBm), if tx-power-mode is set to card rates or all-rates-fixed (see tx-power-mode description below)

tx-power-mode (*all-rates-fixed | card-rates | default | manual-table*; default: **default**) - choose the transmit power mode for the card:

- **all-rates-fixed** - use one transmit power value for all rates, as configured in tx-power
- **card-rates** - use transmit power, that for different rates is calculated according the cards transmit power algorithm, which as an argument takes tx-power value
- **default** - use the default tx-power
- **manual-table** - use the transmit powers as defined in /interface wireless manual-tx-power-table

update-stats-interval (*time*) - how often to update (request from the clients) signal strength and ccq values in /interface wireless registration-table

wds-cost-range (*integer*; default: **50-150**) - range, within which the bridge port cost of the WDS links are adjusted. The calculations are based on the p-throughput value of the respective WDS interface, which represents estimated approimate rhtoughput on the interface, which is mapped on the wds-cost-range scale so that bigger p-throughput would correspond to numerically lower port cost. The cost is recalculated every 20 seconds or when the p-throughput changes more than by 10% since the last recalculation

wds-default-bridge (*name*; default: **none**) - the default bridge for WDS interface. If you use dynamic WDS then it is very useful in cases when wds connection is reset - the newly created dynamic WDS interface will be put in this bridge

wds-default-cost (*integer*; default: **100**) - default bridge port cost of the WDS links

wds-ignore-ssid (yes | no; default: **no**) - if set to 'yes', the AP will create WDS links with any other AP in this frequency. If set to 'no' the ssid values must match on both APs

wds-mode (*disabled | dynamic | static*) - WDS mode:

- **disabled** - WDS interfaces are disabled
- **dynamic** - WDS interfaces are created 'on the fly'
- **static** - WDS interfaces are created manually

wmm-support (*disabled | enabled | required*) - whether to allow (or require) peer to use WMM extensions to provide basic quality of service

Notes

The IEEE 802.11 standard limitation makes it impossible for wireless cards in station mode to work as expected when bridged. That means that if you need to create a bridge, you should not use station mode on that machine. In case you need a bridge on a wireless station, use **station-wds** mode (may only be used in the AP supports WDS). Bridging on the AP side works fine.

It is strongly suggested to leave basic rates at the lowest setting possible.

Using **compression**, the AP can serve approximately 50 clients with compression enabled!

Compression is supported only by Atheros wireless cards.

If **disable-running-check** value is set to **no**, the router determines whether the network interface is up and running - in order to show flag **R** for AP, one or more clients have to be registered to it, for station, it should be connected to an AP. If the interface does not appear as running (**R**), its route in the routing table is shown as **invalid**! If set to **yes**, the interface will always be shown as running.

On Atheros-based cards, encryption (WEP, WPA, etc.) does not work when compression is enabled.

The **tx-power** default setting is the maximum tx-power that the card can use. If you want to use larger tx-rates, you are able to set them, but **do it at your own risk**! Usually, you can use this parameter to reduce the **tx-power**.

In general tx-power controlling properties should be left at the default settings. Changing the default setting may help with some cards in some situations, but without testing, the most common result is degradation of range and throughput. Some of the problems that may occur are: (1) overheating of the power amplifier chip and the card which will cause lower efficiency and more data errors; (2) overdriving the amplifier which will cause more data errors; (3) excessive power usage for the card and this may overload the 3.3V power supply of the board that the card is located on resulting in voltage drop and reboot or excessive temperatures for the board.

For different versions of Atheros chipset there are different value range of **ack-timeout** property:

Chipset version	5ghz		5ghz-turbo		2ghz-b		2ghz-g	
	default	max	default	max	default	max	default	max
5000 (5.2GHz only)	30	204	22	102	N/A	N/A	N/A	N/A
5211 (802.11a/b)	30	409	22	204	109	409	N/A	N/A
5212 (802.11a/b/g)	25	409	22	204	30	409	52	409

If the wireless interfaces are put in **nstreme-dual-slave** mode, all configuration will take place in **/interface wireless nstreme-dual** submenu, described further on in this manual. In that case, configuration made in this submenu will be partially ignored. WDS cannot be used together with the Nstreme-dual.

Some options are not shown by default - use **print advanced** in the console or press the "Advanced Mode" button in Winbox to see all the parameters

Example

This example shows how configure a wireless client.

To see current interface settings:

```
[admin@MikroTik] interface wireless> print
Flags: X - disabled, R - running
 0    name="wlan1" mtu=1500 mac-address=00:0C:42:18:5C:3D arp=enabled
      interface-type=Atheros AR5413 mode=station ssid="MikroTik" frequency=2412
      band=2.4ghz-b scan-list=default antenna-mode=ant-a wds-mode=disabled
      wds-default-bridge=none wds-ignore-ssid=no default-authentication=yes
      default-forwarding=yes default-ap-tx-limit=0 default-client-tx-limit=0
      hide-ssid=no security-profile=default compression=no
[admin@MikroTik] interface wireless>
```

Set the **ssid** to *mmt*, **band** to *2.4-b/g* and enable the interface. Use the monitor command to see the connection status.

```
[admin@MikroTik] interface wireless> set 0 ssid=mmt disabled=no \
band=2.4ghz-b/g
[admin@MikroTik] interface wireless> monitor wlan1
                 status: connected-to-ess
                   band: 2.4ghz-g
              frequency: 2412MHz
                tx-rate: "54Mbps"
                rx-rate: "54Mbps"
                   ssid: "mmt"
                  bssid: 00:0C:42:05:00:14
             radio-name: "000C42050014"
        signal-strength: -23dBm
     tx-signal-strength: -35dBm
            noise-floor: -96dBm
        signal-to-noise: 73dB
                 tx-ccq: 79%
                 rx-ccq: 46%
           p-throughput: 28681
         overall-tx-ccq: 79%
  authenticated-clients: 1
    current-ack-timeout: 56
               wds-link: no
                nstreme: no
           framing-mode: none
       routeros-version: "3.0"
                last-ip: 10.10.10.1
    802.1x-port-enabled: yes
            compression: no
      current-tx-powers: 1Mbps:19(19),2Mbps:19(19),5.5Mbps:19(19),
                         11Mbps:19(19),6Mbps:19(19),9Mbps:19(19),
                         12Mbps:19(19),18Mbps:19(19),24Mbps:19(19),
                         36Mbps:18(18),48Mbps:17(17),54Mbps:16(16)
    notify-external-fdb: no

[admin@MikroTik] interface wireless>
```

The 'ess' stands for Extended Service Set (IEEE 802.11 wireless networking).

Interface Monitor

Command name: ***/interface wireless monitor [interface name]***

Description

With this command you can monitor your link information. It is very useful if you have established a wireless link and want to monitor its status.

Property Description

802.1x-port-enabled (*read-only:* yes | no) - (on station only) whether the data exchange is allowed with the AP (i.e., whether 802.1x authentication is completed, if needed). Compare authenticated-clients and registered-clients

authenticated-clients (*read-only: integer*) - clients that have successfully completed 802.11 authentication process and have associated with the AP. Normally it is possible to exchange data with client right after this step, however WPA/WPA2 needs additional 802.1x authentication and dynamic key exchange procedures that start only after this stage (see registered-clients). For a wireless station this property relates to its AP

band - operating band

bssid (*read-only: MAC address*) - (on station only) MAC address of the AP

compression (*read-only:* yes | no) - (on station only) whether data compression is enabled for this interface

current-ack-timeout (*read-only: integer*) - current value of ack-timeout

current-tx-powers (*read-only: text*) - current transmit power values for every rate supported by the link

framing-mode (*read-only: text*) - (on station only) current framing mode

frequency (*read-only: integer*) - operating frequency

last-ip (*read-only: IP address*) - (on station only) source IP address found in the last IP packet received from the AP

noise-floor (*read-only: text*) - (on station only) received current noise level

notify-external-fdb (*read-only:* yes | no) - whether forwarding database is to be generated from the link's registration table to add known hosts in the local bridge host table (i.e., the associated bridge port is configured to request this information - its respective external-fdb property is set to auto or yes)

nstreme (*read-only:* yes | no) - whether nstreme protocol is used for this link

overall-tx-ccq (*read-only: integer*) - overall link CCQ, for transmitting to the wireless infrastructure, not to aome particular peers

p-throughput (*read-only: integer*) - (on station only) estimated approximate throughput that is expected on the given link, by taking into account the effective transmit rate and hardware retries. Calculated once in 5 seconds

radio-name (*read-only: text*) - (on station only) radio name

registered-clients (*read-only: integer*) - (on AP only) number of fully authenticated clients, that have completed not only 802.11 authentication procedures (as specified in the authenticated-clients propery), but also 802.1x ones. Registered clients are listed in the registration table and are ready for data exchange

routeros-version (*read-only: text*) - (on station only) RouterOS version installed on the AP

rx-ccq (*read-only: integer*: 0..100) - (on station only) Client Connection Quality - a value in percent that shows how effective the receive bandwidth (this value is received from the other end as this value represents its transmission quality) is used regarding the theoretically maximum available bandwidth. Mostly it depends from an amount of retransmited wireless frames.

rx-rate (*read-only: text*) - (on station only) current receive air rate

signal-strength (*read-only: text*) - (on station only) received signal strength

signal-to-noise (*read-only: text*) - (on station only) signal to noise ratio

ssid (*read-only: text*) - (on station only) SSID

status (*read-only: searching-for-frequency* | *radar-detecting* | *running-ap* | *connected-to-ess* | *disabled*) - interface status

tx-ccq (*read-only: integer*: 0..100) - (on station only) Client Connection Quality - a value in percent that shows how effective the transmit bandwidth is used regarding the theoretically maximum available bandwidth. Mostly it depends from an amount of retransmited wireless frames.

tx-rate (*read-only: text*) - (on station only) current transmit air rate

tx-signal-strength (*read-only: text*) - (on station only) received signal strength on the AP side (available if the AP is MikroTik RouterOS)

wds-link (*read-only:* yes | no) - (on station only) whether this link supports WDS (i.e., is in station-wds mode)

Notes

Most values reported in station mode but absent for AP, are available in the registration table for particular stations connected to the AP.

Nstreme Settings

Home menu level: ***/interface wireless nstreme***

Description

You can switch a wireless card to the nstreme mode. In that case the card will work only with nstreme clients.

Property Description

disable-csma (yes | no; default: **no**) - disable CSMA/CA when polling is used (better performance)

enable-nstreme (yes | no; default: **no**) - whether to switch the card into the nstreme mode

enable-polling (yes | no; default: **yes**) - whether to use polling for clients

framer-limit (*integer*; default: **3200**) - maximal frame size

framer-policy (*none* | *best-fit* | *exact-size* | *dynamic-size*; default: **none**) - the method how to combine frames. A number of frames may be combined into a bigger one to reduce the amount of protocol overhead (and thus increase speed). The card is not waiting for frames, but in case a number of packets are queued for transmitting, they can be combined. There are several methods of framing:

- **none** - do nothing special, do not combine packets (framing is disabled)
- **best-fit** - put as much packets as possible in one frame, until the framer-limit limit is met, but do not fragment packets
- **exact-size** - put as much packets as possible in one frame, until the framer-limit limit is met, even if fragmentation will be needed (best performance)
- **dynamic-size** - choose the best frame size dynamically

name (*name*) - reference name of the interface

Notes

The settings here (except for enabling nstreme) are relevant only on Access Point, they are ignored for client devices! The client automatically adapts to the AP settings.

WDS for Nstreme protocol requires using station-wds mode on one of the peers. Configurations with WDS between AP modes (**bridge** and **ap-bridge**) will not work.

Example

To enable the nstreme protocol on the **wlan1** radio with exact-size framing:

```
[admin@MikroTik] interface wireless nstreme> print
 0 name="wlan1" enable-nstreme=no enable-polling=yes disable-csma=no
   framer-policy=none framer-limit=3200
[admin@MikroTik] interface wireless nstreme> set wlan1 enable-nstreme=yes \
\... framer-policy=exact-size
```

Nstreme2 Group Settings

Home menu level: ***/interface wireless nstreme-dual***

Description

Two radios in **nstreme-dual-slave** mode can be grouped together to make nstreme2 Point-to-Point connection. To put wireless interfaces into a nstreme2 group, you should set their **mode** to **nstreme-dual-slave**. Many parameters from **/interface wireless** menu are ignored, using the nstreme2, except:

- frequency-mode
- country
- antenna-gain
- tx-power
- tx-power-mode
- antenna-mode

Property Description

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; default: **enabled**) - Address Resolution Protocol setting

disable-csma (yes | no; default: **no**) - disable CSMA/CA (better performance)

disable-running-check (yes | no) - whether the interface should always be treated as running even if there is no connection to a remote peer

framer-limit (*integer*; default: **2560**) - maximal frame size

framer-policy (*none* | *best-fit* | *exact-size*; default: **none**) - the method how to combine frames. A number of frames may be combined into one bigger one to reduce the amout of protocol overhead (and thus increase speed). The card are not waiting for frames, but in case a number packets are queued for transmitting, they can be combined. There are several methods of framing:

- **none** - do nothing special, do not combine packets
- **best-fit** - put as much packets as possible in one frame, until the framer-limit limit is met, but do not fragment packets
- **exact-size** - put as much packets as possible in one frame, until the framer-limit limit is met, even if fragmentation will be needed (best performance)

mac-address (*read-only: MAC address*) - MAC address of the transmitting wireless card in the set

mtu (*integer*: 0..1600; default: **1500**) - Maximum Transmission Unit

name (*name*) - reference name of the interface

rates-a/g (*multiple choice: 6Mbps, 9Mbps, 12Mbps, 18Mbps, 24Mbps, 36Mbps, 48Mbps, 54Mbps*) - rates to be supported in 802.11a or 802.11g standard

rates-b (*multiple choice: 1Mbps, 2Mbps, 5.5Mbps, 11Mbps*) - rates to be supported in 802.11b standard

remote-mac (*MAC address*; default: **00:00:00:00:00:00**) - which MAC address to connect to (this would be the remote receiver card's MAC address)

rx-band - operating band of the receiving radio

- **2.4ghz-b** - IEEE 802.11b
- **2.4ghz-g** - IEEE 802.11g
- **2.4ghz-g-turbo** - IEEE 802.11g in Atheros proprietary turbo mode (up to 108Mbit)
- **5ghz** - IEEE 802.11a up to 54 Mbit
- **5ghz-turbo** - IEEE 802.11a in Atheros proprietary turbo mode (up to 108Mbit)
- **2ghz-10mhz** - variation of IEEE 802.11g with half the band, and, accordingly, twice lower speed (air rate of up to 27Mbit)
- **2ghz-5mhz** - variation of IEEE 802.11g with quarter the band, and, accordingly, four times lower speed (air rate of up to 13.5Mbit)
- **5ghz-10mhz** - variation of IEEE 802.11a with half the band, and, accordingly, twice lower speed (air rate of up to 27Mbit)
- **5ghz-5mhz** - variation of IEEE 802.11a with quarter the band, and, accordingly, four times lower speed (air rate of up to 13.5Mbit)

rx-frequency (*integer*; default: **5320**) - Frequency to use for receiving frames

rx-radio (*name*) - which radio should be used for receiving frames

tx-band - operating band of the transmitting radio

- **2.4ghz-b** - IEEE 802.11b
- **2.4ghz-g** - IEEE 802.11g
- **2.4ghz-g-turbo** - IEEE 802.11g in Atheros proprietary turbo mode (up to 108Mbit)
- **5ghz** - IEEE 802.11a up to 54 Mbit
- **5ghz-turbo** - IEEE 802.11a in Atheros proprietary turbo mode (up to 108Mbit)
- **2ghz-10mhz** - variation of IEEE 802.11g with half the band, and, accordingly, twice lower speed (air rate of up to 27Mbit)

- **2ghz-5mhz** - variation of IEEE 802.11g with quarter the band, and, accordingly, four times lower speed (air rate of up to 13.5Mbit)
- **5ghz-10mhz** - variation of IEEE 802.11a with half the band, and, accordingly, twice lower speed (air rate of up to 27Mbit)
- **5ghz-5mhz** - variation of IEEE 802.11a with quarter the band, and, accordingly, four times lower speed (air rate of up to 13.5Mbit)

tx-frequency (*integer*; default: **5180**) - Frequency to use for transmitting frames

tx-radio (*name*) - which radio should be used for transmitting frames

Notes

WDS cannot be used on Nstreme-dual links.

The difference between **tx-freq** and **rx-freq** should be about 200MHz (more is recommended) because of the interference that may occur!

You can use different bands for rx and tx links. For example, transmit in **2.4ghz-g-turbo** and receive data, using **2.4ghz-b** band.

Example

To enable the nstreme2 protocol on a router:

1. Having two Atheros based cards which are not used for anything else, to group them into a nstreme interface, switch both of them into **nstreme-dual-slave** mode:

```
[admin@MikroTik] interface wireless> print
Flags: X - disabled, R - running
 0  R name="wlan1" mtu=1500 mac-address=00:0C:42:05:00:14 arp=enabled
      interface-type=Atheros AR5413 mode=station ssid="MikroTik"
      frequency=2412 band=2.4ghz-b/g scan-list=default antenna-mode=ant-a
      wds-mode=disabled wds-default-bridge=none wds-ignore-ssid=no
      default-authentication=yes default-forwarding=yes
      default-ap-tx-limit=0 default-client-tx-limit=0 hide-ssid=no
      security-profile=default compression=no

 1    name="wlan2" mtu=1500 mac-address=00:80:48:41:AF:2A arp=enabled
      interface-type=Atheros AR5413 mode=station ssid="MikroTik" frequency=2412
      band=2.4ghz-b/g scan-list=default antenna-mode=ant-a wds-mode=disabled
      wds-default-bridge=none wds-ignore-ssid=no default-authentication=yes
      default-forwarding=yes default-ap-tx-limit=0 default-client-tx-limit=0
      hide-ssid=no security-profile=default compression=no
[admin@MikroTik] interface wireless> set 0,1 mode=nstreme-dual-slave
```

2. Then add nstreme2 interface with exact-size framing:

```
[admin@MikroTik] interface wireless nstreme-dual> add \
\... framer-policy=exact-size
```

3. Configure which card will be receiving and which - transmitting and specify remote receiver card's MAC address:

```
[admin@MikroTik] interface wireless nstreme-dual> print
Flags: X - disabled, R - running
```

```
  0 X  name="n-streme1" mtu=1500 mac-address=00:00:00:00:00:00 arp=enabled
       disable-running-check=no tx-radio=(unknown) rx-radio=(unknown)
       remote-mac=00:00:00:00:00:00 tx-band=5GHz tx-frequency=5180
       rx-band=5GHz rx-frequency=5320 disable-csma=no
       rates-b=1Mbps,2Mbps,5.5Mbps,11Mbps
       rates-a/g=6Mbps,9Mbps,12Mbps,18Mbps,24Mbps,36Mbps,48Mbps,54Mbps
       framer-policy=exact-size framer-limit=4000
 [admin@MikroTik] interface wireless nstreme-dual> set 0 disabled=no \
 \... tx-radio=wlan1 rx-radio=wlan2 remote-mac=00:0C:42:05:0B:12
 [admin@MikroTik] interface wireless nstreme-dual> print
 Flags: X - disabled, R - running
  0 R  name="n-streme1" mtu=1500 mac-address=00:0C:42:05:0B:12 arp=enabled
       disable-running-check=no tx-radio=wlan1 rx-radio=wlan2
       remote-mac=00:00:00:00:00:00 tx-band=5GHz tx-frequency=5180
       rx-band=5GHz rx-frequency=5320 disable-csma=no
       rates-b=1Mbps,2Mbps,5.5Mbps,11Mbps
       rates-a/g=6Mbps,9Mbps,12Mbps,18Mbps,24Mbps,36Mbps,48Mbps,54Mbps
       framer-policy=exact-size framer-limit=4000
 [admin@MikroTik] interface wireless nstreme-dual>
```

Registration Table

Home menu level: ***/interface wireless registration-table***

Description

In the registration table you can see various information about currently connected clients. It is used only for Access Points.

Property Description

802.1x-port-enabled (*read-only:* yes | no) - whether the data exchange is allowed with the peer (i.e., whether 802.1x authentication is completed, if needed)

ack-timeout (*read-only: integer*) - current value of ack-timeout

ap (*read-only:* yes | no) - whether the connected device is an Access Point or not

ap-tx-limit (*read-only: integer*) - transmit rate limit on the AP, in bits per second

authentication-type (*read-only: none* | *wpa-psk* | *wpa2-psk* | *wpa-eap* | *wpa2-eap*) - authentication method used for the peer

bytes (*read-only: integer*, *integer*) - number of sent and received packet bytes

client-tx-limit (*read-only: integer*) - transmit rate limit on the AP, in bits per second

compression (*read-only:* yes | no) - whether data compresson is used for this peer

encryption (*read-only: aes-ccm* | *tkip*) - unicast encryption algorithm used

frame-bytes (*read-only: integer*, *integer*) - number of sent and received data bytes excluding header information

frames (*read-only: integer, integer*) - number of sent and received 802.11 data frames excluding retransmitted data frames

framing-current-size (*read-only: integer*) - current size of combined frames

framing-limit (*read-only: integer*) - maximal size of combined frames

framing-mode (*read-only: none* | *best-fit* | *exact-size*; default: **none**) - the method how to combine frames

group-encryption (*read-only: aes-ccm | tkip*) - group encryption algorithm used

hw-frame-bytes (*read-only: integer, integer*) - number of sent and received data bytes including header information

hw-frames (*read-only: integer, integer*) - number of sent and received 802.11 data frames including retransmitted data frames

interface (*read-only: name*) - interface that client is registered to

last-activity (*read-only: time*) - last interface data tx/rx activity

last-ip (*read-only: IP address*) - IP address found in the last IP packet received from the registered client

mac-address (*read-only: MAC address*) - MAC address of the registered client

nstreme (*read-only:* yes | no) - whether nstreme protocol is used for this link

p-throughput (*read-only: integer*) - estimated approximate throughput that is expected to the given peer, taking into account the effective transmit rate and hardware retries. Calculated once in 5 seconds

packed-bytes (*read-only: integer, integer*) - number of bytes packed into larger frames for transmitting/receiving (framing)

packed-frames (*read-only: integer, integer*) - number of frames packed into larger ones for transmitting/receiving (framing)

packets (*read-only: integer, integer*) - number of sent and received network layer packets

radio-name (*read-only: text*) - radio name of the peer

routeros-version (*read-only: name*) - RouterOS version of the registered client

rx-ccq (*read-only: integer*: 0..100) - Client Connection Quality - a value in percent that shows how effective the receive bandwidth is used regarding the theoretically maximum available bandwidth. Mostly it depends from an amount of retransmited wireless frames.

rx-rate (*read-only: integer*) - receive data rate

signal-strength (*read-only: integer*) - average strength of the client signal recevied by the AP

signal-to-noise (*read-only: text*) - signal to noise ratio

strength-at-rates (*read-only: text*) - signal strength level at different rates together with time how long were these rates used

tx-ccq (*read-only: integer*: 0..100) - Client Connection Quality - a value in percent that shows how effective the transmit bandwidth is used regarding the theoretically maximum available bandwidth. Mostly it depends from an amount of retransmited wireless frames.

tx-frames-timed-out (*read-only: integer*) - number of frames that have been discarded due to frame-lifetime timeout

tx-rate (*read-only: integer*) - transmit data rate

tx-signal-strength (*read-only: integer*) - average power of the AP transmit signal as received by the client device

uptime (*read-only: time*) - time the client is associated with the access point

wds (*read-only: no | yes*) - whether the connected client is using wds or not

wmm-enabled (*read-only:* yes | no) - whether WMM is used with this peer

Example

To see registration table showing all clients currently associated with the access point:

```
[admin@MikroTik] interface wireless registration-table> print
 # INTERFACE          RADIO-NAME       MAC-ADDRESS       AP  SIGNAL... TX-RATE
 0 wlan1              000C42185C3D     00:0C:42:18:5C:3D no  -38dBm... 54Mbps
[admin@MikroTik] interface wireless registration-table>
```

To get additional statistics:

```
[admin@MikroTik] interface wireless> registration-table print stats
 0 interface=wlan1 radio-name="000C42185C3D" mac-address=00:0C:42:18:5C:3D
   ap=no wds=no rx-rate="1Mbps" tx-rate="54Mbps" packets=696,4147
   bytes=5589,96698 frames=696,4147 frame-bytes=5589,71816
   hw-frames=770,4162 hw-frame-bytes=24661,171784 tx-frames-timed-out=0
   uptime=3h50m35s last-activity=2s440ms signal-strength=-38dBm@1Mbps
   signal-to-noise=54dB
   strength-at-rates=-38dBm@1Mbps 2s440ms,-37dBm@2Mbps 3h50m35s180ms,-
                     37dBm@5.5Mbps 3h50m23s330ms,-36dBm@11Mbps 3h45m8s330ms,-
                     37dBm@9Mbps 3h44m13s340ms,-36dBm@12Mbps 3h43m55s170ms,-
                     36dBm@18Mbps 3h43m43s340ms,-36dBm@24Mbps 3h43m25s180ms,-
                     37dBm@36Mbps 3h43m8s130ms,-42dBm@48Mbps 55s180ms,-
                     41dBm@54Mbps 3s610ms
   tx-signal-strength=-43dBm tx-ccq=66% rx-ccq=88% p-throughput=30119
   ack-timeout=56 nstreme=no framing-mode=none routeros-version="3.0"
   ap-tx-limit=0 client-tx-limit=0 802.1x-port-enabled=yes compression=no
   wmm-enabled=no
  [admin@MikroTik] interface wireless>
```

Connect List

Home menu level: ***/interface wireless connect-list***

Description

The Connect List is a list of rules (order is important), that determine to which AP the station should connect to.

At first, the station is searching for APs all frequencies (from **scan-list**) in the respective band and makes a list of Access Points. If the **ssid** is set under **/interface wireless**, the router removes all Access Points from its AP list which do not have such **ssid**

If a rule is matched and the parameter **connect** is set to **yes**, the station will connect to this AP. If the parameter says **connect=no** or the rule is not matched, we jump to the next rule.

If we have gone through all rules and haven't connected to any AP, yet. The router chooses an AP with the best signal and **ssid** that is set under **/interface wireless**.

In case when the station has not connected to any AP, this process repeats from beginning.

Property Description

area-prefix (*text*) - a string that indicates the beginning from the area string of the AP. If the AP's area begins with area-prefix, then this parameter returns true

connect (yes | no) - whether to connect to AP that matches this rule

interface (*name*) - name of the wireless interface

mac-address (*MAC address*) - MAC address of the AP. If set to 00:00:00:00:00:00, all APs are accepted

security-profile (*name*; default: **none**) - name of the security profile, used to connect to the AP. If none, then those security profile is used which is configured for the respective interface

signal-range (*integer*) - signal strength range in dBm. Rule is matched, if the signal from AP is within this range

ssid (*text*) - the ssid of the AP. If none set, all ssid's are accepted. Different ssids will be meaningful, if the ssid for the respective interface is set to ""

Access List

Home menu level: ***/interface wireless access-list***

Description

The access list is used by the Access Point to restrict associations of clients. This list contains MAC addresses of clients and determines what action to take when client attempts to connect. Also, the forwarding of frames sent by the client is controlled. Note that is is an ordered list (i.e., checked from top to bottom).

The association procedure is as follows: when a new client wants to associate to the AP that is configured on interface **wlanN**, an entry with client's MAC address and interface **wlanN** is looked up sequentially from top to bottom in the access-list. If such entry is found, action specified in the access list is performed, else **default-authentication** and **default-forwarding** arguments of interface **wlanN** are taken.

Property Description

ap-tx-limit (*integer*; default: **0**) - limits data rate for this wireless client (in bps)

- **0** - no limits

authentication (*yes* | *no*; default: **yes**) - whether to accept or to reject this client when it tries to connect

client-tx-limit (*integer*; default: **0**) - limits this client's transmit data rate (in bps). Works only if the client is also a MikroTik Router

- **0** - no limits

forwarding (*yes* | *no*; default: **yes**) - whether to forward the client's frames to other wireless clients

interface (*name*) - name of the respective interface

mac-address (*MAC address*) - MAC address of the client (can be 00:00:00:00:00:00 for any client)

private-algo (*104bit-wep* | *40bit-wep* | *none*) - which encryption algorithm to use

private-key (*text*; default: **""**) - private key of the client. Used for private-algo

private-pre-shared-key (*text*) - private preshared key for that station (in case any of the PSK authentication methods were used)

signal-range (*integer*) - signal strength range in dBm. Rule is matched, if the signal from AP is within this range

time (*time*) - rule is only matched during the specified period of time

Notes

If you have default authentication action for the interface set to yes, you can disallow this node to register at the AP's interface wlanN by setting authentication=no for it. Thus, all nodes except this one will be able to register to the interface wlanN.

If you have default authentication action for the interface set to no, you can allow this node to register at the AP's interface wlanN by setting authentication=yes for it. Thus, only the specified nodes will be able to register to the interface wlanN.

Example

To allow authentication and forwarding for the client 00:01:24:70:3A:BB from the wlan1 interface using WEP 40bit algorithm with the key **1234567890**:

```
[admin@MikroTik] interface wireless access-list> add mac-address= \
\... 00:01:24:70:3A:BB interface=wlan1 private-algo=40bit-wep private-key=1234567890
[admin@MikroTik] interface wireless access-list> print
Flags: X - disabled
 0   mac-address=00:01:24:70:3A:BB interface=wlan1 signal-range=-120.120
     authentication=yes forwarding=yes ap-tx-limit=0 client-tx-limit=0
     private-algo=40bit-wep private-key="1234567890" private-pre-shared-key=""
[admin@MikroTik] interface wireless access-list>
```

Info

Home menu level: ***/interface wireless info***

Description

This facility provides you with general wireless interface information.

Property Description

2ghz-b-channels (*multiple choice, read-only: 2312, 2317, 2322, 2327, 2332, 2337, 2342, 2347, 2352, 2357, 2362, 2367, 2372, 2412, 2417, 2422, 2427, 2432, 2437, 2442, 2447, 2452, 2457, 2462, 2467, 2472, 2484, 2512, 2532, 2552, 2572, 2592, 2612, 2632, 2652, 2672, 2692, 2712, 2732*) - the list of 2GHz IEEE 802.11b channels (frequencies are given in MHz)

2ghz-g-channels (*multiple choice, read-only: 2312, 2317, 2322, 2327, 2332, 2337, 2342, 2347, 2352, 2357, 2362, 2367, 2372, 2412, 2417, 2422, 2427, 2432, 2437, 2442, 2447, 2452, 2457, 2462, 2467, 2472, 2512, 2532, 2552, 2572, 2592, 2612, 2632, 2652, 2672, 2692, 2712, 2732, 2484*) - the list of 2GHz IEEE 802.11g channels (frequencies are given in MHz)

5ghz-channels (*multiple choice, read-only: 4920, 4925, 4930, 4935, 4940, 4945, 4950, 4955, 4960, 4965, 4970, 4975, 4980, 4985, 4990, 4995, 5000, 5005, 5010, 5015, 5020, 5025, 5030, 5035, 5040, 5045, 5050, 5055, 5060, 5065, 5070, 5075, 5080, 5085, 5090, 5095, 5100, 5105, 5110, 5115, 5120, 5125, 5130, 5135, 5140, 5145, 5150, 5155, 5160, 5165, 5170, 5175, 5180, 5185, 5190, 5195, 5200, 5205, 5210, 5215, 5220, 5225, 5230, 5235, 5240, 5245, 5250, 5255, 5260, 5265, 5270, 5275, 5280, 5285, 5290, 5295, 5300, 5305, 5310, 5315, 5320, 5325, 5330, 5335, 5340, 5345, 5350, 5355, 5360, 5365, 5370, 5375, 5380, 5385, 5390, 5395, 5400, 5405, 5410, 5415, 5420, 5425, 5430, 5435, 5440, 5445, 5450, 5455, 5460, 5465, 5470, 5475, 5480, 5485, 5490, 5495, 5500, 5505, 5510, 5515, 5520, 5525, 5530, 5535, 5540, 5545, 5550, 5555, 5560, 5565, 5570, 5575, 5580, 5585, 5590, 5595, 5600, 5605, 5610, 5615, 5620, 5625, 5630, 5635, 5640, 5645, 5650, 5655, 5660, 5665, 5670, 5675, 5680, 5685, 5690, 5695, 5700, 5705, 5710, 5715, 5720, 5725, 5730, 5735, 5740, 5745, 5750, 5755, 5760, 5765, 5770, 5775, 5780, 5785, 5790, 5795, 5800, 5805, 5810, 5815, 5820, 5825, 5830, 5835, 5840,* *5845*, *5850*, *5855*, *5860*, *5865*, *5870*, *5875*, *5880*, *5885*, *5890*, *5895*, *5900*, *5905*, *5910*, *5915*, *5920*, *5925*, *5930*, *5935*, *5940*, *5945*, *5950*, *5955*, *5960*, *5965*, *5970*, *5975*, *5980*, *5985*, *5990*, *5995*, *6000*, *6005*, *6010*, *6015*, *6020*, *6025*, *6030*, *6035*, *6040*, *6045*, *6050*, *6055*, *6060*, *6065*, *6070*, *6075*, *6080*, *6085*, *6090*, *6095*, *6100*) - the list of 5GHz channels (frequencies are given in MHz)

5ghz-turbo-channels (*multiple choice*, *read-only: 4920*, *4925*, *4930*, *4935*, *4940*, *4945*, *4950*, *4955*, *4960*, *4965*, *4970*, *4975*, *4980*, *4985*, *4990*, *4995*, *5000*, *5005*, *5010*, *5015*, *5020*, *5025*, *5030*, *5035*, *5040*, *5045*, *5050*, *5055*, *5060*, *5065*, *5070*, *5075*, *5080*, *5085*, *5090*, *5095*, *5100*, *5105*, *5110*, *5115*, *5120*, *5125*, *5130*, *5135*, *5140*, *5145*, *5150*, *5155*, *5160*, *5165*, *5170*, *5175*, *5180*, *5185*, *5190*, *5195*, *5200*, *5205*, *5210*, *5215*, *5220*, *5225*, *5230*, *5235*, *5240*, *5245*, *5250*, *5255*, *5260*, *5265*, *5270*, *5275*, *5280*, *5285*, *5290*, *5295*, *5300*, *5305*, *5310*, *5315*, *5320*, *5325*, *5330*, *5335*, *5340*, *5345*, *5350*, *5355*, *5360*, *5365*, *5370*, *5375*, *5380*, *5385*, *5390*, *5395*, *5400*, *5405*, *5410*, *5415*, *5420*, *5425*, *5430*, *5435*, *5440*, *5445*, *5450*, *5455*, *5460*, *5465*, *5470*, *5475*, *5480*, *5485*, *5490*, *5495*, *5500*, *5505*, *5510*, *5515*, *5520*, *5525*, *5530*, *5535*, *5540*, *5545*, *5550*, *5555*, *5560*, *5565*, *5570*, *5575*, *5580*, *5585*, *5590*, *5595*, *5600*, *5605*, *5610*, *5615*, *5620*, *5625*, *5630*, *5635*, *5640*, *5645*, *5650*, *5655*, *5660*, *5665*, *5670*, *5675*, *5680*, *5685*, *5690*, *5695*, *5700*, *5705*, *5710*, *5715*, *5720*, *5725*, *5730*, *5735*, *5740*, *5745*, *5750*, *5755*, *5760*, *5765*, *5770*, *5775*, *5780*, *5785*, *5790*, *5795*, *5800*, *5805*, *5810*, *5815*, *5820*, *5825*, *5830*, *5835*, *5840*, *5845*, *5850*, *5855*, *5860*, *5865*, *5870*, *5875*, *5880*, *5885*, *5890*, *5895*, *5900*, *5905*, *5910*, *5915*, *5920*, *5925*, *5930*, *5935*, *5940*, *5945*, *5950*, *5955*, *5960*, *5965*, *5970*, *5975*, *5980*, *5985*, *5990*, *5995*, *6000*, *6005*, *6010*, *6015*, *6020*, *6025*, *6030*, *6035*, *6040*, *6045*, *6050*, *6055*, *6060*, *6065*, *6070*, *6075*, *6080*, *6085*, *6090*, *6095*, *6100*) - the list of 5GHz-turbo channels (frequencies are given in MHz)

ack-timeout-control (*read-only: yes* | *no*) - provides information whether this device supports transmission acceptance timeout control

alignment-mode (*read-only: yes* | *no*) - is the alignment-only mode supported by this interface

burst-support (yes | no) - whether the interface supports data bursts (burst-time)

chip-info (*read-only: text*) - information from EEPROM

default-periodic-calibration (*read-only:* yes | no) - whether the card supports periodic-calibration

firmware (*read-only: text*) - current firmware of the interface (used only for Prism chipset based cards)

interface-type (*read-only: text*) - shows the hardware interface type

noise-floor-control (*read-only: yes* | *no*) - does this interface support noise-floor-thershold detection

nstreme-support (*read-only:* yes | no) - whether the card supports n-streme protocol

scan-support (yes | no) - whether the interface supports scan function ('/interface wireless scan')

supported-bands (*multiple choice*, *read-only: 2ghz-b*, *5ghz*, *5ghz-turbo*, *2ghz-g*) - the list of supported bands

tx-power-control (*read-only: yes* | *no*) - provides information whether this device supports transmission power control

virtual-aps (*read-only: yes* | *no*) - whether this interface supports Virtual Access Points ('/interface wireless add')

Notes

There is a special argument for the print command - print count-only. It forces the print command to print only the count of information topics.

/interface wireless info print command shows only channels supported by a particular card.

Example

```
[admin@MikroTik] interface wireless info> print
 0 interface-type=Atheros AR5413 chip-info="mac:0xa/0x5, phy:0x61, a5:0x63, a2:0x0,
eeprom:0x5002" pci-info="00:04.0"
capabilities=tx-power-control,ack-timeout-control,virtual-ap,alignment-mode,noise-floor-control,scanni
                power-channel,wmm
   default-periodic-calibration=enabled
supported-bands=2ghz-b,5ghz,5ghz-turbo,2ghz-g,2ghz-g-turbo,5ghz-10mhz,5ghz-5mhz,2ghz-10mhz,2ghz-5mhz
2ghz-b-channels=2192:0,2197:0,2202:0,2207:0,2212:0,2217:0,2222:0,2227:0,2232:0,2237:0,2242:0,2247:0,22
2287:0,2292:0,2297:0,2302:0,2307:0,2312:0,2317:0,2322:0,2327:0,2332:0,2337:0,2342:0,2347:0,2352:0,2357
2382:0,2387:0,2392:0,2397:0,2402:0,2407:0,2412:0,2417:0,2422:0,2427:0,2432:0,2437:0,2442:0,2447:0,2452
2477:0,2482:0,2487:0,2492:0,2497:0,2502:0,2507:0,2224:0,2229:0,2234:0,2239:0,2244:0,2249:0,2254:0,2259
2284:0,2289:0,2294:0,2299:0,2304:0,2309:0,2314:0,2319:0,2324:0,2329:0,2334:0,2339:0,2344:0,2349:0,2354
2379:0,2384:0,2389:0,2394:0,2399:0,2404:0,2409:0,2414:0,2419:0,2424:0,2429:0,2434:0,2439:0,2444:0,2449
2474:0,2479:0,2484:0,2489:0,2494:0,2499:0,2504:0,2509:0,2514:0,2519:0,2524:0,2529:0,2534:0,2539:0
5ghz-channels=4920:0,4925:0,4930:0,4935:0,4940:0,4945:0,4950:0,4955:0,4960:0,4965:0,4970:0,4975:0,4980
5015:0,5020:0,5025:0,5030:0,5035:0,5040:0,5045:0,5050:0,5055:0,5060:0,5065:0,5070:0,5075:0,5080:0,5085
5110:0,5115:0,5120:0,5125:0,5130:0,5135:0,5140:0,5145:0,5150:0,5155:0,5160:0,5165:0,5170:0,5175:0,5180
5205:0,5210:0,5215:0,5220:0,5225:0,5230:0,5235:0,5240:0,5245:0,5250:0,5255:0,5260:0,5265:0,5270:0,5275
5300:0,5305:0,5310:0,5315:0,5320:0,5325:0,5330:0,5335:0,5340:0,5345:0,5350:0,5355:0,5360:0,5365:0,5370
5395:0,5400:0,5405:0,5410:0,5415:0,5420:0,5425:0,5430:0,5435:0,5440:0,5445:0,5450:0,5455:0,5460:0,5465
5490:0,5495:0,5500:0,5505:0,5510:0,5515:0,5520:0,5525:0,5530:0,5535:0,5540:0,5545:0,5550:0,5555:0,5560
5585:0,5590:0,5595:0,5600:0,5605:0,5610:0,5615:0,5620:0,5625:0,5630:0,5635:0,5640:0,5645:0,5650:0,5655
5680:0,5685:0,5690:0,5695:0,5700:0,5705:0,5710:0,5715:0,5720:0,5725:0,5730:0,5735:0,5740:0,5745:0,5750
5775:0,5780:0,5785:0,5790:0,5795:0,5800:0,5805:0,5810:0,5815:0,5820:0,5825:0,5830:0,5835:0,5840:0,5845
5870:0,5875:0,5880:0,5885:0,5890:0,5895:0,5900:0,5905:0,5910:0,5915:0,5920:0,5925:0,5930:0,5935:0,5940
5965:0,5970:0,5975:0,5980:0,5985:0,5990:0,5995:0,6000:0,6005:0,6010:0,6015:0,6020:0,6025:0,6030:0,6035
                 6060:0,6065:0,6070:0,6075:0,6080:0,6085:0,6090:0,6095:0,6100:0
5ghz-turbo-channels=4920:0,4925:0,4930:0,4935:0,4940:0,4945:0,4950:0,4955:0,4960:0,4965:0,4970:0,4975
5015:0,5020:0,5025:0,5030:0,5035:0,5040:0,5045:0,5050:0,5055:0,5060:0,5065:0,5070:0,5075:0,5080:0,5085
5110:0,5115:0,5120:0,5125:0,5130:0,5135:0,5140:0,5145:0,5150:0,5155:0,5160:0,5165:0,5170:0,5175:0,5180
5205:0,5210:0,5215:0,5220:0,5225:0,5230:0,5235:0,5240:0,5245:0,5250:0,5255:0,5260:0,5265:0,5270:0,5275
5300:0,5305:0,5310:0,5315:0,5320:0,5325:0,5330:0,5335:0,5340:0,5345:0,5350:0,5355:0,5360:0,5365:0,5370
5395:0,5400:0,5405:0,5410:0,5415:0,5420:0,5425:0,5430:0,5435:0,5440:0,5445:0,5450:0,5455:0,5460:0,5465
5490:0,5495:0,5500:0,5505:0,5510:0,5515:0,5520:0,5525:0,5530:0,5535:0,5540:0,5545:0,5550:0,5555:0,5560
5585:0,5590:0,5595:0,5600:0,5605:0,5610:0,5615:0,5620:0,5625:0,5630:0,5635:0,5640:0,5645:0,5650:0,5655
5680:0,5685:0,5690:0,5695:0,5700:0,5705:0,5710:0,5715:0,5720:0,5725:0,5730:0,5735:0,5740:0,5745:0,5750
5775:0,5780:0,5785:0,5790:0,5795:0,5800:0,5805:0,5810:0,5815:0,5820:0,5825:0,5830:0,5835:0,5840:0,5845
5870:0,5875:0,5880:0,5885:0,5890:0,5895:0,5900:0,5905:0,5910:0,5915:0,5920:0,5925:0,5930:0,5935:0,5940
5965:0,5970:0,5975:0,5980:0,5985:0,5990:0,5995:0,6000:0,6005:0,6010:0,6015:0,6020:0,6025:0,6030:0,6035
                       6060:0,6065:0,6070:0,6075:0,6080:0,6085:0,6090:0,6095:0,6100:0
2ghz-g-channels=2192:0,2197:0,2202:0,2207:0,2212:0,2217:0,2222:0,2227:0,2232:0,2237:0,2242:0,2247:0,22
2287:0,2292:0,2297:0,2302:0,2307:0,2312:0,2317:0,2322:0,2327:0,2332:0,2337:0,2342:0,2347:0,2352:0,2357
2382:0,2387:0,2392:0,2397:0,2402:0,2407:0,2412:0,2417:0,2422:0,2427:0,2432:0,2437:0,2442:0,2447:0,2452
2477:0,2482:0,2487:0,2492:0,2497:0,2502:0,2507:0,2224:0,2229:0,2234:0,2239:0,2244:0,2249:0,2254:0,2259
2284:0,2289:0,2294:0,2299:0,2304:0,2309:0,2314:0,2319:0,2324:0,2329:0,2334:0,2339:0,2344:0,2349:0,2354
2379:0,2384:0,2389:0,2394:0,2399:0,2404:0,2409:0,2414:0,2419:0,2424:0,2429:0,2434:0,2439:0,2444:0,2449
2474:0,2479:0,2484:0,2489:0,2494:0,2499:0,2504:0,2509:0,2514:0,2519:0,2524:0,2529:0,2534:0,2539:0
2ghz-g-turbo-channels=2192:0,2197:0,2202:0,2207:0,2212:0,2217:0,2222:0,2227:0,2232:0,2237:0,2242:0,224
2282:0,2287:0,2292:0,2297:0,2302:0,2307:0,2312:0,2317:0,2322:0,2327:0,2332:0,2337:0,2342:0,2347:0,2352
2372:0,2377:0,2382:0,2387:0,2392:0,2397:0,2402:0,2407:0,2412:0,2417:0,2422:0,2427:0,2432:0,2437:0,2442
2462:0,2467:0,2472:0,2477:0,2482:0,2487:0,2492:0,2497:0,2502:0,2507:0,2224:0,2229:0,2234:0,2239:0,2244
2264:0,2269:0,2274:0,2279:0,2284:0,2289:0,2294:0,2299:0,2304:0,2309:0,2314:0,2319:0,2324:0,2329:0,2334
2354:0,2359:0,2364:0,2369:0,2374:0,2379:0,2384:0,2389:0,2394:0,2399:0,2404:0,2409:0,2414:0,2419:0,2424
2444:0,2449:0,2454:0,2459:0,2464:0,2469:0,2474:0,2479:0,2484:0,2489:0,2494:0,2499:0,2504:0,2509:0,2514
                         2534:0,2539:0
5ghz-10mhz-power-channels=4920:0,4925:0,4930:0,4935:0,4940:0,4945:0,4950:0,4955:0,4960:0,4965:0,4970:0
5010:0,5015:0,5020:0,5025:0,5030:0,5035:0,5040:0,5045:0,5050:0,5055:0,5060:0,5065:0,5070:0,5075:0,5080
5100:0,5105:0,5110:0,5115:0,5120:0,5125:0,5130:0,5135:0,5140:0,5145:0,5150:0,5155:0,5160:0,5165:0,5170
5190:0,5195:0,5200:0,5205:0,5210:0,5215:0,5220:0,5225:0,5230:0,5235:0,5240:0,5245:0,5250:0,5255:0,5260
5280:0,5285:0,5290:0,5295:0,5300:0,5305:0,5310:0,5315:0,5320:0,5325:0,5330:0,5335:0,5340:0,5345:0,5350
5370:0,5375:0,5380:0,5385:0,5390:0,5395:0,5400:0,5405:0,5410:0,5415:0,5420:0,5425:0,5430:0,5435:0,5440
5460:0,5465:0,5470:0,5475:0,5480:0,5485:0,5490:0,5495:0,5500:0,5505:0,5510:0,5515:0,5520:0,5525:0,5530
5550:0,5555:0,5560:0,5565:0,5570:0,5575:0,5580:0,5585:0,5590:0,5595:0,5600:0,5605:0,5610:0,5615:0,5620
5640:0,5645:0,5650:0,5655:0,5660:0,5665:0,5670:0,5675:0,5680:0,5685:0,5690:0,5695:0,5700:0,5705:0,5710
5730:0,5735:0,5740:0,5745:0,5750:0,5755:0,5760:0,5765:0,5770:0,5775:0,5780:0,5785:0,5790:0,5795:0,5800
5820:0,5825:0,5830:0,5835:0,5840:0,5845:0,5850:0,5855:0,5860:0,5865:0,5870:0,5875:0,5880:0,5885:0,5890
5910:0,5915:0,5920:0,5925:0,5930:0,5935:0,5940:0,5945:0,5950:0,5955:0,5960:0,5965:0,5970:0,5975:0,5980
6000:0,6005:0,6010:0,6015:0,6020:0,6025:0,6030:0,6035:0,6040:0,6045:0,6050:0,6055:0,6060:0,6065:0,6070
```

```
                             6090:0,6095:0,6100:0
5ghz-5mhz-power-channels=4920:0,4925:0,4930:0,4935:0,4940:0,4945:0,4950:0,4955:0,4960:0,4965:0,4
5010:0,5015:0,5020:0,5025:0,5030:0,5035:0,5040:0,5045:0,5050:0,5055:0,5060:0,5065:0,5070:0,5075:
5100:0,5105:0,5110:0,5115:0,5120:0,5125:0,5130:0,5135:0,5140:0,5145:0,5150:0,5155:0,5160:0,5165:
5190:0,5195:0,5200:0,5205:0,5210:0,5215:0,5220:0,5225:0,5230:0,5235:0,5240:0,5245:0,5250:0,5255:
5280:0,5285:0,5290:0,5295:0,5300:0,5305:0,5310:0,5315:0,5320:0,5325:0,5330:0,5335:0,5340:0,5345:
5370:0,5375:0,5380:0,5385:0,5390:0,5395:0,5400:0,5405:0,5410:0,5415:0,5420:0,5425:0,5430:0,5435:
5460:0,5465:0,5470:0,5475:0,5480:0,5485:0,5490:0,5495:0,5500:0,5505:0,5510:0,5515:0,5520:0,5525:
5550:0,5555:0,5560:0,5565:0,5570:0,5575:0,5580:0,5585:0,5590:0,5595:0,5600:0,5605:0,5610:0,5615:
5640:0,5645:0,5650:0,5655:0,5660:0,5665:0,5670:0,5675:0,5680:0,5685:0,5690:0,5695:0,5700:0,5705:
5730:0,5735:0,5740:0,5745:0,5750:0,5755:0,5760:0,5765:0,5770:0,5775:0,5780:0,5785:0,5790:0,5795:
5820:0,5825:0,5830:0,5835:0,5840:0,5845:0,5850:0,5855:0,5860:0,5865:0,5870:0,5875:0,5880:0,5885:
5910:0,5915:0,5920:0,5925:0,5930:0,5935:0,5940:0,5945:0,5950:0,5955:0,5960:0,5965:0,5970:0,5975:
6000:0,6005:0,6010:0,6015:0,6020:0,6025:0,6030:0,6035:0,6040:0,6045:0,6050:0,6055:0,6060:0,6065:
                            6090:0,6095:0,6100:0
2ghz-10mhz-power-channels=2192:0,2197:0,2202:0,2207:0,2212:0,2217:0,2222:0,2227:0,2232:0,2237:0,
2282:0,2287:0,2292:0,2297:0,2302:0,2307:0,2312:0,2317:0,2322:0,2327:0,2332:0,2337:0,2342:0,2347:
2372:0,2377:0,2382:0,2387:0,2392:0,2397:0,2402:0,2407:0,2412:0,2417:0,2422:0,2427:0,2432:0,2437:
2462:0,2467:0,2472:0,2477:0,2482:0,2487:0,2492:0,2497:0,2502:0,2507:0,2224:0,2229:0,2234:0,2239:
2264:0,2269:0,2274:0,2279:0,2284:0,2289:0,2294:0,2299:0,2304:0,2309:0,2314:0,2319:0,2324:0,2329:
2354:0,2359:0,2364:0,2369:0,2374:0,2379:0,2384:0,2389:0,2394:0,2399:0,2404:0,2409:0,2414:0,2419:
2444:0,2449:0,2454:0,2459:0,2464:0,2469:0,2474:0,2479:0,2484:0,2489:0,2494:0,2499:0,2504:0,2509:
                             2534:0,2539:0
2ghz-5mhz-power-channels=2192:0,2197:0,2202:0,2207:0,2212:0,2217:0,2222:0,2227:0,2232:0,2237:0,2
2282:0,2287:0,2292:0,2297:0,2302:0,2307:0,2312:0,2317:0,2322:0,2327:0,2332:0,2337:0,2342:0,2347:
2372:0,2377:0,2382:0,2387:0,2392:0,2397:0,2402:0,2407:0,2412:0,2417:0,2422:0,2427:0,2432:0,2437:
2462:0,2467:0,2472:0,2477:0,2482:0,2487:0,2492:0,2497:0,2502:0,2507:0,2224:0,2229:0,2234:0,2239:
2264:0,2269:0,2274:0,2279:0,2284:0,2289:0,2294:0,2299:0,2304:0,2309:0,2314:0,2319:0,2324:0,2329:
2354:0,2359:0,2364:0,2369:0,2374:0,2379:0,2384:0,2389:0,2394:0,2399:0,2404:0,2409:0,2414:0,2419:
2444:0,2449:0,2454:0,2459:0,2464:0,2469:0,2474:0,2479:0,2484:0,2489:0,2494:0,2499:0,2504:0,2509:
                            2534:0,2539:0
[admin@MikroTik] interface wireless>
```

Virtual Access Point Interface

Home menu level: ***/interface wireless***

Description

Virtual Access Point (VAP) interface is used to have an additional AP. You can create a new AP with different **ssid** and **mac-address**. It can be compared with a VLAN where the **ssid** from VAP is the VLAN **tag** and the hardware interface is the VLAN switch.

You can add up to 128 VAP interfaces for each hardware interface.

RouterOS supports VAP feature for Atheros AR5212 and newer.

Property Description

area (*text*; default: **""**) - string value that is used to describe an Access Point. Connect List on the Client's side comparing this string value with area-prefix string value makes decision whether allow a Client connect to the AP. If area-prefix match the entire area string or only the beginning of it the Client is allowed to connect to the AP

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*) - ARP mode

default-ap-tx-limit (*integer*; default: **0**) - limits data rate for each wireless client (in bps)

- **0** - no limits

default-authentication (*yes* | *no*; default: **yes**) - whether to accept or reject a client that wants to associate, but is not in the access-list

default-client-tx-limit (*integer*; default: **0**) - limits each client's transmit data rate (in bps). Works only if the client is also a MikroTik Router

- **0** - no limits

default-forwarding (*yes* | *no*; default: **yes**) - whether to forward frames to other AP clients or not

disable-running-check (*yes* | *no*; default: **no**) - disable running check. For 'broken' cards it is a good idea to set this value to 'yes'

disabled (*yes* | *no*; default: **yes**) - whether to disable the interface or not

hide-ssid (*yes* | *no*; default: **no**) - whether to hide ssid or not in the beacon frames:

- **yes** - ssid is not included in the beacon frames. AP replies only to probe-requests with the given ssid
- **no** - ssid is included in beacon frames. AP replies to probe-requests with the given ssid and to 'broadcast ssid'

mac-address (*MAC address*; default: **02:00:00:AA:00:00**) - MAC address of VAP. You can define your own value for mac-address

master-interface (*name*) - hardware interface to use for VAP

max-station-count (*integer*; default: **2007**) - number of clients that can connect to this AP simultaneously

mtu (*integer*: 68..1600; default: **1500**) - Maximum Transmission Unit

name (*name*; default: **wlanN**) - interface name

proprietary-extensions (*pre-2.9.25* | *post-2.9.25*; default: **post-2.9.25**) - the method to insert additional information (MikroTik proprietary extensions) into the wireless frames. This option is needed to workaround incompatibility between the old (pre-2.9.25) method and new Intel Centrino PCI-Express cards

- **pre-2.9.25** - include extensions in the form accepted by older RouterOS versions. This will include the new format as well, so this mode is compatiblewith all RouterOS versions. This mode is incompatible with wireless clients built on the new Centrino wireless chipset and may as well be incompatible with some other stations

security-profile (*text*; default: **default**) - which security profile to use. Define security profiles under /interface wireless security-profiles where you can setup WPA or WEP wireless security, for further details, see the Security Profiles section of this manual

ssid (*text*; default: **MikroTik**) - the service set identifier

update-stats-interval (*time*) - how often to update (request from the clients) signal strength and ccq values in /interface wireless registration-table

wds-cost-range (*integer*; default: **50-150**) - range, within which the bridge port cost of the WDS links are adjusted. The calculations are based on the p-throughput value of the respective WDS interface, which represents estimated approimate rhtoughput on the interface, which is mapped on the wds-cost-range scale so that bigger p-throughput would correspond to numerically lower port cost. The cost is recalculated every 20 seconds or when the p-throughput changes more than by 10% since the last recalculation

wds-default-bridge (*name*; default: **none**) - the default bridge for WDS interface. If you use dynamic WDS then it is very useful in cases when wds connection is reset - the newly created dynamic WDS interface will be put in this bridge

wds-default-cost (*integer*; default: **100**) - default bridge port cost of the WDS links

wds-ignore-ssid (yes | no; default: **no**) - if set to 'yes', the AP will create WDS links with any other

AP in this frequency. If set to 'no' the ssid values must match on both APs

wds-mode (*disabled* | *dynamic* | *static*) - WDS mode:

- **disabled** - WDS interfaces are disabled
- **dynamic** - WDS interfaces are created 'on the fly'
- **static** - WDS interfaces are created manually

wmm-support (*disabled* | *enabled* | *required*) - whether to allow (or require) peer to use WMM extensions to provide basic quality of service

Notes

The VAP MAC address is set by default to the same address as the physical interface has, with the second bit of the first byte set (i.e., the MAC address would start with 02). If that address is already used by some other wireless or VAP interface, it is increased by 1 until a free spot is found. When manually assigning MAC address, keep in mind that it should have the first bit of the first byte unset (so it should not be like 01, or A3). Note also that it is recommended to keep the MAC adress of VAP as similar (in terms of bit values) to the MAC address of the physical interface it is put onto, as possible, because the more different the addresses are, the more it affects performance.

WDS Interface Configuration

Home menu level: ***/interface wireless wds***

Description

WDS (Wireless Distribution System) allows packets to pass from one wireless AP (Access Point) to another, just as if the APs were ports on a wired Ethernet switch. APs must use the same standard (802.11a, 802.11b or 802.11g) and work on the same frequencies in order to connect to each other.

There are two possibilities to create a WDS interface:

- **dynamic** - is created 'on the fly' and appers under wds menu as a dynamic interface
- **static** - is created manually

Property Description

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; default: **enabled**) - Address Resolution Protocol

- **disabled** - the interface will not use ARP
- **enabled** - the interface will use ARP
- **proxy-arp** - the interface will use the ARP proxy feature
- **reply-only** - the interface will only reply to the requests originated to its own IP addresses. Neighbour MAC addresses will be resolved using /ip arp statically set table only

disable-running-check (*yes* | *no*; default: **no**) - disable running check. For 'broken' wireless cards it is a good idea to set this value to 'yes'

mac-address (*read-only: MAC address*; default: **00:00:00:00:00:00**) - MAC address of the master-interface. Specifying master-interface, this value will be set automatically

master-interface (*name*) - wireless interface which will be used by WDS

mtu (*integer*: 0..65336; default: **1500**) - Maximum Transmission Unit

name (*name*; default: **wdsN**) - WDS interface name

wds-address (*MAC address*) - MAC address of the remote WDS host

Notes

When the link between WDS devices, using **wds-mode=dynamic**, goes down, the dynamic WDS interfaces disappear and if there are any IP addresses set on this interface, their 'interface' setting will change to **(unknown)**. When the link comes up again, the 'interface' value will not change - it will remain as **(unknown)**. That's why it is not recommended to add IP addresses to dynamic WDS interfaces.

If you want to use dynamic WDS in a bridge, set the **wds-default-bridge** value to desired bridge interface name. When the link will go down and then it comes up, the dynamic WDS interface will be put in the specified bridge automatically.

As the routers which are in WDS mode have to communicate at equal frequencies, it is not recommended to use **WDS** and **DFS** simultaneously - it is most probable that these routers will not connect to each other.

WDS significantly faster than EoIP (up to 10-20% on RouterBOARD 500 systems), so it is recommended to use WDS whenever possible.

Example

```
[admin@MikroTik] interface wireless wds> add master-interface=wlan1 \
\... wds-address=00:0B:6B:30:2B:27 disabled=no
[admin@MikroTik] interface wireless wds> print
Flags: X - disabled, R - running, D - dynamic
  0  R  name="wds1" mtu=1500 mac-address=00:0B:6B:30:2B:23 arp=enabled
        disable-running-check=no master-inteface=wlan1
        wds-address=00:0B:6B:30:2B:27

[admin@MikroTik] interface wireless wds>
```

Align

Home menu level: ***/interface wireless align***

Description

This feature is created to position wireless links. The **align** submenu describes properties which are used if **/interface wireless mode** is set to **alignment-only**. In this mode the interface 'listens' to those packets which are sent to it from other devices working on the same channel. The interface also can send special packets which contains information about its parameters.

Property Description

active-mode (*yes* | *no*; default: **yes**) - whether the interface will receive and transmit 'alignment' packets or it will only receive them

audio-max (*integer*; default: **-20**) - signal-strength at which audio (beeper) frequency will be the highest

audio-min (*integer*; default: **-100**) - signal-strength at which audio (beeper) frequency will be the lowest

audio-monitor (*MAC address*; default: **00:00:00:00:00:00**) - MAC address of the remote host which will be 'listened'

filter-mac (*MAC address*; default: **00:00:00:00:00:00**) - in case if you want to receive packets from only one remote host, you should specify here its MAC address

frame-size (*integer*: 200..1500; default: **300**) - size of 'alignment' packets that will be transmitted

frames-per-second (*integer*: 1..100; default: **25**) - number of frames that will be sent per second (in active-mode)

receive-all (*yes* | *no*; default: **no**) - whether the interface gathers packets about other 802.11 standard packets or it will gather only 'alignment' packets

ssid-all (*yes* | *no*; default: **no**) - whether you want to accept packets from hosts with other ssid than yours

Command Description

test-audio (*integer*) - test the beeper for 10 seconds

Notes

If you are using the command **/interface wireless align monitor** then it will automatically change the wireless interface's mode from **station**, **bridge** or **ap-bridge** to **alignment-only**.

Example

```
[admin@MikroTik] interface wireless align> print
           frame-size: 300
          active-mode: yes
          receive-all: yes
        audio-monitor: 00:00:00:00:00:00
           filter-mac: 00:00:00:00:00:00
             ssid-all: no
    frames-per-second: 25
            audio-min: -100
            audio-max: -20
[admin@MikroTik] interface wireless align>
```

Align Monitor

Command name: ***/interface wireless align monitor***

Description

This command is used to monitor current signal parameters to/from a remote host.

Property Description

address (*read-only: MAC address*) - MAC address of the remote host

avg-rxq (*read-only: integer*) - average signal strength of received packets since last display update on screen

correct (*read-only: percentage*) - how many undamaged packets were received

last-rx (*read-only: time*) - time in seconds before the last packet was received

last-tx (*read-only: time*) - time in seconds when the last TXQ info was received

rxq (*read-only: integer*) - signal strength of last received packet

ssid (*read-only: text*) - service set identifier

txq (*read-only: integer*) - the last received signal strength from our host to the remote one

Example

```
[admin@MikroTik] interface wireless align> monitor wlan2
 # ADDRESS           SSID          RXQ AVG-RXQ LAST-RX TXQ LAST-TX CORRECT
 0 00:01:24:70:4B:FC wirelesa      -60 -60     0.01    -67 0.01    100 %

[admin@MikroTik] interface wireless align>
```

Frequency Monitor

Command name: ***/interface wireless frequency-monitor***

Description

Aproximately shows how loaded are the wireless channels.

Property Description

freq (*read-only: integer*) - shows current channel

use (*read-only: percentage*) - shows usage in current channel

Example

Monitor 802.11b network load:

```
[admin@MikroTik] interface wireless> frequency-monitor wlan1

FREQ          USE
2412MHz       3.8%
2417MHz       9.8%
2422MHz       2%
2427MHz       0.8%
2432MHz       0%
2437MHz       0.9%
2442MHz       0.9%
2447MHz       2.4%
2452MHz       3.9%
2457MHz       7.5%
2462MHz       0.9%
```

To monitor other bands, change the the **band** setting for the respective wireless interface.

Manual Transmit Power Table

Home menu level: ***/interface wireless manual-tx-power-table***

Description

In this submenu you can define signal strength for each rate. You should be aware that you can damage your wireless card if you set higher output power than it is allowed. Note that the values in this table are set in **dBm**! **NOT in mW!** Therefore this table is used mainly to reduce the transmit power of the card.

Property Description

manual-tx-powers (*text*) - define tx-power in dBm for each rate, separate by commas

Example

To set the following transmit powers at each rates: 1Mbps@10dBm, 2Mbps@10dBm, 5.5Mbps@9dBm, 11Mbps@7dBm, do the following:

```
[admin@MikroTik] interface wireless manual-tx-power-table> print
 0 name="wlan1" manual-tx-powers=1Mbps:17,2Mbps:17,5.5Mbps:17,11Mbps:17,6Mbps:17
,
                               9Mbps:17,12Mbps:17,18Mbps:17,24Mbps:17,
                               36Mbps:17,48Mbps:17,54Mbps:17

[admin@MikroTik] interface wireless manual-tx-power-table> set 0 \
   manual-tx-powers=1Mbps:10,2Mbps:10,5.5Mbps:9,11Mbps:7

[admin@MikroTik] interface wireless manual-tx-power-table> print
 0 name="wlan1" manual-tx-powers=1Mbps:10,2Mbps:10,5.5Mbps:9,11Mbps:7
[admin@MikroTik] interface wireless manual-tx-power-table>
```

Network Scan

Command name: ***/interface wireless scan interface_name***

Description

This is a feature that allows you to scan all avaliable wireless networks. While scanning, the card unregisters itself from the access point (in station mode), or unregisters all clients (in bridge or ap-bridge mode). Thus, network connections are lost while scanning.

Property Description

address (*read-only: MAC address*) - MAC address of the AP

band (*read-only: text*) - in which standard does the AP operate

bss (*read-only: yes* | *no*) - basic service set

freeze-time-interval (*time*; default: **1s**) - time in seconds to refresh the displayed data

freq (*read-only: integer*) - the frequency of AP

interface_name (*name*) - the name of interface which will be used for scanning APs

privacy (*read-only: yes* | *no*) - whether all data is encrypted or not

signal-strength (*read-only: integer*) - signal strength in dBm

ssid (*read-only: text*) - service set identifier of the AP

Example

Scan the 5GHz band:

```
[admin@MikroTik] interface wireless> scan wlan1
Flags: A - active, B - bss, P - privacy, R - routeros-network, N - nstreme
      ADDRESS           SSID              BAND       FREQ SIG RADIO-NAME
AB R  00:0C:42:05:00:28 test              5ghz       5180 -77 000C42050028
AB R  00:02:6F:20:34:82 aap1              5ghz       5180 -73 00026F203482
AB    00:0B:6B:30:80:0F www               5ghz       5180 -84
AB R  00:0B:6B:31:B6:D7 www               5ghz       5180 -81 000B6B31B6D7
AB R  00:0B:6B:33:1A:D5 R52_test_new      5ghz       5180 -79 000B6B331AD5
AB R  00:0B:6B:33:0D:EA short5            5ghz       5180 -70 000B6B330DEA
AB R  00:0B:6B:31:52:69 MikroTik          5ghz       5220 -69 000B6B315269
AB R  00:0B:6B:33:12:BF long2             5ghz       5260 -55 000B6B3312BF
-- [Q quit|D dump|C-z pause]
[admin@MikroTik] interface wireless>
```

Security Profiles

Home menu level: ***/interface wireless security-profiles***

Description

This section provides WEP (Wired Equivalent Privacy) and WPA/WPA2 (Wi-Fi Protected Access) functions to wireless interfaces.

WPA

The Wi-Fi Protected Access is a combination of 802.1X, EAP, MIC, TKIP and AES. This is a easy to configure and secure wireless mechanism. It has been later updated to version 2, to provide greater security.

Pairwise master key caching for EAP authentification is supported for WPA2. This means that disconnected client can connect without repeated EAP authentication if keys are still valid (changed to interface or security profile configuration, restart, or Session-Timeout in case of RADIUS authentication).

WEP

The Wired Equivalent Privacy encrypts data only between 802.11 devices, using static keys. It is not considered a very secure wireless data encryption mechanism, though it is better than no encryption at all.

The configuration of WEP is quite simple, using MikroTik RouterOS security profiles.

Property Description

authentication-types (*multiple choice: wpa-psk | wpa2-psk | wpa-eap | wpa2-eap*; default: "") - the list of accepted authentication types. APs will advertise the listed types. Stations will choose the AP, which supports the "best" type from the list (WPA2 is always preferred to WPA1; EAP is preferred to PSK)

eap-methods (*multiple choice: eap-tls | passthrough*) - the ordered list of EAP methods. APs will to propose to the stations one by one (if first method listed is rejected, the next one is tried). Stations will accept first proposed method that will be on the list

- **eap-tls** - Use TLS certificates for authentication
- **passthrough** - relay the authentication process to the RADIUS server (not used by the stations)

group-ciphers (*multiple choice: tkip | aes-ccm*) - a set of ciphers used to encrypt frames sent to all wireless station (broadcast transfers) in the order of preference

- **tkip** - Temporal Key Integrity Protocol - encryption protocol, compatible with lagacy WEP equipment, but enhanced to correct some of WEP flaws
- **aes-ccm** - more secure WPA encryption protocol, based on the reliable AES (Advanced Encryption Standard). Networks free of WEP legacy should use only this

group-key-update (*time*; default: **5m**) - how often to update group key. This parameter is used only if the wireless card is configured as an Access Point

interim-update (*time*) - default update interval for RADIUS accounting, if RADIUS server has not provided different value

mode (*none* | *static-keys-optional* | *static-keys-required* | *dynamic-keys*; default: **none**) - security mode:

- **none** - do not encrypt packets and do not accept encrypted packets
- **static-keys-optional** - if there is a static-sta-private-key set, use it. Otherwise, if the interface is set in an AP mode, do not use encryption, if the the interface is in station mode, use encryption if the static-transmit-key is set
- **static-keys-required** - encrypt all packets and accept only encrypted packets
- **dynamic-keys** - generate encryptioon keys dynamically

name (*name*) - descriptive name for the security profile

radius-eap-accounting (yes | no; default: **no**) - use RADUIS accounting if EAP authentication is used

radius-mac-accounting (yes | no; default: **no**) - use RADIUS accounting, providing MAC address as username

radius-mac-authentication (*no* | *yes*; default: **no**) - whether to use RADIUS server for MAC authentication

radius-mac-caching (*time*; default: **disabled**) - how long the RADIUS authentication reply for MAC address authentication if considered valid (and thus can be cached for faster reauthentication)

radius-mac-format (*text*; default: **XX:XX:XX:XX:XX:XX**) - MAC address format to use for communication with RADIUS server

radius-mac-mode (*as-username* | *as-username-and-password*; default: **as-username**) - whether to use MAC address as username only or ad both username and password for RADIUS authentication

static-algo-0 (*none* | *40bit-wep* | *104bit-wep* | *aes-ccm* | *tkip*; default: **none**) - which encryption algorithm to use:

- **none** - do not use encryption and do not accept encrypted packets
- **40bit-wep** - use the 40bit encryption (also known as 64bit-wep) and accept only these packets
- **104bit-wep** - use the 104bit encryption (also known as 128bit-wep) and accept only these packets
- **aes-ccm** - use the AES-CCM (Advanced Encryption Standard in Counter with CBC-MAC) encryption algorithm and accept only these packets
- **tkip** - use the TKIP (Temporal Key Integrity Protocol) and accept only these packets

static-algo-1 (*none* | *40bit-wep* | *104bit-wep* | *aes-ccm* | *tkip*; default: **none**) - which encryption algorithm to use:

- **none** - do not use encryption and do not accept encrypted packets
- **40bit-wep** - use the 40bit encryption (also known as 64bit-wep) and accept only these packets
- **104bit-wep** - use the 104bit encryption (also known as 128bit-wep) and accept only these packets
- **aes-ccm** - use the AES-CCM (Advanced Encryption Standard in Counter with CBC-MAC) encryption algorithm and accept only these packets
- **tkip** - use the TKIP (Temporal Key Integrity Protocol) and accept only these packets

static-algo-2 (*none* | *40bit-wep* | *104bit-wep* | *aes-ccm* | *tkip*; default: **none**) - which encryption algorithm to use:

- **none** - do not use encryption and do not accept encrypted packets
- **40bit-wep** - use the 40bit encryption (also known as 64bit-wep) and accept only these packets
- **104bit-wep** - use the 104bit encryption (also known as 128bit-wep) and accept only these packets
- **aes-ccm** - use the AES-CCM (Advanced Encryption Standard in Counter with CBC-MAC) encryption algorithm and accept only these packets
- **tkip** - use the TKIP (Temporal Key Integrity Protocol) and accept only these packets

static-algo-3 (*none* | *40bit-wep* | *104bit-wep* | *aes-ccm* | *tkip*; default: **none**) - which encryption algorithm to use:

- **none** - do not use encryption and do not accept encrypted packets
- **40bit-wep** - use the 40bit encryption (also known as 64bit-wep) and accept only these packets
- **104bit-wep** - use the 104bit encryption (also known as 128bit-wep) and accept only these packets
- **aes-ccm** - use the AES-CCM (Advanced Encryption Standard in Counter with CBC-MAC) encryption algorithm and accept only these packets
- **tkip** - use the TKIP (Temporal Key Integrity Protocol) and accept only these packets

static-key-0 (*text*) - hexadecimal key which will be used to encrypt packets with the 40bit-wep or 104bit-wep algorithm (algo-0). If AES-CCM is used, the key must consist of even number of characters and must be at least 32 characters long. For TKIP, the key must be at least 64 characters long and also must consist of even number characters

static-key-1 (*text*) - hexadecimal key which will be used to encrypt packets with the 40bit-wep or 104bit-wep algorithm (algo-1). If AES-CCM is used, the key must consist of even number of characters and must be at least 32 characters long. For TKIP, the key must be at least 64 characters long and also must consist of even number characters

static-key-2 (*text*) - hexadecimal key which will be used to encrypt packets with the 40bit-wep or 104bit-wep algorithm (algo-2). If AES-CCM is used, the key must consist of even number of characters and must be at least 32 characters long. For TKIP, the key must be at least 64 characters long and also must consist of even number characters

static-key-3 (*text*) - hexadecimal key which will be used to encrypt packets with the 40bit-wep or 104bit-wep algorithm (algo-3). If AES-CCM is used, the key must consist of even number of characters and must be at least 32 characters long. For TKIP, the key must be at least 64 characters long and also must consist of even number characters

static-sta-private-algo (*none* | *40bit-wep* | *104bit-wep* | *aes-ccm* | *tkip*) - algorithm to use if the static-sta-private-key is set. Used to commumicate between 2 devices

static-sta-private-key (*text*) - if this key is set in station mode, use this key for encryption. In AP mode you have to specify static-private keys in the access-list or use the Radius server using radius-mac-authentication. Used to commumicate between 2 devices

static-transmit-key (*static-key-0* | *static-key-1* | *static-key-2* | *static-key-3*; default: **static-key-0**) - which key to use for broadcast packets. Used in AP mode

supplicant-identity (*text*; default: **MikroTik**) - EAP supplicant identity to use for RADIUS EAP authentication

tls-certificate (*name*) - select the certificate for this device from the list of imported certificates

tls-mode (*no-certificates* | *dont-verify-certificate* | *verify-certificate*; default: **no-certificates**) - TLS certificate mode

- **no-certificates** - certificates are negotiated dynamically using anonymous Diffie-Hellman MODP 2048 bit algorithm
- **dont-verify-certificate** - require a certificate, but do not chack, if it has been signed by the available CA certificate
- **verify-certificate** - require a certificate and verify that it has been signed by the available CA certificate

unicast-ciphers (*multiple choice: tkip* | *aes-ccm*) - a set of ciphers used to encrypt frames sent to individual wireless station (unicast transfers) in the order of preference

- **tkip** - Temporal Key Integrity Protocol - encryption protocol, compatible with lagacy WEP equipment, but enhanced to correct some of WEP flaws
- **aes-ccm** - more secure WPA encryption protocol, based on the reliable AES (Advanced Encryption Standard). Networks free of WEP legacy should use only this

wpa-pre-shared-key (*text*; default: **""**) - string, which is used as the WPA Pre Shared Key. It must be the same on AP and station to communicate

wpa2-pre-shared-key (*text*; default: **""**) - string, which is used as the WPA2 Pre Shared Key. It must be the same on AP and station to communicate

Notes

The keys used for encryption are in hexadecimal form. If you use **40bit-wep**, the key has to be 10 characters long, if you use **104bit-wep**, the key has to be 26 characters long.

Prism card doesn't report that the use of WEP is required for all data type frames, which means that some clients will not see that access point uses encryption and will not be able to connect to such AP. This is a Prism hardware problem and can not be fixed. Use Atheros-based cards (instead of Prism) on APs if you want to provide WEP in your wireless network.

Wireless encryption cannot work together with wireless compression.

Sniffer

Home menu level: ***/interface wireless sniffer***

Description

With wireless sniffer you can sniff packets from wireless networks.

Property Description

channel-time (*time*; default: **200ms**) - how long to sniff each channel, if multiple-channels is set to yes

file-limit (*integer*; default: **10**) - limits file-name's file size (measured in kilobytes)

file-name (*text*; default: **""**) - name of the file where to save packets in PCAP format. If file-name is not defined, packets are not saved into a file

memory-limit (*integer*; default: **1000**) - how much memory to use (in kilobytes) for sniffed packets

multiple-channels (yes | no; default: **no**) - whether to sniff multiple channels or a single channel

- **no** - wireless sniffer sniffs only one channel in frequency that is configured in /interface wireless
- **yes** - sniff in all channels that are listed in the scan-list in /interface wireless

only-headers (yes | no; default: **no**) - sniff only wireless packet heders

receive-errors (yes | no; default: **no**) - whether to receive packets with CRC errors

streaming-enabled (yes | no; default: **no**) - whether to send packets to server in TZSP format

streaming-max-rate (*integer*; default: **0**) - how many packets per second the router will accept

- **0** - no packet per second limitation

streaming-server (*IP address*; default: **0.0.0.0**) - streaming server's IP address

Sniffer Sniff

Home menu level: ***/interface wireless sniffer sniff***

Description

Wireless Sniffer Sniffs packets

Property Description

file-over-limit-packets (*read-only: integer*) - how many packets are dropped because of exceeding file-limit

file-saved-packets (*read-only: integer*) - number of packets saved to file

file-size (*read-only: integer*) - current file size (kB)

memory-over-limit-packets (*read-only: integer*) - number of packets that are dropped because of exceeding memory-limit

memory-saved-packets (*read-only: integer*) - how many packets are stored in mermory

memory-size (*read-only: integer*) - how much memory is currently used for sniffed packets (kB)

processed-packets (*read-only: integer*) - number of sniffed packets

real-file-limit (*read-only: integer*) - the real file size limit. It is calculated from the beginning of sniffing to reserve at least 1MB free space on the disk

real-memory-limit (*read-only: integer*) - the real memory size limit. It is calculated from the beginning of sniffing to reserve at least 1MB of free space in the memory

stream-dropped-packets (*read-only: integer*) - number of packets that are dropped because of exceeding streaming-max-rate

stream-sent-packets (*read-only: integer*) - number of packets that are sent to the streaming server

Command Description

save - saves sniffed packets from the memory to file-name in PCAP format

Sniffer Packets

Description

Wireless Sniffer sniffed packets. If packets Cyclic Redundancy Check (CRC) field detects error, it will be displayed by crc-error flag.

Property Description

band (*read-only: text*) - wireless band

dst (*read-only: MAC address*) - the receiver's MAC address

freq (*read-only: integer*) - frequency

interface (*read-only: text*) - wireless interface that captures packets

signal@rate (*read-only: text*) - at which signal-strength and rate was the packet received

src (*read-only: MAC address*) - the sender's MAC address

time (*read-only: time*) - time when the packet was received, starting from the beginning of sniffing

type (*read-only: assoc-req* | *assoc-resp* | *reassoc-req* | *reassoc-resp* | *probe-req* | *probe-resp* | *beacon* | *atim* | *disassoc* | *auth* | *deauth* | *ps-poll* | *rts* | *cts* | *ack* | *cf-end* | *cf-endack* | *data* | *d-cfack* | *d-cfpoll* | *d-cfackpoll* | *data-null* | *nd-cfack* | *nd-cfpoll* | *nd-cfackpoll*) - type of the sniffed packet

Example

Sniffed packets:

```
[admin@MikroTik] interface wireless sniffer packet> pr
Flags: E - crc-error
 #   FREQ SIGNAL@RATE    SRC               DST               TYPE
 0   2412 -73dBm@1Mbps   00:0B:6B:31:00:53 FF:FF:FF:FF:FF:FF beacon
 1   2412 -91dBm@1Mbps   00:02:6F:01:CE:2E FF:FF:FF:FF:FF:FF beacon
 2   2412 -45dBm@1Mbps   00:02:6F:05:68:D3 FF:FF:FF:FF:FF:FF beacon
 3   2412 -72dBm@1Mbps   00:60:B3:8C:98:3F FF:FF:FF:FF:FF:FF beacon
 4   2412 -65dBm@1Mbps   00:01:24:70:3D:4E FF:FF:FF:FF:FF:FF probe-req
 5   2412 -60dBm@1Mbps   00:01:24:70:3D:4E FF:FF:FF:FF:FF:FF probe-req
 6   2412 -61dBm@1Mbps   00:01:24:70:3D:4E FF:FF:FF:FF:FF:FF probe-req
```

Snooper

Home menu level: ***/interface wireless snooper***

Description

With wireless snooper you can monitor the traffic load on each channel.

Property Description

channel-time (*time*; default: **200ms**) - how long to snoop each channel, if multiple-channels is set to yes

multiple-channels (yes | no; default: **no**) - whether to snoop multiple channels or a single channel

- **no** - wireless snooper snoops only one channel in frequency that is configured in /interface wireless
- **yes** - snoop in all channels that are listed in the scan-list in /interface wireless

receive-errors (yes | no; default: **no**) - whether to receive packets with CRC errors

Command Description

snoop - starts monitoring wireless channels

- **wireless interface name** - interface that monitoring is performed on
- **BAND** - operating band

Example

Snoop 802.11b network:

```
[admin@MikroTik] interface wireless snooper> snoop wlan1
BAND        FREQ    USE    BW         NET-COUNT STA-COUNT
2.4ghz-b    2412MHz 1.5%   11.8kbps   2         2
2.4ghz-b    2417MHz 1.3%   6.83kbps   0         1
2.4ghz-b    2422MHz 0.6%   4.38kbps   1         1
2.4ghz-b    2427MHz 0.6%   4.43kbps   0         0
2.4ghz-b    2432MHz 0.3%   2.22kbps   0         0
2.4ghz-b    2437MHz 0%     0bps       0         0
2.4ghz-b    2442MHz 1%     8.1kbps    0         0
2.4ghz-b    2447MHz 1%     8.22kbps   1         1
2.4ghz-b    2452MHz 1%     8.3kbps    0         0
2.4ghz-b    2457MHz 0%     0bps       0         0
2.4ghz-b    2462MHz 0%     0bps       0         0

[admin@MikroTik] interface wireless snooper>
```

Xpeed SDSL Interface

Document revision 1.2 (February 6, 2008, 2:56 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS supports the Xpeed 300 SDSL PCI Adapter hardware with speeds up to 2.32Mbps. This device can operate either using Frame Relay or PPP type of connection. SDSL (Single-line Digital Subscriber Line or Symmetric Digital Subscriber Line) stands for the type of DSL that uses only one of the two cable pairs for transmission. SDSL allows residential or small office users to share the same telephone for data transmission and voice or fax telephony.

Specifications

Packages required: ***synchronous***
License required: ***level4***
Home menu level: ***/interface xpeed***
Standards and Technologies: ***PPP (RFC 1661)***, ***Frame Relay (RFC 1490)***
Hardware usage: ***Not significant***

Additional Documents

- Xpeed homepage

Xpeed Interface Configuration

Home menu level: ***/interface xpeed***

Property Description

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; default: **enabled**) - Address Resolution Protocol

- **disabled** - the interface will not use ARP protocol
- **enabled** - the interface will use ARP protocol
- **proxy-arp** - the interface will be an ARP proxy
- **reply-only** - the interface will only reply to the requests originated to its own IP addresses, but neighbor MAC addresses will be gathered from /ip arp statically set table only

bridged-ethernet (*yes* | *no*; default: **yes**) - if the adapter operates in bridged Ethernet mode

cr (*0* | *2*; default: **0**) - a special mask value to be used when speaking with certain buggy vendor equipment. Can be 0 or 2

dlci (*integer*; default: **16**) - defines the DLCI to be used for the local interface. The DLCI field identifies which logical circuit the data travels over

lmi-mode (*off* | *line-termination* | *network-termination* | *network-termination-bidirectional*; default: **off**) - defines how the card will perform LMI protocol negotiation

- **off** - no LMI will be used
- **line-termination** - LMI will operate in LT (Line Termination) mode
- **network-termination** - LMI will operate in NT (Network Termination) mode
- **network-termination-bidirectional** - LMI will operate in bidirectional NT mode

mac-address (*MAC address*) - MAC address of the card

mode (*network-termination* | *line-termination*; default: **line-termination**) - interface mode, either line termination (LT) or network termination (NT)

mtu (*integer*; default: **1500**) - Maximum Transmission Unit

name (*name*) - interface name

sdsl-invert (*yes* | *no*; default: **no**) - whether the clock is phase inverted with respect to the Transmitted Data interchange circuit. This configuration option is useful when long cable lengths between the Termination Unit and the DTE are causing data errors

sdsl-speed (*integer*; default: **2320**) - SDSL connection speed

sdsl-swap (*yes* | *no*; default: **no**) - whether or not the Xpeed 300 SDSL Adapter performs bit swapping. Bit swapping can maximize error performance by attempting to maintain an acceptable margin for each bin by equalizing the margin across all bins through bit reallocation

Example

To enable interface:

```
[admin@r1] interface> print
Flags: X - disabled, R - running, D - dynamic, S - slave
 #    NAME                                              TYPE             MTU
 0 R  outer                                             ether            1500
 1 R  inner                                             ether            1500
 2 X  xpeed1                                            xpeed            1500
[admin@r1] interface> enable 2
[admin@r1] interface> print
Flags: X - disabled, R - running, D - dynamic, S - slave
 #    NAME                                              TYPE             MTU
 0 R  outer                                             ether            1500
 1 R  inner                                             ether            1500
 2 R  xpeed1                                            xpeed            1500
[admin@r1] interface>
```

Frame Relay Configuration Examples

MikroTik Router to MikroTik Router

Consider the following network setup with MikroTik router connected via SDSL line using Xpeed interface to another MikroTik router with Xpeed 300 SDSL adapter. SDSL line can refer a common patch cable included with the Xpeed 300 SDSL adapter (such a connection is called Back-to-Back). Lets name the first router **r1** and the second **r2**.

Router **r1** setup

The following setup is identical to one in the first example:

```
[admin@r1] ip address> add inter=xpeed1 address 1.1.1.1/24
[admin@r1] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   1.1.1.1/24         1.1.1.0         1.1.1.255       xpeed1

[admin@r1] interface xpeed> print
Flags: X - disabled
  0   name="xpeed1" mtu=1500 mac-address=00:05:7A:00:00:08 arp=enabled
      mode=network-termination sdsl-speed=2320 sdsl-invert=no sdsl-swap=no
      bridged-ethernet=yes dlci=16 lmi-mode=off cr=0
[admin@r1] interface xpeed>
```

Router **r2** setup

First, we need to add a suitable IP address:

```
[admin@r2] ip address> add inter=xpeed1 address 1.1.1.2/24
[admin@r2] ip address> pri
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   1.1.1.2/24         1.1.1.0         1.1.1.255       xpeed1
```

Then, some changes in **xpeed** interface configuration should be done:

```
[admin@r2] interface xpeed> print
Flags: X - disabled
  0   name="xpeed1" mtu=1500 mac-address=00:05:7A:00:00:08 arp=enabled
      mode=network-termination sdsl-speed=2320 sdsl-invert=no sdsl-swap=no
      bridged-ethernet=yes dlci=16 lmi-mode=off cr=0
[admin@r2] interface xpeed> set 0 mode=line-termination
[admin@r2] interface xpeed>
```

Now **r1** and **r2** can ping each other.

MikroTik Router to Cisco Router

Let us consider the following network setup with MikroTik Router with Xpeed interface connected to a leased line with a CISCO router at the other end.

MikroTik router setup:

```
[admin@r1] ip address> add inter=xpeed1 address 1.1.1.1/24
[admin@r1] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   1.1.1.1/24         1.1.1.0         1.1.1.255       xpeed1
```

```
[admin@r1] interface xpeed> print
Flags: X - disabled
  0   name="xpeed1" mtu=1500 mac-address=00:05:7A:00:00:08 arp=enabled
      mode=network-termination sdsl-speed=2320 sdsl-invert=no sdsl-swap=no
      bridged-ethernet=yes dlci=42 lmi-mode=off cr=0
[admin@r1] interface xpeed>
```

Cisco router setup

```
CISCO# show running-config
Building configuration...
Current configuration...

...
!
ip subnet-zero
no ip domain-lookup
frame-relay switching
!
interface Ethernet0
 description connected to EthernetLAN
 ip address 10.0.0.254 255.255.255.0
!
interface Serial0
 description connected to Internet
 no ip address
 encapsulation frame-relay IETF
 serial restart-delay 1
 frame-relay lmi-type ansi
 frame-relay intf-type dce
!
interface Serial0.1 point-to-point
 ip address 1.1.1.2 255.255.255.0
 no arp frame-relay
 frame-relay interface-dlci 42
!
...
end.

Send ping to MikroTik router

CISCO#ping 1.1.1.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/31/32 ms
CISCO#
```

Troubleshooting

Description

- **I tried to connect two routers as shown in MT-to-MT, but nothing happens**
  The link indicators on both cards must be on. If it's not, check the cable or interface configuration. One adapter should use LT mode and the other NT mode. You can also change **sdsl-swap** and **sdsl-invert** parameters on the router running LT mode if you have a very long line

EoIP

Document revision 1.5 (September 11, 2007, 9:06 GMT)

This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Ethernet over IP (EoIP) Tunneling is a MikroTik RouterOS protocol that creates an Ethernet tunnel between two routers on top of an IP connection. The EoIP interface appears as an Ethernet interface. When the bridging function of the router is enabled, all Ethernet traffic (all Ethernet protocols) will be bridged just as if there where a physical Ethernet interface and cable between the two routers (with bridging enabled). This protocol makes multiple network schemes possible.

Network setups with EoIP interfaces:

- Possibility to bridge LANs over the Internet
- Possibility to bridge LANs over encrypted tunnels
- Possibility to bridge LANs over 802.11b 'ad-hoc' wireless networks

Quick Setup Guide

To make an EoIP tunnel between 2 routers which have IP addresses **10.5.8.1** and **10.1.0.1**:

1. On router with IP address **10.5.8.1**, add an EoIP interface and set its MAC address:

```
/interface eoip add remote-address=10.1.0.1 tunnel-id=1 mac-address=00-00-5E-80-00-01 \
\... disabled=no
```

2. On router with IP address **10.1.0.1**, add an EoIP interface and set its MAC address::

```
/interface eoip add remote-address=10.5.8.1 tunnel-id=1 mac-address=00-00-5E-80-00-02 \
\... disabled=no
```

Now you can add IP addresses to the created EoIP interfaces from the same subnet.

Specifications

Packages required: ***system***
License required: ***level1 (limited to 1 tunnel), level3***
Home menu level: ***/interface eoip***
Standards and Technologies: ***GRE (RFC1701)***
Hardware usage: ***Not significant***

Description

EoIP interface may be configured between two routers that have active IP level connection. The EoIP tunnel may run over an IPIP tunnel, a PPTP 128bit encrypted tunnel, a PPPoE connection, or any other connection that transports IP.

Specific Properties:

- Each EoIP tunnel interface can connect with one remote router which has a corresponding interface configured with the same 'Tunnel ID'.
- The EoIP interface appears as an Ethernet interface under the interface list.
- This interface supports all features of an Ethernet interface. IP addresses and other tunnels may be run over the interface.
- The EoIP protocol encapsulates Ethernet frames in GRE (IP protocol number 47) packets (just like PPTP) and sends them to the remote side of the EoIP tunnel.
- Maximal number of EoIP tunnels is 65536.

Notes

WDS significantly faster than EoIP on wireless links (up to 10-20% on RouterBOARD 500 systems), so it is recommended to use WDS whenever possible.

EoIP Setup

Home menu level: ***/interface eoip***

Property Description

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; default: **enabled**) - Address Resolution Protocol

mac-address (*MAC address*) - MAC address of the EoIP interface. The address numeration authority allows to use MAC addresses in the range from 00:00:5E:80:00:00 to 00:00:5E:FF:FF:FF freely. Other addresses can be used, but not recommended. You should keep the MAC addresses unique within one bridged network

mtu (*integer*; default: **1500**) - Maximum Transmission Unit. The default value provides maximal compatibility, although it may lead to decreasing performance on wireless links due to fragmentation. If you can increase MTU on all links inbetween, you may be able to regain optimal performance

name (*name*; default: **eoip-tunnelN**) - interface name for reference

remote-address - the IP address of the other side of the EoIP tunnel - must be a MikroTik router

tunnel-id (*integer*) - a unique tunnel identifier, which must match th other side of the tunnel

Notes

tunnel-id is method of identifying tunnel. There should not be tunnels with the same **tunnel-id** on the same router. **tunnel-id** on both participant routers must be equal.

mtu should be set to 1500 to eliminate packet refragmentation inside the tunnel (that allows transparent bridging of Ethernet-like networks, so that it would be possible to transport full-sized Ethernet frame over the tunnel).

When bridging EoIP tunnels, it is highly recommended to set unique MAC addresses for each tunnel for the bridge algorithms to work correctly. For EoIP interfaces you can use MAC addresses that are in the range from **00-00-5E-80-00-00** to **00-00-5E-FF-FF-FF**, which IANA has reserved for such cases. Alternatively, you can set the second bit of the first byte to mark the address as locally administered address, assigned by network administrator, and use any MAC address, you just need to ensure they are unique between the hosts connected to one bridge.

Example

To add and enable an EoIP tunnel named **to_mt2** to the **10.5.8.1** router, specifying **tunnel-id** of **1**:

```
[admin@MikroTik] interface eoip> add name=to_mt2 remote-address=10.5.8.1 \
\... tunnel-id 1
[admin@MikroTik] interface eoip> print
Flags: X - disabled, R - running
  0 X  name="to_mt2" mtu=1500 arp=enabled remote-address=10.5.8.1 tunnel-id=1

[admin@MikroTik] interface eoip> enable 0
[admin@MikroTik] interface eoip> print
Flags: X - disabled, R - running
  0 R  name="to_mt2" mtu=1500 arp=enabled remote-address=10.5.8.1 tunnel-id=1

[admin@MikroTik] interface eoip>
```

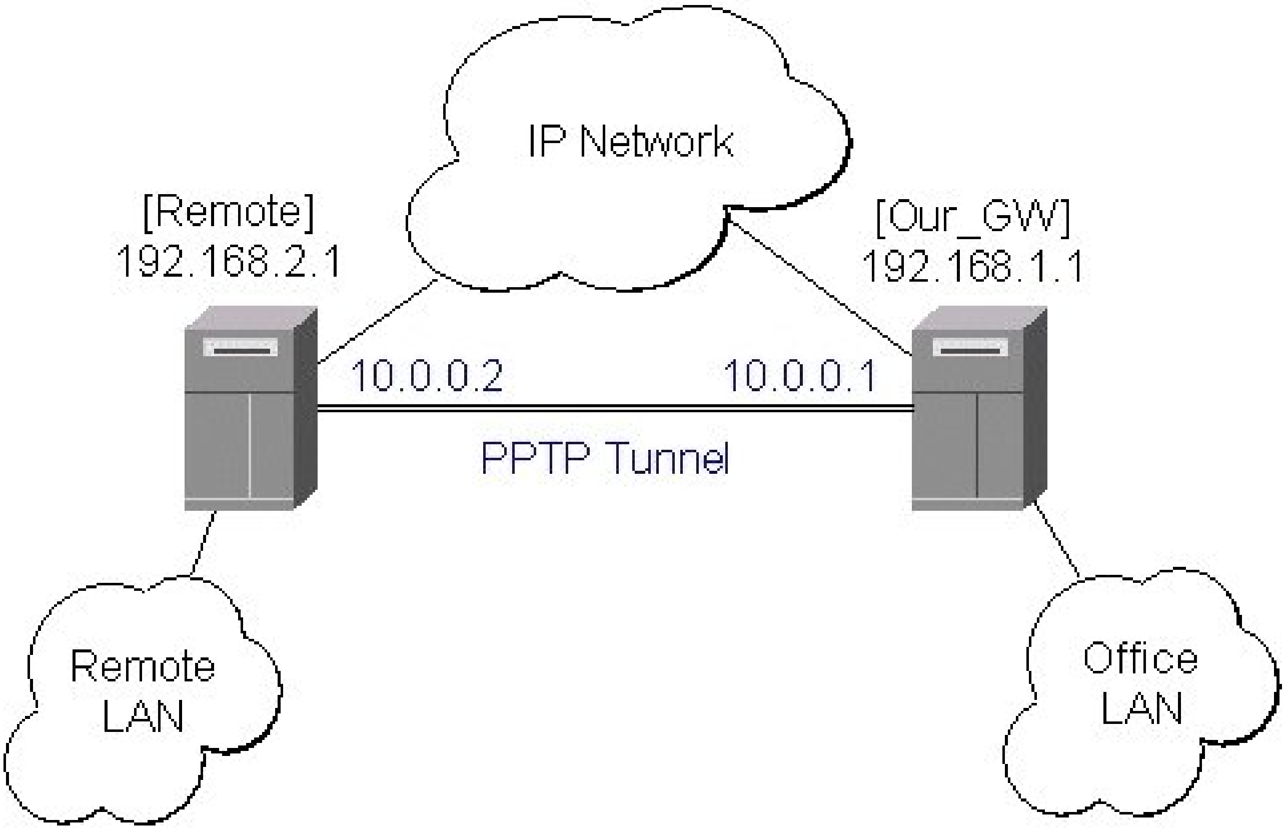
EoIP Application Example

Description

Let us assume we want to bridge two networks: 'Office LAN' and 'Remote LAN'. The networks are connected to an IP network through the routers [Our_GW] and [Remote]. The IP network can be a private intranet or the Internet. Both routers can communicate with each other through the IP network.

Example

Our goal is to create a secure channel between the routers and bridge both networks through it. The network setup diagram is as follows:



To make a secure Ethernet bridge between two routers you should:

1. Create a PPTP tunnel between them. Our_GW will be the pptp server:

```
[admin@Our_GW] interface pptp-server> /ppp secret add name=joe service=pptp \
\... password=top_s3 local-address=10.0.0.1 remote-address=10.0.0.2
[admin@Our_GW] interface pptp-server> add name=from_remote user=joe
[admin@Our_GW] interface pptp-server> server set enable=yes
[admin@Our_GW] interface pptp-server> print
Flags: X - disabled, D - dynamic, R - running
 #     NAME         USER         MTU        CLIENT-AD... UPTIME   ENCODING
 0     from_remote  joe
[admin@Our_GW] interface pptp-server>

The Remote router will be the pptp client:

[admin@Remote] interface pptp-client> add name=pptp user=joe \
\... connect-to=192.168.1.1 password=top_s3 mtu=1500 mru=1500
[admin@Remote] interface pptp-client> enable pptp
[admin@Remote] interface pptp-client> print
Flags: X - disabled, R - running
  0  R name="pptp" mtu=1500 mru=1500 connect-to=192.168.1.1 user="joe"
       password="top_s2" profile=default add-default-route=no

[admin@Remote] interface pptp-client> monitor pptp
      status: "connected"
      uptime: 39m46s
    encoding: "none"

[admin@Remote] interface pptp-client>
```

See the PPTP Interface Manual for more details on setting up encrypted channels.

2. Configure the EoIP tunnel by adding the eoip tunnel interfaces at both routers. Use the ip addresses of the pptp tunnel interfaces when specifying the argument values for the EoIP tunnel:

```
[admin@Our_GW] interface eoip> add name="eoip-remote" tunnel-id=0 \
\... remote-address=10.0.0.2
[admin@Our_GW] interface eoip> enable eoip-remote
[admin@Our_GW] interface eoip> print
Flags: X - disabled, R - running
  0    name=eoip-remote mtu=1500 arp=enabled remote-address=10.0.0.2 tunnel-id=0
[admin@Our_GW] interface eoip>

[admin@Remote] interface eoip> add name="eoip" tunnel-id=0 \
\... remote-address=10.0.0.1
[admin@Remote] interface eoip> enable eoip-main
[admin@Remote] interface eoip> print
Flags: X - disabled, R - running
  0   name=eoip mtu=1500 arp=enabled remote-address=10.0.0.1 tunnel-id=0

[Remote] interface eoip>
```

3. Enable bridging between the EoIP and Ethernet interfaces on both routers.
On the Our_GW:

```
[admin@Our_GW] interface bridge> add
[admin@Our_GW] interface bridge> print
Flags: X - disabled, R - running
 0  R name="bridge1" mtu=1500 arp=enabled mac-address=00:00:00:00:00:00
      protocol-mode=none priority=0x8000 auto-mac=yes
      admin-mac=00:00:00:00:00:00 max-message-age=20s forward-delay=15s
      transmit-hold-count=6 ageing-time=5m
[admin@Our_GW] interface bridge> port add bridge=bridge1 interface=eoip-remote
[admin@Our_GW] interface bridge> port add bridge=bridge1 interface=office-eth
[admin@Our_GW] interface bridge> port print
Flags: X - disabled, I - inactive, D - dynamic
 #    INTERFACE      BRIDGE  PRIORITY PATH-COST
 0    eoip-remote    bridge1 128      10
 1    office-eth     bridge1 128      10
[admin@Our_GW] interface bridge>
```

And the same for the Remote:

```
[admin@Remote] interface bridge> add
[admin@Remote] interface bridge> print
Flags: X - disabled, R - running
 0  R name="bridge1" mtu=1500 arp=enabled mac-address=00:00:00:00:00:00
      protocol-mode=none priority=0x8000 auto-mac=yes
      admin-mac=00:00:00:00:00:00 max-message-age=20s forward-delay=15s
      transmit-hold-count=6 ageing-time=5m
[admin@Remote] interface bridge> port add bridge=bridge1 interface=ether
[admin@Remote] interface bridge> port add bridge=bridge1 interface=eoip-main
[admin@Remote] interface bridge> port print
Flags: X - disabled, I - inactive, D - dynamic
 #    INTERFACE      BRIDGE  PRIORITY PATH-COST
 0    ether          bridge1 128      10
 1    eoip-main      bridge1 128      10
[admin@Remote] interface bridge>
```

4. Addresses from the same network can be used both in the Office LAN and in the Remote LAN.

Troubleshooting

Description

- **The routers can ping each other but EoIP tunnel does not seem to work!**
  Check the MAC addresses of the EoIP interfaces - they should not be the same!

IP Security

Document revision 3.6 (October 10, 2007, 12:17 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Specifications

Packages required: ***security***
License required: ***level1***
Home menu level: ***/ip ipsec***
Standards and Technologies: ***IPsec***
Hardware usage: *consumes a lot of CPU time (Intel Pentium MMX or AMD K6 suggested as a minimal configuration)*

Description

IPsec (IP Security) supports secure (encrypted, digitally signed) communications over IP networks.

Encryption

After packet is src-natted (if needed), but before putting it into interface queue, IPsec policy database is consulted to find out if packet should be encrypted. Security Policy Database (SPD) is a list of rules that have two parts:

- **Packet matching** - packet source/destination, protocol and ports (for TCP and UDP) are compared to values in policy rules, one after another
- **Action** - if rule matches action specified in rule is performed:

- - **none** - continue with the packet as if there was no IPsec
  - **discard** - drop the packet
  - **encrypt** - apply IPsec transformations to the packet

Each SPD rule can be associated with several Security Associations (SA) that determine packet encryption parameters (key, algorithm, SPI).

Note that packet can only be encrypted if there is a usable SA for policy rule. Same SA may be used for different policies, unless especially prohibited by a policy. By setting SPD rule security "level" user can control what happens when there is no valid SA for policy rule:

- **use** - if there is no valid SA, send packet unencrypted (like accept rule)
- **require** - drop packet, and ask IKE daemon to establish a new SA.
- **unique** - same as require, but establish a unique SA for this policy (i.e., this SA may not be shared with other policy)

Decryption

When encrypted packet is received for local host (after **dst-nat** and **input** filter), the appropriate SA is looked up to decrypt it (using packet source, destination, security protocol and SPI value). If no SA is found, the packet is dropped. If SA is found, packet is decrypted. Then decrypted packet's fields are compared to the policy rule that SA is linked to. If the packet does not match the policy rule, it is dropped. If the packet is decrypted fine (or authenticated fine) it is "received once more" - it goes through **dst-nat** and routing (which finds out what to do - either forward or deliver locally) again.

Note that before **forward** and **input** firewall chains, a packet that was not decrypted on local host is compared with SPD reversing its matching rules. If SPD requires encryption (there is valid SA associated with matching SPD rule), the packet is dropped. This is called incoming policy check.

Internet Key Exchange

The Internet Key Exchange (IKE) is a protocol that provides authenticated keying material for Internet Security Association and Key Management Protocol (ISAKMP) framework. There are other key exchange schemes that work with ISAKMP, but IKE is the most widely used one. Together they provide means for authentication of hosts and automatic management of security associations (SA).

Most of the time IKE daemon is doing nothing. There are two possible situations when it is activated:

- There is some traffic caught by a policy rule which needs to become encrypted or authenticated, but the policy doesn't have any SAs. The policy notifies IKE daemon about that, and IKE daemon initiates connection to remote host.
- IKE daemon responds to remote connection.

In both cases, peers establish connection and execute 2 phases:

- **Phase 1** - The peers agree upon algorithms they will use in the following IKE messages and authenticate. The keying material used to derive keys for all SAs and to protect following ISAKMP exchanges between hosts is generated also.
- **Phase 2** - The peers establish one or more SAs that will be used by IPsec to encrypt data. All SAs established by IKE daemon will have lifetime values (either limiting time, after which SA will become invalid, or amount of data that can be encrypted by this SA, or both).

There are two lifetime values - soft and hard. When SA reaches it's soft lifetime treshold, the IKE daemon receives a notice and starts another phase 2 exchange to replace this SA with fresh one. If SA reaches hard lifetime, it is discarded.

IKE can optionally provide a Perfect Forward Secrecy (PFS), which is a property of key exchanges, that, in turn, means for IKE that compromising the long term phase 1 key will not allow to easily gain access to all IPsec data that is protected by SAs established through this phase 1. It means an additional keying material is generated for each phase 2.

Generation of keying material is computationally very expensive. *Exempli gratia*, the use of modp8192 group can take several seconds even on very fast computer. It usually takes place once per phase 1 exchange, which happens only once between any host pair and then is kept for long time. PFS adds this expensive operation also to each phase 2 exchange.

Diffie-Hellman Groups

Diffie-Hellman (DH) key exchange protocol allows two parties without any initial shared secret to create one securely. The following Modular Exponential (MODP) and Elliptic Curve (EC2N) Diffie-Hellman (also known as "Oakley") Groups are supported:

Diffie-Hellman Group	Name	Reference
Group 1	768 bit MODP group	RFC2409
Group 2	1024 bits MODP group	RFC2409
Group 3	EC2N group on GP(2^155)	RFC2409
Group 4	EC2N group on GP(2^185)	RFC2409
Group 5	1536 bits MODP group	RFC3526

IKE Traffic

To avoid problems with IKE packets hit some SPD rule and require to encrypt it with not yet established SA (that this packet perhaps is trying to establish), locally originated packets with UDP source port 500 are not processed with SPD. The same way packets with UDP destination port 500 that are to be delivered locally are not processed in incoming policy check.

Setup Procedure

To get IPsec to work with automatic keying using IKE-ISAKMP you will have to configure **policy**, **peer** and **proposal** (optional) entries.

For manual keying you will have to configure **policy** and **manual-sa** entries.

Policy Settings

Home menu level: ***/ip ipsec policy***

Description

Policy table is needed to determine whether security settings should be applied to a packet.

Property Description

action (*none* | *discard* | *encrypt*; default: **accept**) - specifies what action to undertake with a packet that matches the policy

- **none** - pass the packet unchanged
- **discard** - drop the packet
- **encrypt** - apply transformations specified in this policy and it's SA

dont-fragment (*clear* | *inherit* | *set*; default: **clear**) - The state of the don't fragment IP header field

- **clear** - clear (unset) the field, so that packets previously marked as don't fragment can be fragmented. This setting is recommended as the packets are getting larger when IPsec protocol is applied to them, so large packets with don't fragment flag will not be able to pass the router
- **inherit** - do not change the field
- **set** - set the field, so that each packet matching the rule will not be fragmented. Not recommended

dst-address (*IP addressnetmaskport*; default: **0.0.0.0/32:any**) - destination IP address

dynamic (*read-only: flag*) - whether the rule has been created dynamically

in-accepted (*integer*) - how many incoming packets were passed through by the policy without an attempt to decrypt

in-dropped (*integer*) - how many incoming packets were dropped by the policy without an attempt to decrypt

in-transformed (*integer*) - how many incoming packets were decrypted (ESP) and/or verified (AH) by the policy

inactive (*read-only: flag*) - whether the rule is inactive (it may become inactive due to some misconfiguration)

ipsec-protocols (*multiple choice: ah* | *esp*; default: **esp**) - specifies what combination of Authentication Header and Encapsulating Security Payload protocols you want to apply to matched traffic. AH is applied after ESP, and in case of tunnel mode ESP will be applied in tunnel mode and AH - in transport mode

level (*unique* | *require* | *use*; default: **require**) - specifies what to do if some of the SAs for this policy cannot be found:

- **use** - skip this transform, do not drop packet and do not acquire SA from IKE daemon
- **require** - drop packet and acquire SA
- **unique** - drop packet and acquire a unique SA that is only used with this particular policy

manual-sa (*name*; default: **none**) - name of manual-sa template that will be used to create SAs for this policy

- **none** - no manual keys are set

out-accepted (*integer*) - how many outgoing packets were passed through by the policy without an attempt to encrypt

out-dropped (*integer*) - how many outgoing packets were dropped by the policy without an attempt to encrypt

out-transformed (*integer*) - how many outgoing packets were encrypted (ESP) and/or signed (AH)

ph2-state (*read-only: expired* | *no-phase2* | *established*) - indication of the progress of key establishing

- **expired** - there are some leftovers from previous phase2. In general it is similar to no-phase2
- **no-phase2** - no keys are estabilished at the moment
- **estabilished** - Appropriate SAs are in place and everything should be working fine

priority (*integer*; default: **0**) - policy ordering classificator (signed integer). Larger number means higher priority

proposal (*name*; default: **default**) - name of proposal information that will be sent by IKE daemon to establish SAs for this policy

protocol (*nameinteger*; default: **all**) - IP packet protocol to match

sa-dst-address (*IP address*; default: **0.0.0.0**) - SA destination IP address (remote peer)

sa-src-address (*IP address*; default: **0.0.0.0**) - SA source IP address (local peer)

src-address (*IP addressnetmaskport*; default: **0.0.0.0/32:any**) - source IP address

tunnel (yes | no; default: **no**) - specifies whether to use tunnel mode

Notes

All packets are IPIP encapsulated in tunnel mode, and their new IP header's **src-address** and **dst-address** are set to **sa-src-address** and **sa-dst-address** values of this policy. If you do not use tunnel mode (*id est* you use transport mode), then only packets whose source and destination addresses are the same as **sa-src-address** and **sa-dst-address** can be processed by this policy. Transport mode can only work with packets that originate at and are destined for IPsec peers (hosts that established security associations). To encrypt traffic between networks (or a network and a host) you have to use tunnel mode.

It is good to have **dont-fragment** cleared because encrypted packets are always bigger than original and thus they may need fragmentation.

If you are using IKE to establish SAs automatically, then policies on both routers must exactly match each other, *id est* **src-address=1.2.3.0/27** on one router and **dst-address=1.2.3.0/28** on another would not work. Source address values on one router MUST be equal to destination address values on the other one, and vice versa.

Example

To add a policy to encrypt all the traffic between two hosts (10.0.0.147 and 10.0.0.148), we need do the following:

```
[admin@MikroTik] ip ipsec policy> add sa-src-address=10.0.0.147 \
\... sa-dst-address=10.0.0.148 action=encrypt
[admin@MikroTik] ip ipsec policy> print
Flags: X - disabled, D - dynamic, I - inactive
 0    src-address=10.0.0.147/32:any dst-address=10.0.0.148/32:any protocol=all
     action=encrypt level=require ipsec-protocols=esp tunnel=no
     sa-src-address=10.0.0.147 sa-dst-address=10.0.0.148 proposal=default
     manual-sa=none priority=0

[admin@MikroTik] ip ipsec policy>
```

to view the policy statistics, do the following:

```
[admin@MikroTik] ip ipsec policy> print stats
Flags: X - disabled, D - dynamic, I - inactive
  0   src-address=10.0.0.147/32:any dst-address=10.0.0.148/32:any
      protocol=all ph2-state=no-phase2 in-accepted=0 in-dropped=0
      out-accepted=0 out-dropped=0 encrypted=0 not-encrypted=0 decrypted=0
      not-decrypted=0

[admin@MikroTik] ip ipsec policy>
```

Peers

Home menu level: ***/ip ipsec peer***

Description

Peer configuration settings are used to establish connections between IKE daemons (phase 1 configuration). This connection then will be used to negotiate keys and algorithms for SAs.

Property Description

address (*IP addressnetmaskport*; default: **0.0.0.0/32:500**) - address prefix. If remote peer's address matches this prefix, then this peer configuration is used while authenticating and establishing phase 1. If several peer's addresses matches several configuration entries, the most specific one (i.e. the one with largest netmask) will be used

auth-method (*pre-shared-key* | *rsa-signature*; default: **pre-shared-key**) - authentication method

- **pre-shared-key** - authenticate by a password (secret) string shared between the peers
- **rsa-signature** - authenticate using a pair of RSA certificates

certificate (*name*) - name of a certificate on the local side (signing packets; the certificate must have private key). Only needed if RSA signature authentication method is used

dh-group (*multiple choice: ec2n155* | *ec2n185* | *modp768* | *modp1024* | *modp1536*; default: **modp1024**) - Diffie-Hellman group (cipher strength)

enc-algorithm (*multiple choice: des* | *3des* | *aes-128* | *aes-192* | *aes-256*; default: **3des**) - encryption algorithm. Algorithms are named in strength increasing order

exchange-mode (*multiple choice: main | aggressive | base*; default: **main**) - different ISAKMP phase 1 exchange modes according to RFC 2408. Do not use other modes then main unless you know what you are doing

generate-policy (yes | no; default: **no**) - allow this peer to establish SA for non-existing policies. Such policies are created dynamically for the lifetime of SA. This way it is possible, for example, to create IPsec secured L2TP tunnels, or any other setup where remote peer's IP address is not known at the configuration time

hash-algorithm (*multiple choice: md5 | sha1*; default: **md5**) - hashing algorithm. SHA (Secure Hash Algorithm) is stronger, but slower

lifebytes (*integer*; default: **0**) - phase 1 lifetime: specifies how much bytes can be transferred before SA is discarded

- **0** - SA expiration will not be due to byte count excess

lifetime (*time*; default: **1d**) - phase 1 lifetime: specifies how long the SA will be valid; SA will be discarded after this time

nat-traversal (yes | no; default: **no**) - use Linux NAT-T mechanism to solve IPsec incompatibility with NAT routers inbetween IPsec peers. This can only be used with ESP protocol (AH is not supported by design, as it signes the complete packet, including IP header, which is changed by NAT, rendering AH signature invalid). The method encapsulates IPsec ESP traffic into UDP streams in order to overcome some minor issues that made ESP incompatible with NAT

proposal-check (*multiple choice: claim | exact | obey | strict*; default: **strict**) - phase 2 lifetime check logic:

- **claim** - take shortest of proposed and configured lifetimes and notify initiator about it
- **exact** - require lifetimes to be the same
- **obey** - accept whatever is sent by an initiator
- **strict** - if proposed lifetime is longer than the default then reject proposal otherwise accept proposed lifetime

remote-certificate (*name*) - name of a certificate for authenticating the remote side (validating packets; no private key required). Only needed if RSA signature authentication method is used

secret (*text*; default: **""**) - secret string (in case pre-shared key authentication is used). If it starts with '0x', it is parsed as a hexadecimal value

send-initial-contact (yes | no; default: **yes**) - specifies whether to send initial IKE information or wait for remote side

Notes

AES (Advanced Encryption Standard) encryption algorithms are much faster than DES, so it is recommended to use this algorithm class whenever possible. But, AES's speed is also its drawback as it potentially can be cracked faster, so use AES-256 when you need security or AES-128 when speed is also important.

Both peers MUST have the same encryption and authentication algorithms, DH group and exchange mode. Some legacy hardware may support only DES and MD5.

You should set **generate-policy** flag to **yes** only for trusted peers, because there is no verification done for the established policy. To protect yourself against possible unwanted events, add policies with **action=none** for all networks you don't want to be encrypted at the top of policy list. Since dynamic policies are added at the bottom of the list, they will not be able to override your configuration. Alternatively you can use policy priorities to enforce some policies to be active always.

Example

To define new peer configuration for **10.0.0.147** peer with **secret=gwejimezyfopmekun**:

```
[admin@MikroTik] ip ipsec peer>add address=10.0.0.147/32 \
\... secret=gwejimezyfopmekun
[admin@MikroTik] ip ipsec peer> print
Flags: X - disabled
  0   address=10.0.0.147/32:500 auth-method=pre-shared-key
      secret="gwejimezyfopmekun" generate-policy=no exchange-mode=main
      send-initial-contact=yes nat-traversal=no proposal-check=obey
      hash-algorithm=md5 enc-algorithm=3des dh-group=modp1024 lifetime=1d
      lifebytes=0

[admin@MikroTik] ip ipsec peer>
```

Remote Peer Statistics

Home menu level: ***/ip ipsec remote-peers***

Description

This submenu provides you with various statistics about remote peers that currently have established phase 1 connections with this router. Note that if peer doesn't show up here, it doesn't mean that no IPsec traffic is being exchanged with it. For example, manually configured SAs will not show up here.

Property Description

local-address (*read-only: IP address*) - local ISAKMP SA address

remote-address (*read-only: IP address*) - peer's IP address

side (*multiple choice, read-only: initiator | responder*) - shows which side initiated the connection

- **initiator** - phase 1 negotiation was started by this router
- **responder** - phase 1 negotiation was started by peer

state (*read-only: text*) - state of phase 1 negotiation with the peer

- **estabilished** - normal working state

Example

To see currently estabilished SAs:

```
[admin@MikroTik] ip ipsec> remote-peers print
  0 local-address=10.0.0.148 remote-address=10.0.0.147 state=established
    side=initiator
[admin@MikroTik] ip ipsec>
```

Installed SAs

Home menu level: ***/ip ipsec installed-sa***

Description

This facility provides information about installed security associations including the keys

Property Description

add-lifetime (*read-only: time*) - soft/hard expiration time counted from installation of SA

addtime (*read-only: text*) - time when this SA was installed

auth-algorithm (*multiple choice, read-only: none | md5 | sha1*) - authentication algorithm used in SA

auth-key (*read-only: text*) - authentication key presented as a hex string

current-bytes (*read-only: integer*) - amount of data processed by this SA's crypto algorithms

dst-address (*read-only: IP address*) - destination address of SA taken from respective policy

enc-algorithm (*multiple choice, read-only: none | des | 3des | aes*) - encryption algorithm used in SA

enc-key (*read-only: text*) - encryption key presented as a hex string (not applicable to AH SAs)

lifebytes (*read-only: integer*) - soft/hard expiration threshold for amount of processed data

replay (*read-only: integer*) - size of replay window presented in bytes. This window protects the receiver against replay attacks by rejecting old or duplicate packets

spi (*read-only: integer*) - SPI value of SA, represented in hexadecimal form

src-address (*read-only: IP address*) - source address of SA taken from respective policy

state (*multiple choice, read-only: larval | mature | dying | dead*) - SA living phase

use-lifetime (*read-only: time*) - soft/hard expiration time counted from the first use of SA

usetime (*read-only: text*) - time when this SA was first used

Example

Sample printout looks as follows:

```
[admin@MikroTik] ip ipsec> installed-sa print
Flags: A - AH, E - ESP, P - pfs
  0 E   spi=E727605 src-address=10.0.0.148 dst-address=10.0.0.147
        auth-algorithm=sha1 enc-algorithm=3des replay=4 state=mature
        auth-key="ecc5f4aee1b297739ec88e324d7cfb8594aa6c35"
        enc-key="d6943b8ea582582e449bde085c9471ab0b209783c9eb4bbd"
        addtime=jan/28/2003 20:55:12 add-lifetime=24m/30m
        usetime=jan/28/2003 20:55:23 use-lifetime=0s/0s current-bytes=128
        lifebytes=0/0

  1 E   spi=E15CEE06 src-address=10.0.0.147 dst-address=10.0.0.148
        auth-algorithm=sha1 enc-algorithm=3des replay=4 state=mature
        auth-key="8ac9dc7ecebfed9cd1030ae3b07b32e8e5cb98af"
        enc-key="8a8073a7afd0f74518c10438a0023e64cc660ed69845ca3c"
        addtime=jan/28/2003 20:55:12 add-lifetime=24m/30m
        usetime=jan/28/2003 20:55:12 use-lifetime=0s/0s current-bytes=512
        lifebytes=0/0
[admin@MikroTik] ip ipsec>
```

Flushing Installed SA Table

Command name: ***/ip ipsec installed-sa flush***

Description

Sometimes after incorrect/incomplete negotiations took place, it is required to flush manually the installed SA table so that SA could be renegotiated. This option is provided by the **flush** command.

Property Description

sa-type (*multiple choice: ah* | *all* | *esp*; default: **all**) - specifies SA types to flush

- **ah** - delete AH protocol SAs only
- **esp** - delete ESP protocol SAs only
- **all** - delete both ESP and AH protocols SAs

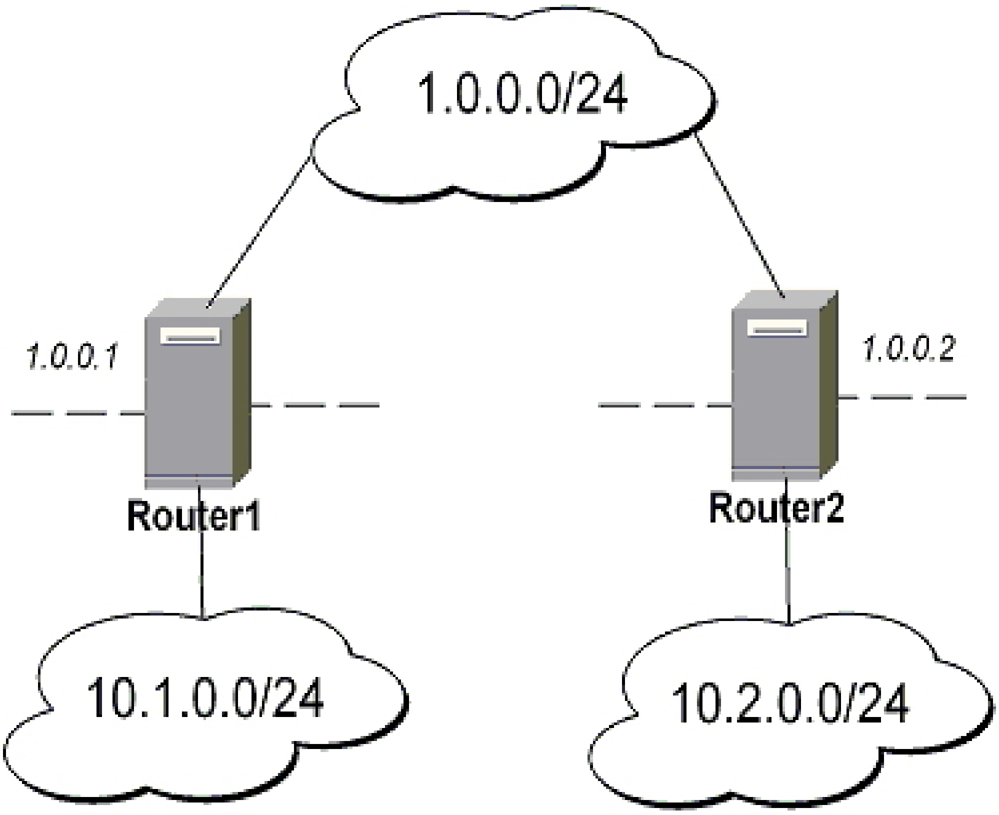
Example

To flush all the SAs installed:

```
[admin@MikroTik] ip ipsec installed-sa> flush
[admin@MikroTik] ip ipsec installed-sa> print
[admin@MikroTik] ip ipsec installed-sa>
```

Application Examples

MikroTik Router to MikroTik Router



- transport mode example using ESP with automatic keying
  - for **Router1**

```
[admin@Router1] > ip ipsec policy add sa-src-address=1.0.0.1 sa-dst-address=1.0.0.2 \
\... action=encrypt
[admin@Router1] > ip ipsec peer add address=1.0.0.2 \
\... secret="gvejimezyfopmekun"
```

- for **Router2**

```
[admin@Router2] > ip ipsec policy add sa-src-address=1.0.0.2 sa-dst-address=1.0.0.1 \
\... action=encrypt
[admin@Router2] > ip ipsec peer add address=1.0.0.1 \
\... secret="gvejimezyfopmekun"
```

- transport mode example using ESP with automatic keying and automatic policy generating on Router 1 and static policy on Router 2
  - for **Router1**

```
[admin@Router1] > ip ipsec peer add address=1.0.0.0/24 \
\... secret="gvejimezyfopmekun" generate-policy=yes
```

  - for **Router2**

```
[admin@Router2] > ip ipsec policy add sa-src-address=1.0.0.2 sa-dst-address=1.0.0.1 \
\... action=encrypt
[admin@Router2] > ip ipsec peer add address=1.0.0.1 \
\... secret="gvejimezyfopmekun"
```

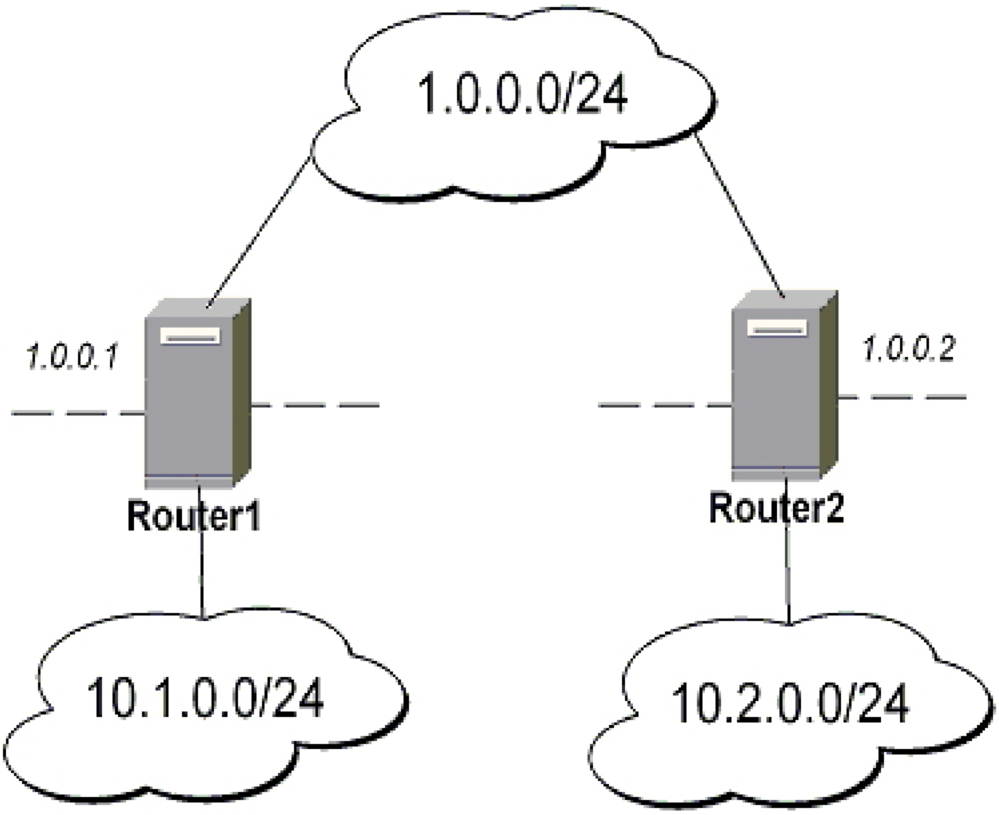
- tunnel mode example using AH with manual keying
  - for **Router1**

```
[admin@Router1] > ip ipsec manual-sa add name=ah-sa1 \
\... ah-spi=0x101/0x100 ah-key=abcfed
[admin@Router1] > ip ipsec policy add src-address=10.1.0.0/24 \
\... dst-address=10.2.0.0/24 action=encrypt ipsec-protocols=ah \
\... tunnel=yes sa-src=1.0.0.1 sa-dst=1.0.0.2 manual-sa=ah-sa1
```

  - for **Router2**

```
[admin@Router2] > ip ipsec manual-sa add name=ah-sa1 \
\... ah-spi=0x100/0x101 ah-key=abcfed
[admin@Router2] > ip ipsec policy add src-address=10.2.0.0/24 \
\... dst-address=10.1.0.0/24 action=encrypt ipsec-protocols=ah \
\... tunnel=yes sa-src=1.0.0.2 sa-dst=1.0.0.1 manual-sa=ah-sa1
```

IPsec Between two Masquerading MikroTik Routers



1. Add accept and masquerading rules in SRC-NAT
   - for **Router1**

```
[admin@Router1] > ip firewall nat add chain=srcnat src-address=10.1.0.0/24 \
\... dst-address=10.2.0.0/24 action=accept
[admin@Router1] > ip firewall nat add chain=srcnat out-interface=public \
\... action=masquerade
```

   - for **Router2**

```
[admin@Router2] > ip firewall nat chain=srcnat add src-address=10.2.0.0/24 \
\... dst-address=10.1.0.0/24 action=accept
[admin@Router2] > ip firewall nat chain=srcnat add out-interface=public \
\... action=masquerade
```

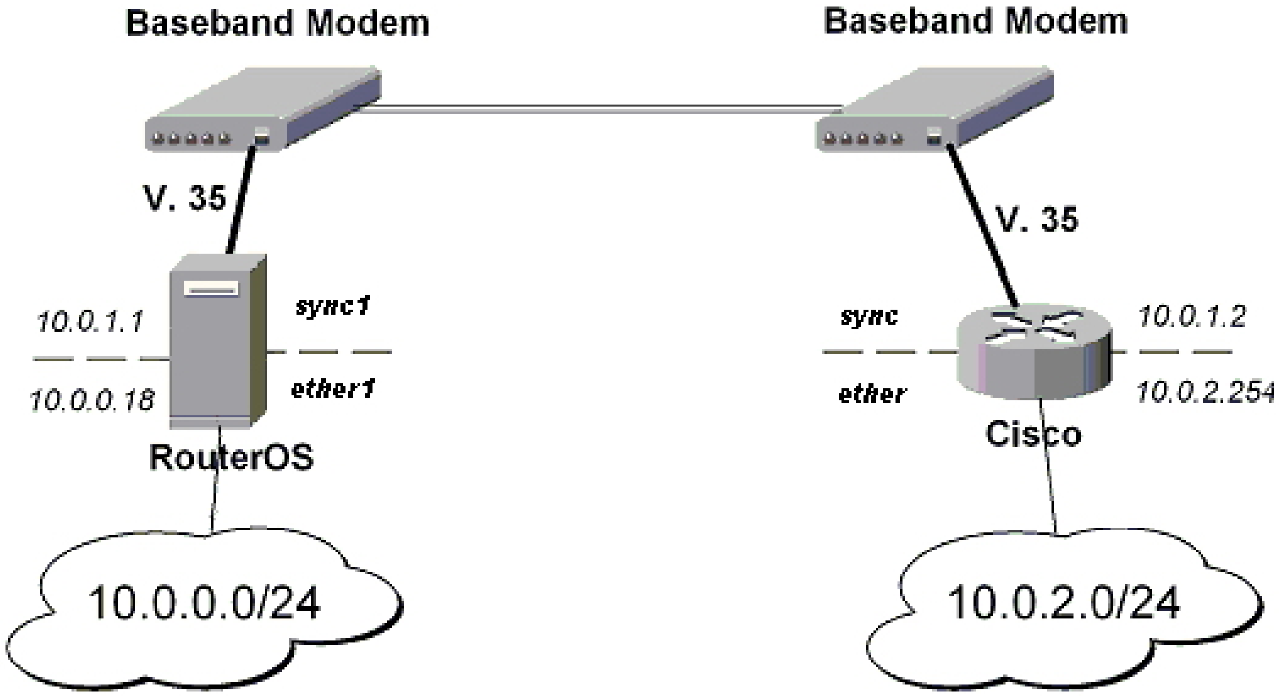
2. configure IPsec
   - for **Router1**

```
[admin@Router1] > ip ipsec policy add src-address=10.1.0.0/24 \
\... dst-address=10.2.0.0/24 action=encrypt tunnel=yes \
\... sa-src-address=1.0.0.1 sa-dst-address=1.0.0.2
[admin@Router1] > ip ipsec peer add address=1.0.0.2 \
\... exchange-mode=aggressive secret="gvejimezyfopmekun"
```

   - for **Router2**

```
[admin@Router2] > ip ipsec policy add src-address=10.2.0.0/24 \
\... dst-address=10.1.0.0/24 action=encrypt tunnel=yes \
\... sa-src-address=1.0.0.2 sa-dst-address=1.0.0.1
[admin@Router2] > ip ipsec peer add address=1.0.0.1 \
\... exchange-mode=aggressive secret="gvejimezyfopmekun"
```

MikroTik router to CISCO Router



We will configure IPsec in tunnel mode in order to protect traffic between attached subnets.

1. Add peer (with phase1 configuration parameters), DES and SHA1 will be used to protect IKE traffic
   - for **MikroTik** router

```
[admin@MikroTik] > ip ipsec peer add address=10.0.1.2 \
\... secret="gvejimezyfopmekun" enc-algorithm=des
```

   - for **CISCO** router

```
! Configure ISAKMP policy (phase1 config, must match configuration
! of "/ip ipsec peer" on RouterOS). Note that DES is default
! encryption algorithm on Cisco. SHA1 is default authentication
! algorithm
crypto isakmp policy 9
  encryption des
  authentication pre-share
  group 2
  hash md5
  exit

! Add preshared key to be used when talking to RouterOS
crypto isakmp key gvejimezyfopmekun address 10.0.1.1 255.255.255.255
```

2. Set encryption proposal (phase2 proposal - settings that will be used to encrypt actual data) to use DES to encrypt data
   - for **MikroTik** router

```
[admin@MikroTik] > ip ipsec proposal set default enc-algorithms=des
```

   - for **CISCO** router

```
! Create IPsec transform set - transformations that should be applied to
! traffic - ESP encryption with DES and ESP authentication with SHA1
! This must match "/ip ipsec proposal"
crypto ipsec transform-set myset esp-des esp-sha-hmac
  mode tunnel
  exit
```

3. Add policy rule that matches traffic between subnets and requires encryption with ESP in tunnel mode

   - for **MikroTik** router

```
[admin@MikroTik] > ip ipsec policy add \
\... src-address=10.0.0.0/24 dst-address=10.0.2.0/24 action=encrypt \
\... tunnel=yes sa-src=10.0.1.1 sa-dst=10.0.1.2
```

   - for **CISCO** router

```
! Create access list that matches traffic that should be encrypted
access-list 101 permit ip 10.0.2.0 0.0.0.255 10.0.0.0 0.0.0.255
! Create crypto map that will use transform set "myset", use peer 10.0.1.1
! to establish SAs and encapsulate traffic and use access-list 101 to
! match traffic that should be encrypted
crypto map mymap 10 ipsec-isakmp
  set peer 10.0.1.1
  set transform-set myset
  set pfs group2
  match address 101
  exit
! And finally apply crypto map to serial interface:
interface Serial 0
  crypto map mymap
  exit
```

4. Testing the IPsec tunnel

   - on **MikroTik** router we can see installed SAs

```
[admin@MikroTik] ip ipsec installed-sa> print
Flags: A - AH, E - ESP, P - pfs
  0 E   spi=9437482 src-address=10.0.1.1 dst-address=10.0.1.2
        auth-algorithm=sha1 enc-algorithm=des replay=4 state=mature
        auth-key="9cf2123b8b5add950e3e67b9eac79421d406aa09"
        enc-key="ffe7ec65b7a385c3" addtime=jul/12/2002 16:13:21
        add-lifetime=24m/30m usetime=jul/12/2002 16:13:21 use-lifetime=0s/0s
        current-bytes=71896 lifebytes=0/0
  1 E   spi=319317260 src-address=10.0.1.2 dst-address=10.0.1.1
        auth-algorithm=sha1 enc-algorithm=des replay=4 state=mature
        auth-key="7575f5624914dd312839694db2622a318030bc3b"
        enc-key="633593f809c9d6af" addtime=jul/12/2002 16:13:21
        add-lifetime=24m/30m usetime=jul/12/2002 16:13:21 use-lifetime=0s/0s
        current-bytes=0 lifebytes=0/0
[admin@MikroTik] ip ipsec installed-sa>
```

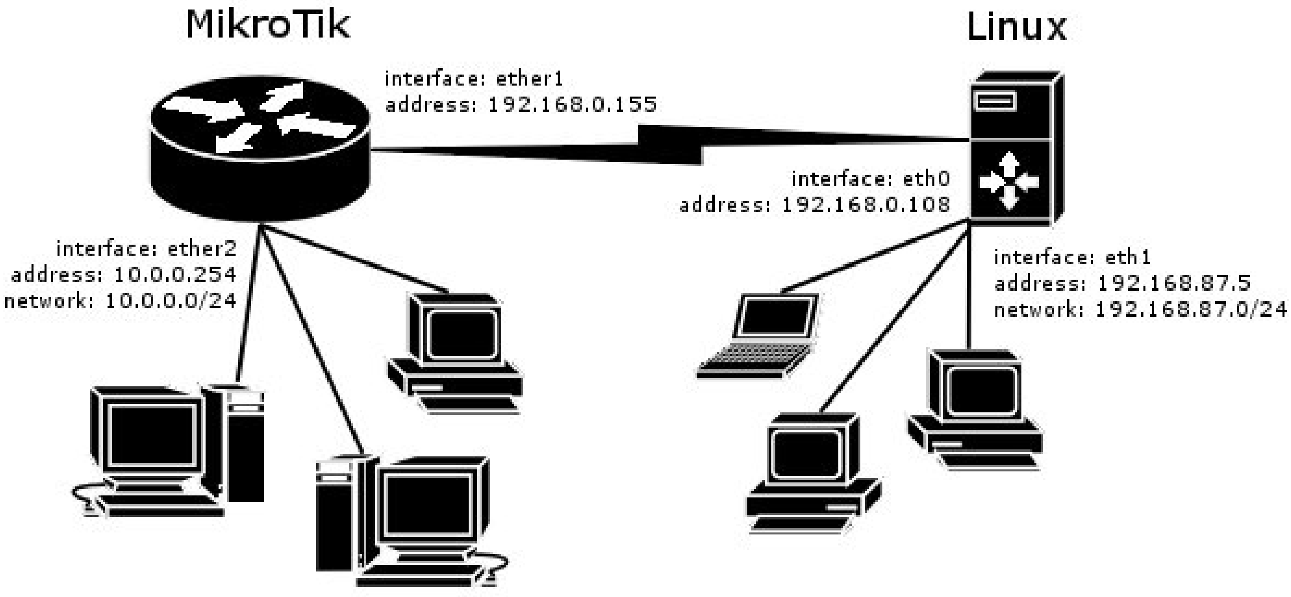
   - on **CISCO** router

```
cisco# show interface Serial 0
interface: Serial1
    Crypto map tag: mymap, local addr. 10.0.1.2
   local  ident (addr/mask/prot/port): (10.0.2.0/255.255.255.0/0/0)
   remote ident (addr/mask/prot/port): (10.0.0.0/255.255.255.0/0/0)
   current_peer: 10.0.1.1
     PERMIT, flags={origin_is_acl,}
    #pkts encaps: 1810, #pkts encrypt: 1810, #pkts digest 1810
    #pkts decaps: 1861, #pkts decrypt: 1861, #pkts verify 1861
    #pkts compressed: 0, #pkts decompressed: 0
    #pkts not compressed: 0, #pkts compr. failed: 0, #pkts decompress failed: 0
    #send errors 0, #recv errors 0
     local crypto endpt.: 10.0.1.2, remote crypto endpt.: 10.0.1.1
     path mtu 1500, media mtu 1500
```

```
      current outbound spi: 1308650C
      inbound esp sas:
       spi: 0x90012A(9437482)
         transform: esp-des esp-sha-hmac ,
         in use settings ={Tunnel, }
         slot: 0, conn id: 2000, flow_id: 1, crypto map: mymap
         sa timing: remaining key lifetime (k/sec): (4607891/1034)
         IV size: 8 bytes
         replay detection support: Y
      inbound ah sas:
      inbound pcp sas:
      outbound esp sas:
       spi: 0x1308650C(319317260)
         transform: esp-des esp-sha-hmac ,
         in use settings ={Tunnel, }
         slot: 0, conn id: 2001, flow_id: 2, crypto map: mymap
         sa timing: remaining key lifetime (k/sec): (4607893/1034)
         IV size: 8 bytes
         replay detection support: Y
      outbound ah sas:
      outbound pcp sas:
```

MikroTik Router and Linux FreeS/WAN

In the test scenario we have 2 private networks: 10.0.0.0/24 connected to the MT and 192.168.87.0/24 connected to Linux. MT and Linux are connected together over the "public" network 192.168.0.0/24:



- FreeS/WAN configuration:

```
 config setup
     interfaces="ipsec0=eth0"
     klipsdebug=none
     plutodebug=all
     plutoload=%search
     plutostart=%search
     uniqueids=yes

 conn %default
     keyingtries=0
     disablearrivalcheck=no
     authby=rsasig

 conn mt
```

```
     left=192.168.0.108
     leftsubnet=192.168.87.0/24
     right=192.168.0.155
     rightsubnet=10.0.0.0/24
     authby=secret
     pfs=no
     auto=add
```

- **ipsec.secrets** config file:

```
192.168.0.108 192.168.0.155 : PSK "gvejimezyfopmekun"
```

- MikroTik Router configuration:

```
[admin@MikroTik] > /ip ipsec peer add address=192.168.0.108 \
\... secret="gvejimezyfopmekun" hash-algorithm=md5  enc-algorithm=3des \
\... dh-group=modp1024 lifetime=28800s

[admin@MikroTik] > /ip ipsec proposal auth-algorithms=md5 \
\... enc-algorithms=3des pfs-group=none

[admin@MikroTik] > /ip ipsec policy add sa-src-address=192.168.0.155 \
\... sa-dst-address=192.168.0.108 src-address=10.0.0.0/24 \
\... dst-address=192.168.87.0/24 tunnel=yes
```

IPIP Tunnel Interfaces

Document revision 1.3 (October 10, 2007, 14:06 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The IPIP tunneling implementation on the MikroTik RouterOS is RFC 2003 compliant. IPIP tunnel is a simple protocol that encapsulates IP packets in IP to make a tunnel between two routers. The IPIP tunnel interface appears as an interface under the interface list. Many routers, including Cisco and Linux based, support this protocol. This protocol makes multiple network schemes possible.

IP tunneling protocol adds the following possibilities to a network setups:

- to tunnel Intranets over the Internet
- to use it instead of source routing

Quick Setup Guide

To make an IPIP tunnel between 2 MikroTik routers with IP addresses **10.5.8.104** and **10.1.0.172**, using IPIP tunnel addresses 10.0.0.1 and 10.0.0.2, follow the next steps.

- Configuration on router with IP address **10.5.8.104**:
    1. Add an IPIP interface (by default, its name will be **ipip1**):

```
[admin@MikroTik] interface ipip> add local-address=10.5.8.104 \
remote-address=10.1.0.172 disabled=no
```

    2. Add an IP address to created **ipip1** interface:

```
[admin@MikroTik] ip address> add address=10.0.0.1/24 interface=ipip1
```

- Configuration on router with IP address **10.1.0.172**:

1. Add an IPIP interface (by default, its name will be **ipip1**):

```
[admin@MikroTik] interface ipip> add local-address=10.1.0.172 \
remote-address=10.5.8.104 disabled=no
```

2. Add an IP address to created **ipip1** interface:

```
[admin@MikroTik] ip address> add address=10.0.0.2/24 interface=ipip1
```

Specifications

Packages required: ***system***
License required: ***level1 (limited to 1 tunnel), level3 (200 tunnels), level5 (unlimited)***
Home menu level: ***/interface ipip***
Standards and Technologies: ***IPIP (RFC 2003)***
Hardware usage: ***Not significant***

Additional Documents

- RFC1853
- RFC2003
- RFC1241

IPIP Setup

Home menu level: ***/interface ipip***

Description

An IPIP interface should be configured on two routers that have the possibility for an IP level connection and are RFC 2003 compliant. The IPIP tunnel may run over any connection that transports IP. Each IPIP tunnel interface can connect with one remote router that has a corresponding interface configured. An unlimited number of IPIP tunnels may be added to the router. There may only be one tunnel between a pair of IP addresses, so if you need various different tunnels between same hosts, use more than one IP address. For more details on IPIP tunnels, see RFC 2003.

Property Description

local-address (*IP address*) - local address on router which sends IPIP traffic to the remote host

mtu (*integer*; default: **1480**) - Maximum Transmission Unit. Should be set to 1480 bytes to avoid fragmentation of packets. May be set to 1500 bytes if mtu path discovery is not working properly on links

name (*name*; default: **ipipN**) - interface name for reference

remote-address (*IP address*) - the IP address of the remote host of the IPIP tunnel - may be any RFC 2003 compliant router

Notes

Use **/ip address add** command to assign an **IP address** to the IPIP interface.

There is no authentication or 'state' for this interface. The bandwidth usage of the interface may be monitored with the **monitor** feature from the **interface** menu.

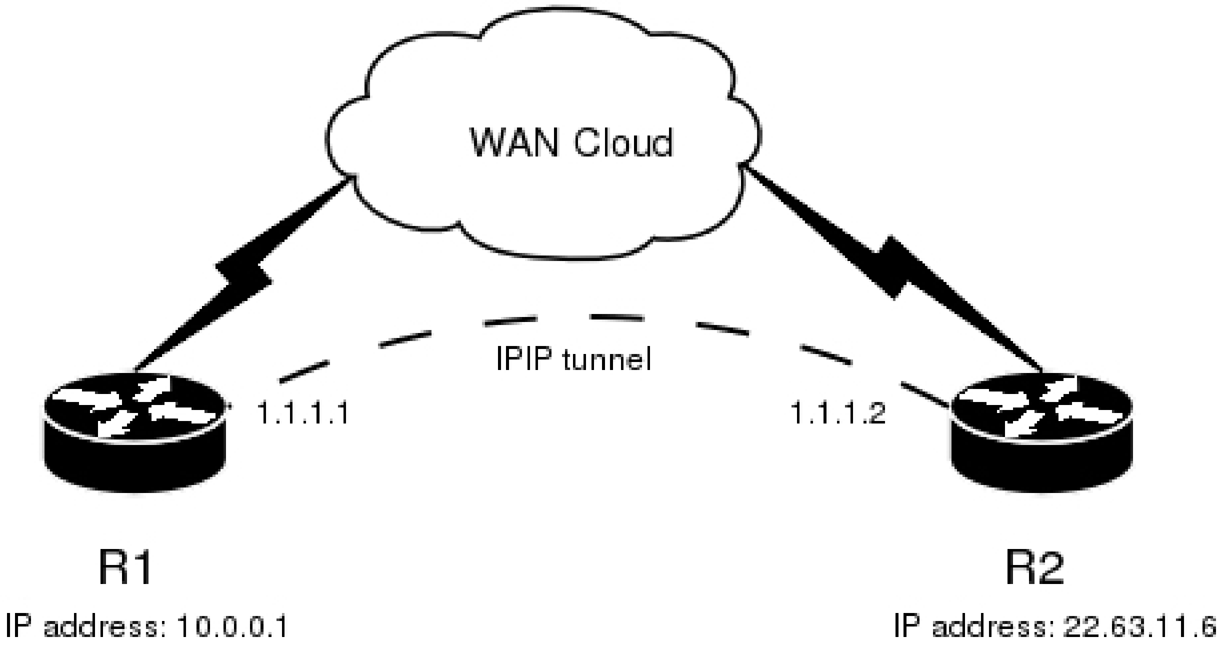
MikroTik RouterOS IPIP implementation has been tested with Cisco 1005. The sample of the Cisco 1005 configuration is given below:

```
interface Tunnel0
 ip address 10.3.0.1 255.255.255.0
 tunnel source 10.0.0.171
 tunnel destination 10.0.0.204
 tunnel mode ipip
```

Application Examples

Description

Suppose we want to add an IPIP tunnel between routers **R1** and **R2**:



At first, we need to configure IPIP interfaces and then add **IP addresses** to them.

The configuration for router **R1** is as follows:

```
[admin@MikroTik] interface ipip> add
local-address: 10.0.0.1
remote-address: 22.63.11.6
[admin@MikroTik] interface ipip> print
Flags: X - disabled, R - running, D - dynamic
  #    NAME                               MTU   LOCAL-ADDRESS   REMOTE-ADDRESS
  0 X  ipip1                              1480  10.0.0.1        22.63.11.6

[admin@MikroTik] interface ipip> enable 0
[admin@MikroTik] interface ipip> /ip address add address 1.1.1.1/24 interface=ipip1
```

The configuration of the **R2** is shown below:

```
[admin@MikroTik] interface ipip> add local-address=22.63.11.6 remote-address=10.0.0.1
[admin@MikroTik] interface ipip> print
Flags: X - disabled, R - running, D - dynamic
  #    NAME                           MTU   LOCAL-ADDRESS   REMOTE-ADDRESS
  0 X  ipip1                          1480  22.63.11.6      10.0.0.1

[admin@MikroTik] interface ipip> enable 0
[admin@MikroTik] interface ipip> /ip address add address 1.1.1.2/24 interface=ipip1
```

Now both routers can ping each other:

```
[admin@MikroTik] interface ipip> /ping 1.1.1.2
1.1.1.2 64 byte ping: ttl=64 time=24 ms
1.1.1.2 64 byte ping: ttl=64 time=19 ms
1.1.1.2 64 byte ping: ttl=64 time=20 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 19/21.0/24 ms
[admin@MikroTik] interface ipip>
```

L2TP Tunnel

Document revision 1.5 (January 16, 2008, 9:09 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

L2TP (Layer 2 Tunnel Protocol) supports encrypted tunnels over IP. The MikroTik RouterOS implementation includes support for both L2TP client and server.

General applications of L2TP tunnels include:

- secure router-to-router tunnels over the Internet
- linking (bridging) local Intranets or LANs

- extending PPP user connections to a remote location (for example, to separate authentication and Internet access points for ISP)
- accessing an Intranet/LAN of a company for remote (mobile) clients (employees)

Each L2TP connection is composed of a server and a client. The MikroTik RouterOS may function as a server or client or, for various configurations, it may be the server for some connections and client for other connections.

Quick Setup Guide

To make a L2TP tunnel between 2 MikroTik routers with IP addresses **10.5.8.104** (L2TP server) and **10.1.0.172** (L2TP client), follow the next steps.

- Configuration on L2TP server router:
  1. Add a L2TP user:

```
[admin@L2TP-Server] ppp secret> add name=user password=passwd \
\... local-address=10.0.0.1 remote-address=10.0.0.2
```

  2. Enable the L2TP server

```
[admin@L2TP-Server] interface l2tp-server server> set enabled=yes
```

- Configuration on L2TP client router:
  1. Add a L2TP client:

```
[admin@L2TP-Client] interface l2tp-client> add user=user password=passwd \
\... connect-to=10.5.8.104
```

Specifications

Packages required: ***ppp***
License required: ***level1 (limited to 1 tunnel), level3 (limited to 200 tunnels), level5***
Home menu level: ***/interface l2tp-server, /interface l2tp-client***
Standards and Technologies: ***L2TP (RFC 2661)***
Hardware usage: ***Not significant***

Description

L2TP is a secure tunnel protocol for transporting IP traffic using PPP. L2TP encapsulates PPP in virtual lines that run over IP, Frame Relay and other protocols (that are not currently supported by MikroTik RouterOS). L2TP incorporates PPP and MPPE (Microsoft Point to Point Encryption) to make encrypted links. The purpose of this protocol is to allow the Layer 2 and PPP endpoints to reside on different devices interconnected by a packet-switched network. With L2TP, a user has a Layer 2 connection to an access concentrator - **LAC** (e.g., modem bank, ADSL DSLAM, etc.), and the concentrator then tunnels individual PPP frames to the Network Access Server - **NAS**. This allows the actual processing of PPP packets to be separated from the termination of the Layer 2 circuit. From the user's perspective, there is no functional difference between having the L2 circuit terminate in a NAS directly or using L2TP.

It may also be useful to use L2TP just as any other tunneling protocol with or without encryption. The L2TP standard says that the most secure way to encrypt data is using L2TP over IPsec (**Note** that it is default mode for Microsoft L2TP client) as all L2TP control and data packets for a particular tunnel appear as homogeneous UDP/IP data packets to the IPsec system.

Multilink PPP (MP) is supported in order to provide MRRU (the ability to transmit full-sized 1500 and larger packets) and bridging over PPP links (using Bridge Control Protocol (BCP) that allows to send raw Ethernet frames over PPP links). This way it is possible to setup bridging without EoIP. The bridge should either have an administratively set MAC address or an Ethernet-like interface in it, as PPP links do not have MAC addresses.

L2TP includes PPP authentication and accounting for each L2TP connection. Full authentication and accounting of each connection may be done through a RADIUS client or locally.

MPPE 40bit RC4 and MPPE 128bit RC4 encryption are supported.

L2TP traffic uses UDP protocol for both control and data packets. UDP port 1701 is used only for link establishment, further traffic is using any available UDP port (which may or may not be 1701). This means that L2TP can be used with most firewalls and routers (even with NAT) by enabling UDP traffic to be routed through the firewall or router.

L2TP Client Setup

Home menu level: ***/interface l2tp-client***

Property Description

add-default-route (*yes* | *no*; default: **no**) - whether to use the server which this client is connected to as its default router (gateway)

allow (*multiple choice: mschap2, mschap1, chap, pap*; default: **mschap2, mschap1, chap, pap**) - the protocol to allow the client to use for authentication

connect-to (*IP address*) - The IP address of the L2TP server to connect to

max-mru (*integer*; default: **1460**) - Maximum Receive Unit. The optimal value is the MRU of the interface the tunnel is working over decreased by 40 (so, for 1500-byte Ethernet link, set the MRU to 1460 to avoid fragmentation of packets)

max-mtu (*integer*; default: **1460**) - Maximum Transmission Unit. The optimal value is the MTU of the interface the tunnel is working over decreased by 40 (so, for 1500-byte Ethernet link, set the MTU to 1460 to avoid fragmentation of packets)

mrru (*integer*: 512..65535; default: **disabled**) - maximum packet size that can be received on the link. If a packet is bigger than tunnel MTU, it will be split into multiple packets, allowing full size IP or Ethernet packets to be sent over the tunnel

- **disabled** - disable MRRU on this link

name (*name*; default: **l2tp-outN**) - interface name for reference

password (*text*; default: **""**) - user password to use when logging to the remote server

profile (*name*; default: **default**) - profile to use when connecting to the remote server

user (*text*) - user name to use when logging on to the remote server

Notes

Specifying MRRU means enabling MP (Multilink PPP) over single link. This protocol is used to split big packets into smaller ones. Under Windows it can be enabled in Networking tag, Settings button, "Negotiate multi-link for single link connections". Their MRRU is hardcoded to 1614. This setting is usefull to overcome PathMTU discovery failures. The MP should be enabled on both peers.

Example

To set up L2TP client named **test2** using username **john** with password **john** to connect to the **10.1.1.12** L2TP server and use it as the default gateway:

```
[admin@MikroTik] interface l2tp-client> add name=test2 connect-to=10.1.1.12 \
\... user=john add-default-route=yes password=john
[admin@MikroTik] interface l2tp-client> print
Flags: X - disabled, R - running
  0 X  name="test2" max-mtu=1460 max-mru=1460 mrru=disabled connect-to=10.1.1.12
       user="john" password="john" profile=default add-default-route=yes
       allow=pap,chap,mschap1,mschap2
[admin@MikroTik] interface l2tp-client> enable 0
```

Monitoring L2TP Client

Command name: ***/interface l2tp-client monitor***

Property Description

encoding (*text*) - encryption and encoding (if asymmetric, separated with '/') being used in this connection

idle-time (*read-only: time*) - time since the last packet has been transmitted over this link

mru (*read-only: integer*) - effective MRU of the link

mtu (*read-only: integer*) - effective MTU of the link

status (*text*) - status of the client

- **dialing** - attempting to make a connection
- **verifying password...** - connection has been established to the server, password verification in progress
- **connected** - self-explanatory
- **terminated** - interface is not enabled or the other side will not establish a connection

uptime (*time*) - connection time displayed in days, hours, minutes and seconds

Example

Example of an established connection:

```
[admin@MikroTik] interface l2tp-client> monitor test2
     status: "connected"
     uptime: 6h44m9s
  idle-time: 6h44m9s
   encoding: "MPPE128 stateless"
        mtu: 1460
        mru: 1460
[admin@MikroTik] interface l2tp-client>
```

L2TP Server Setup

Home menu level: ***/interface l2tp-server server***

Description

The L2TP server creates a dynamic interface for each connected L2TP client. The L2TP connection count from clients depends on the license level you have. Level1 license allows 1 L2TP client, Level3 or Level4 licenses up to 200 clients, and Level5 or Level6 licenses do not have L2TP client limitations.

To create L2TP users, you should consult the PPP secret and PPP Profile manuals. It is also possible to use the MikroTik router as a RADIUS client to register the L2TP users, see the manual how to do it.

Property Description

authentication (*multiple choice: pap | chap | mschap1 | mschap2*; default: **mschap2**) - authentication algorithm

default-profile - default profile to use

enabled (*yes* | *no*; default: **no**) - defines whether L2TP server is enabled or not

keepalive-timeout (*time*; default: **30**) - defines the time period (in seconds) after which the router is starting to send keepalive packets every second. If no traffic and no keepalive responses has came for that period of time (i.e. 2 * keepalive-timeout), not responding client is proclaimed disconnected

max-mru (*integer*; default: **1460**) - Maximum Receive Unit. The optimal value is the MRU of the interface the tunnel is working over decreased by 40 (so, for 1500-byte ethernet link, set the MRU to 1460 to avoid fragmentation of packets)

max-mtu (*integer*; default: **1460**) - Maximum Transmission Unit. The optimal value is the MTU of the interface the tunnel is working over decreased by 40 (so, for 1500-byte ethernet link, set the MTU to 1460 to avoid fragmentation of packets)

mrru (*integer*: 512..65535; default: **disabled**) - maximum packet size that can be received on the link. If a packet is bigger than tunnel MTU, it will be split into multiple packets, allowing full size IP or Ethernet packets to be sent over the tunnel

- **disabled** - disable MRRU on this link

Notes

Specifying MRRU means enabling MP (Multilink PPP) over single link. This protocol is used to split big packets into smaller ones. Under Windows it can be enabled in Networking tag, Settings button, "Negotiate multi-link for single link connections". Their MRRU is hardcoded to 1614. This setting is usefull to overcome PathMTU discovery failures. The MP should be enabled on both peers.

Example

To enable L2TP server:

```
[admin@MikroTik] interface l2tp-server server> set enabled=yes
[admin@MikroTik] interface l2tp-server server> print
             enabled: yes
             max-mtu: 1460
```

```
            max-mru: 1460
               mrru: disabled
     authentication: mschap2,mschap1
  keepalive-timeout: 30
    default-profile: default
[admin@MikroTik] interface l2tp-server server>
```

L2TP Tunnel Interfaces

Home menu level: ***/interface l2tp-server***

Description

There are two types of interface (tunnel) items in PPTP server configuration - static users and dynamic connections. An interface is created for each tunnel established to the given server. Static interfaces are added administratively if there is a need to reference the particular interface name (in firewall rules or elsewhere) created for the particular user. Dynamic interfaces are added to this list automatically whenever a user is connected and its username does not match any existing static entry (or in case the entry is active already, as there can not be two separate tunnel interfaces referenced by the same name). Dynamic interfaces appear when a user connects and disappear once the user disconnects, so it is impossible to reference the tunnel created for that use in router configuration (for example, in firewall), so if you need a persistent rules for that user, create a static entry for him/her. Otherwise it is safe to use dynamic configuration. **Note** that in both cases PPP users must be configured properly - static entries do not replace PPP configuration.

Property Description

client-address (*read-only: IP address*) - shows the IP address of the connected client

encoding (*read-only: text*) - encryption and encoding (if asymmetric, separated with '/') being used in this connection

mru (*read-only: integer*) - client's MRU

mtu (*read-only: integer*) - client's MTU

name (*name*) - interface name

uptime (*read-only: time*) - shows how long the client is connected

user (*name*) - the name of the user that is configured statically or added dynamically

Example

To add a static entry for **ex1** user:

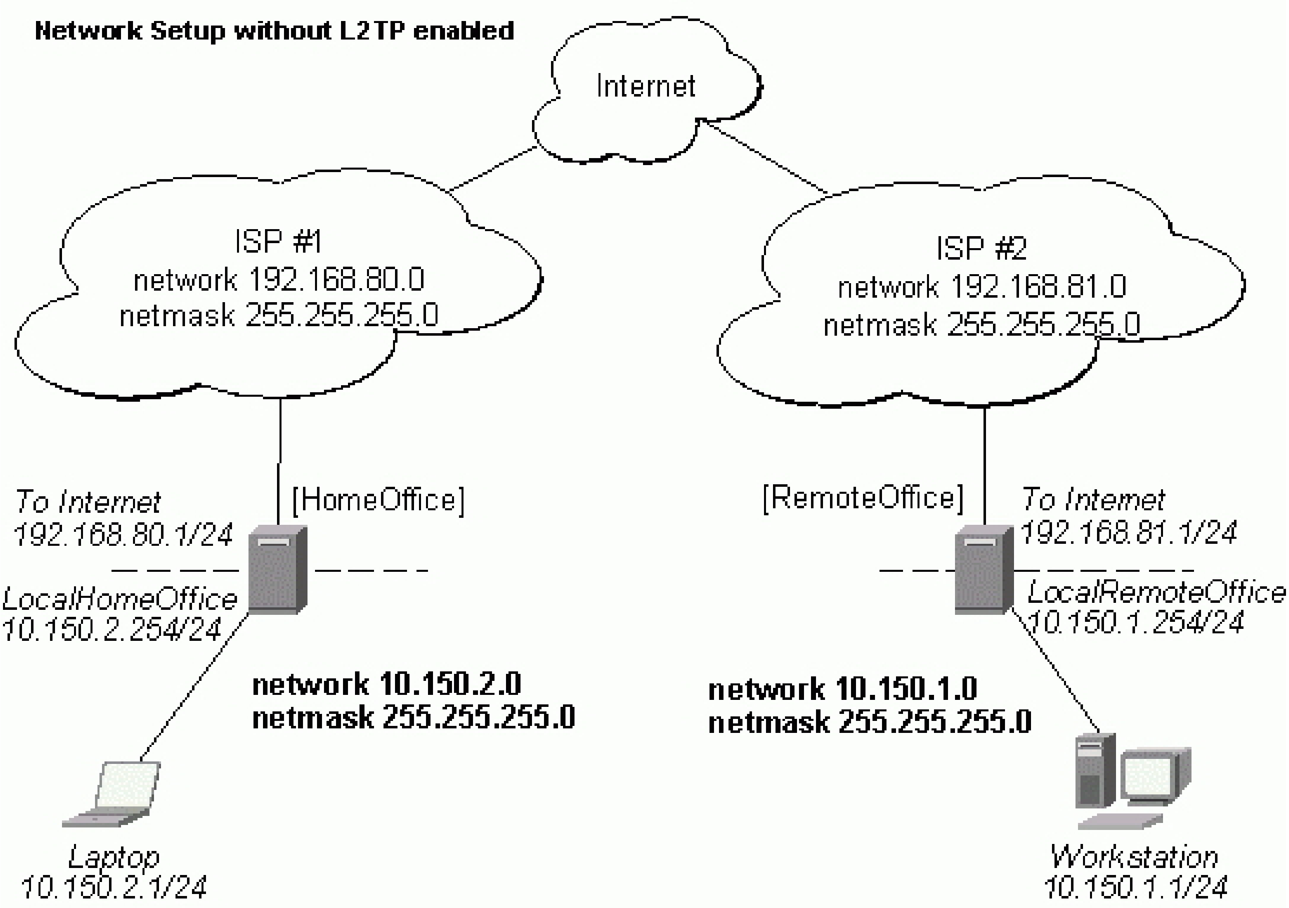
```
[admin@MikroTik] interface l2tp-server> add user=ex1
[admin@MikroTik] interface l2tp-server> print
Flags: X - disabled, D - dynamic, R - running
  #     NAME                 USER         MTU   CLIENT-ADDRESS  UPTIME   ENC...
  0  DR <l2tp-ex>            ex           1460  10.0.0.202      6m32s    none
  1     l2tp-in1             ex1
[admin@MikroTik] interface l2tp-server>
```

In this example an already connected user **ex** is shown besides the one we just added. Now the interface named **l2tp-in1** can be referenced from anywhere in RouterOS configuration like a regular interface.

L2TP Application Examples

Router-to-Router Secure Tunnel Example

The following is an example of connecting two Intranets using an encrypted L2TP tunnel over the Internet.



There are two routers in this example:

- [HomeOffice]
  Interface LocalHomeOffice 10.150.2.254/24
  Interface ToInternet 192.168.80.1/24
- [RemoteOffice]
  Interface ToInternet 192.168.81.1/24
  Interface LocalRemoteOffice 10.150.1.254/24

Each router is connected to a different ISP. One router can access another router through the Internet.

On the L2TP server a user must be set up for the client:

```
[admin@HomeOffice] ppp secret> add name=ex service=l2tp password=lkjrht
local-address=10.0.103.1 remote-address=10.0.103.2
[admin@HomeOffice] ppp secret> print detail
Flags: X - disabled
  0   name="ex" service=l2tp caller-id="" password="lkjrht" profile=default
      local-address=10.0.103.1 remote-address=10.0.103.2 routes==""
```

```
[admin@HomeOffice] ppp secret>
```

Then the user should be added in the L2TP server list:

```
[admin@HomeOffice] interface l2tp-server> add user=ex
[admin@HomeOffice] interface l2tp-server> print
Flags: X - disabled, D - dynamic, R - running
  #     NAME                 USER         MTU   CLIENT-ADDRESS  UPTIME   ENC...
  0     l2tp-in1             ex
[admin@HomeOffice] interface l2tp-server>
```

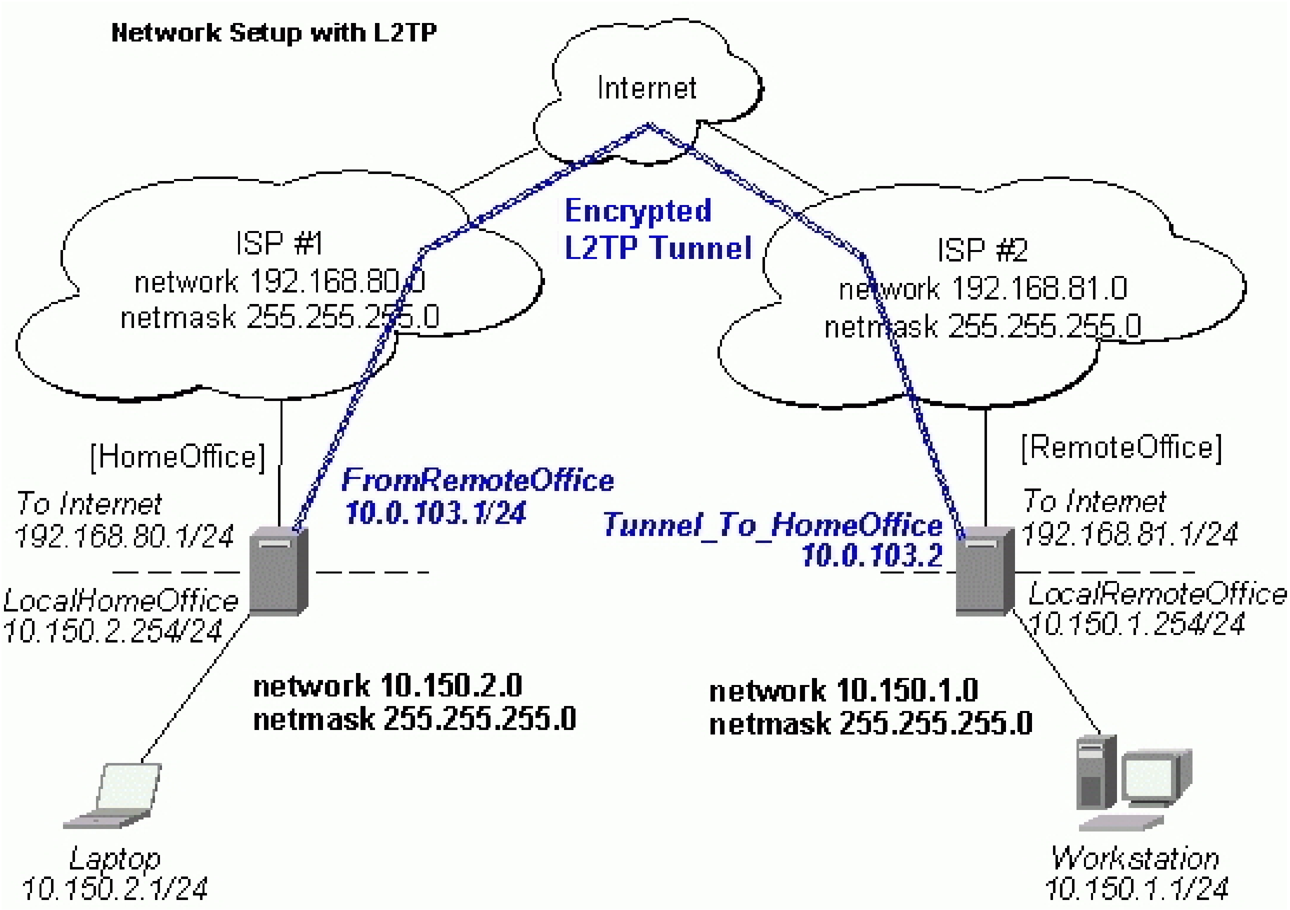
And finally, the server must be enabled:

```
[admin@HomeOffice] interface l2tp-server server> set enabled=yes
[admin@HomeOffice] interface l2tp-server server> print
            enabled: yes
            max-mtu: 1460
            max-mru: 1460
               mrru: disabled
      authentication: mschap2
  keepalive-timeout: 30
    default-profile: default
[admin@HomeOffice] interface l2tp-server server>
```

Add a L2TP client to the RemoteOffice router:

```
[admin@RemoteOffice] interface l2tp-client> add connect-to=192.168.80.1 user=ex \
\... password=lkjrht disabled=no
[admin@RemoteOffice] interface l2tp-client> print
Flags: X - disabled, R - running
  0  R name="l2tp-out1" mtu=1460 mru=1460 mrru=disabled connect-to=192.168.80.1
       user="ex" password="lkjrht" profile=default add-default-route=no
       allow=pap,chap,mschap1,mschap2
[admin@RemoteOffice] interface l2tp-client>
```

Thus, a L2TP tunnel is created between the routers. This tunnel is like an Ethernet point-to-point connection between the routers with IP addresses 10.0.103.1 and 10.0.103.2 at each router. It enables 'direct' communication between the routers over third party networks.



To route the local Intranets over the L2TP tunnel you need to add these routes:

```
[admin@HomeOffice] > ip route add dst-address 10.150.1.0/24 gateway 10.0.103.2
[admin@RemoteOffice] > ip route add dst-address 10.150.2.0/24 gateway 10.0.103.1
```

On the L2TP server it can alternatively be done using **routes** parameter of the user configuration:

```
[admin@HomeOffice] ppp secret> print detail
Flags: X - disabled
  0   name="ex" service=l2tp caller-id="" password="lkjrht" profile=default
      local-address=10.0.103.1 remote-address=10.0.103.2 routes==""

[admin@HomeOffice] ppp secret> set 0 routes="10.150.1.0/24 10.0.103.2 1"
[admin@HomeOffice] ppp secret> print detail
Flags: X - disabled
  0   name="ex" service=l2tp caller-id="" password="lkjrht" profile=default
      local-address=10.0.103.1 remote-address=10.0.103.2
      routes="10.150.1.0/24 10.0.103.2 1"

[admin@HomeOffice] ppp secret>
```

Test the L2TP tunnel connection:

```
[admin@RemoteOffice]> /ping 10.0.103.1
10.0.103.1 pong: ttl=255 time=3 ms
10.0.103.1 pong: ttl=255 time=3 ms
10.0.103.1 pong: ttl=255 time=3 ms
ping interrupted
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 3/3.0/3 ms
```

Test the connection through the L2TP tunnel to the LocalHomeOffice interface:

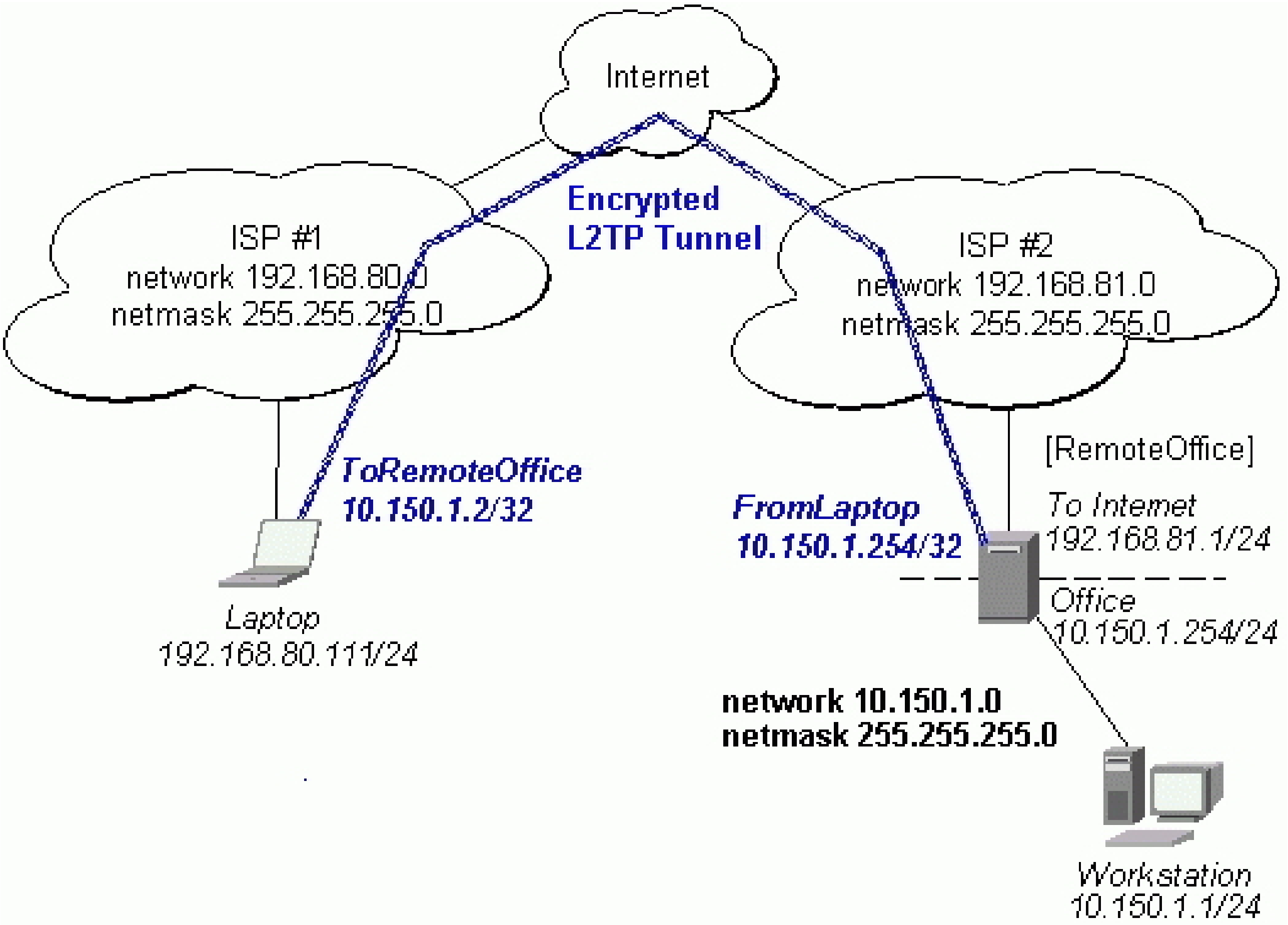
```
[admin@RemoteOffice]> /ping 10.150.2.254
10.150.2.254 pong: ttl=255 time=3 ms
10.150.2.254 pong: ttl=255 time=3 ms
10.150.2.254 pong: ttl=255 time=3 ms
ping interrupted
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 3/3.0/3 ms
```

To bridge a LAN over this secure tunnel, please see the example in the 'EoIP' section of the manual. To set the maximum speed for traffic over this tunnel, please consult the 'Queues' section.

Connecting a Remote Client via L2TP Tunnel

The following example shows how to connect a computer to a remote office network over L2TP encrypted tunnel giving that computer an IP address from the same network as the remote office has (without need of bridging over EoIP tunnels).

Please, consult the respective manual on how to set up a L2TP client with the software you are using.



The router in this example:

- [RemoteOffice]
  Interface ToInternet 192.168.81.1/24
  Interface Office 10.150.1.254/24

The client computer can access the router through the Internet.

On the L2TP server a user must be set up for the client:

```
[admin@RemoteOffice] ppp secret> add name=ex service=l2tp password=lkjrht
local-address=10.150.1.254 remote-address=10.150.1.2
[admin@RemoteOffice] ppp secret> print detail
Flags: X - disabled
  0   name="ex" service=l2tp caller-id="" password="lkjrht" profile=default
      local-address=10.150.1.254 remote-address=10.150.1.2 routes==""

[admin@RemoteOffice] ppp secret>
```

Then the user should be added in the L2TP server list:

```
[admin@RemoteOffice] interface l2tp-server> add name=FromLaptop user=ex
[admin@RemoteOffice] interface l2tp-server> print
Flags: X - disabled, D - dynamic, R - running
  #     NAME                 USER         MTU   CLIENT-ADDRESS  UPTIME   ENC...
  0     FromLaptop           ex
[admin@RemoteOffice] interface l2tp-server>
```

And the server must be enabled:

```
[admin@RemoteOffice] interface l2tp-server server> set enabled=yes
[admin@RemoteOffice] interface l2tp-server server> print
            enabled: yes
            max-mtu: 1460
            max-mru: 1460
               mrru: disabled
     authentication: mschap2
  keepalive-timeout: 30
    default-profile: default
[admin@RemoteOffice] interface l2tp-server server>
```

Finally, the proxy APR must be enabled on the 'Office' interface:

```
[admin@RemoteOffice] interface ethernet> set Office arp=proxy-arp
[admin@RemoteOffice] interface ethernet> print
Flags: X - disabled, R - running
  #    NAME                 MTU   MAC-ADDRESS       ARP
  0  R ToInternet           1500  00:30:4F:0B:7B:C1 enabled
  1  R Office               1500  00:30:4F:06:62:12 proxy-arp
[admin@RemoteOffice] interface ethernet>
```

L2TP Setup for Windows

Microsoft provides L2TP client support for Windows XP, 2000, NT4, ME and 98. Windows 2000 and XP include support in the Windows setup or automatically install L2TP. For 98, NT and ME, installation requires a download from Microsoft (L2TP/IPsec VPN Client).

For more information, see:

Microsoft L2TP/IPsec VPN Client Microsoft L2TP/IPsec VPN Client

On Windows 2000, L2TP setup without IPsec requires editing registry:

Disabling IPsec for the Windows 2000 Client

Disabling IPSEC Policy Used with L2TP

Troubleshooting

Description

- **I use firewall and I cannot establish L2TP connection**
  Make sure UDP connections can pass through both directions between your sites.
- **My Windows L2TP/IPsec VPN Client fails to connect to L2TP server with "Error 789" or "Error 781"**
  The error messages 789 and 781 occur when IPsec is not configured properly on both ends. See the respective documentation on how to configure IPsec in the Microsoft L2TP/IPsec VPN Client and in the MikroTik RouterOS. If you do not want to use IPsec, it can be easily switched off on the client side. Note: if you are using Windows 2000, you need to edit system registry using regedt32.exe or regedit.exe. Add the following registry value to **HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Rasman\Parameters**:

```
Value Name: ProhibitIpSec
Data Type: REG_DWORD
Value: 1
```

You must restart the Windows 2000 for the changes to take effect

For more information on configuring Windows 2000, see:

- Configuring Cisco IOS and Windows 2000 Clients for L2TP Using Microsoft IAS
- Disabling IPSEC Policy Used with L2TP
- How to Configure a L2TP/IPsec Connection Using Pre-shared Key Authentication

PPPoE

Document revision 1.7 (January 16, 2008, 9:13 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The PPPoE (Point to Point Protocol over Ethernet) protocol provides extensive user management, network management and accounting benefits to ISPs and network administrators. Currently PPPoE is used mainly by ISPs to control client connections for xDSL and cable modems as well as plain Ethernet networks. PPPoE is an extension of the standard Point to Point Protocol (PPP). The difference between them is expressed in transport method: PPPoE employs Ethernet instead of serial modem connection.

Generally speaking, PPPoE is used to hand out IP addresses to clients based on the username (and workstation, if desired) authentication as opposed to workstation only authentication, when static IP addresses or DHCP are used. It is adviced not to use static IP addresses or DHCP on the same interfaces as PPPoE for obvious security reasons.

MikroTik RouterOS can act as a RADIUS client - you can use a RADIUS server to authenticate PPPoE clients and use accounting for them.

A PPPoE connection is composed of a client and an access concentrator (server). The client may be any computer that has the PPPoE client protocol support installed. MikroTik RouterOS supports both - client and access concentrator sides of PPPoE. The PPPoE client and server work over any Ethernet level interface on the router - wireless 802.11 (Aironet, Cisco, WaveLan, Prism, Atheros), 10/100/1000 Mbit/s Ethernet, RadioLan and EoIP (Ethernet over IP tunnel). MPPE 40bit RSA and MPPE 128bit RSA encryption is supported.

Multilink PPP (MP) is supported in order to provide MRRU (the ability to transmit full-sized 1500 and larger packets) and bridging over PPP links (using Bridge Control Protocol (BCP) that allows to send raw Ethernet frames over PPP links). This way it is possible to setup bridging without EoIP. The bridge should either have an administratively set MAC address or an Ethernet-like interface in it, as PPP links do not have MAC addresses.

Note that when RADIUS server is authenticating a user with CHAP, MS-CHAPv1 or MS-CHAPv2, the RADIUS protocol does not use shared secret, it is used only in authentication reply. So if you have a wrong shared secret, RADIUS server will accept the request. You can use **/radius monitor** command to see **bad-replies** parameter. This value should increase whenever a client tries to connect.

Supported connections

- MikroTik RouterOS PPPoE client to any PPPoE server (access concentrator)
- MikroTik RouterOS server (access concentrator) to multiple PPPoE clients (clients are avaliable for almost all operating systems and most routers)

Quick Setup Guide

- To configure MikroTik RouterOS to be a PPPoE client
  1. Just add a pppoe-client:

```
/interface pppoe-client add name=pppoe-user-mike user=user password=passwd \
\... interface=wlan1 service-name=internet disabled=no
```

- To configure MikroTik RouterOS to be an Access Concentrator (PPPoE Server)
  1. Add an address pool for the clients from **10.1.1.62** to **10.1.1.72**, called pppoe-pool:

```
/ip pool add name="pppoe-pool" ranges=10.1.1.62-10.1.1.72
```

  2. Add PPP profile, called **pppoe-profile** where **local-address** will be the router's address and clients will have an address from **pppoe-pool**:

```
/ppp profile add name="pppoe-profile" local-address=10.1.1.1 remote-address=pppoe-pool
```

  3. Add a user with username **mike** and password **123**:

```
/ppp secret add name=user password=passwd service=pppoe profile=pppoe-profile
```

  4. Now add a pppoe server:

```
/interface pppoe-server server add service-name=internet interface=wlan1 \
\... default-profile=pppoe-profile
```

Specifications

Packages required: ***ppp***
License required: ***level1 (limited to 1 interface), level3 (limited to 200 interfaces), level4 (limited to 200 interfaces), level5 (limited to 500 interfaces), level6 (unlimited)***
Home menu level: ***/interface pppoe-server, /interface pppoe-client***
Standards and Technologies: ***PPPoE (RFC 2516)***
Hardware usage: *PPPoE server may require additional RAM (uses approx. 9KiB (plus extra 10KiB for packet queue, if data rate limitation is used) for each connection) and CPU power. Maximum of 65535 connections is supported.*

Additional Documents

Links for PPPoE documentation:

- http://www.faqs.org/rfcs/rfc2516.html

PPPoE Clients:

- RASPPPoE for Windows 95, 98, 98SE, ME, NT4, 2000, XP, .NET
  http://www.raspppoe.com/

PPPoE Client Setup

Home menu level: ***/interface pppoe-client***

Property Description

ac-name (*text*; default: **""**) - this may be left blank and the client will connect to any access concentrator that offers the "service" name selected

add-default-route (*yes* | *no*; default: **no**) - whether to add a default route automatically

allow (*multiple choice: mschap2*, *mschap1*, *chap*, *pap*; default: **mschap2, mschap1, chap, pap**) - the protocol to allow the client to use for authentication

dial-on-demand (*yes* | *no*; default: **no**) - connects to AC only when outbound traffic is generated and disconnects when there is no traffic for the period set in the idle-timeout value

interface (*name*) - interface the PPPoE server can be reached through

max-mru (*integer*; default: **1460**) - Maximum Receive Unit. The optimal value is the MRU of the interface the tunnel is working over decreased by 40 (so, for 1500-byte Ethernet link, set the MRU to 1460 to avoid fragmentation of packets)

max-mtu (*integer*; default: **1460**) - Maximum Transmission Unit. The optimal value is the MTU of the interface the tunnel is working over decreased by 40 (so, for 1500-byte Ethernet link, set the MTU to 1460 to avoid fragmentation of packets)

mrru (*integer*: 512..65535; default: **disabled**) - maximum packet size that can be received on the link. If a packet is bigger than tunnel MTU, it will be split into multiple packets, allowing full size IP or Ethernet packets to be sent over the tunnel

- **disabled** - disable MRRU on this link

name (*name*; default: **pppoe-out1**) - name of the PPPoE interface

password (*text*; default: "") - a user password used to connect the PPPoE server

profile (*name*) - default profile for the connection

service-name (*text*; default: "") - specifies the service name set on the access concentrator. Leave it blank unless you have many services and need to specify the one you need to connect to

use-peer-dns (*yes* | *no*; default: **no**) - whether to set the router's default DNS to the PPP peer DNS (i.e. whether to get DNS settings from the peer)

user (*text*; default: "") - a user name that is present on the PPPoE server

Notes

Note for Windows. Some connection instructions may use the form where the "phone number", such as "MikroTik_AC\mt1", is specified to indicate that "MikroTik_AC" is the access concentrator name and "mt1" is the service name.

Specifying MRRU means enabling MP (Multilink PPP) over single link. This protocol is used to split big packets into smaller ones. Under Windows it can be enabled in Networking tag, Settings button, "Negotiate multi-link for single link connections". Their MRRU is hardcoded to 1614. This setting is usefull to overcome PathMTU discovery failures. The MP should be enabled on both peers.

Example

To add and enable PPPoE client on the **ether1** interface connecting to the AC that provides **testSN** service using user name **user** with the password **passwd**:

```
[admin@RemoteOffice] interface pppoe-client> add interface=ether1 \
\... service-name=testSN user=user password=passwd disabled=no
[admin@RemoteOffice] interface pppoe-client> print
Flags: X - disabled, R - running
 0  R name="pppoe-out1" max-mtu=1480 max-mru=1480 mrru=disabled interface=ether1
      user="user" password="passwd" profile=default service-name="testSN"
      ac-name="" add-default-route=no dial-on-demand=no use-peer-dns=no
      allow=pap,chap,mschap1,mschap2
[admin@RemoteOffice] interface pppoe-client>
```

Monitoring PPPoE Client

Command name: ***/interface pppoe-client monitor***

Property Description

ac-mac (*MAC address*) - MAC address of the access concentrator (AC) the client is connected to

ac-name (*text*) - name of the AC the client is connected to

encoding (*text*) - encryption and encoding (if asymmetric, separated with '/') being used in this connection

mru (*read-only: integer*) - effective MRU of the link

mtu (*read-only: integer*) - effective MTU of the link

service-name (*text*) - name of the service the client is connected to

status (*text*) - status of the client

- **dialing** - attempting to make a connection
- **verifying password...** - connection has been established to the server, password verification in progress
- **connected** - self-explanatory
- **terminated** - interface is not enabled or the other side will not establish a connection

uptime (*time*) - connection time displayed in days, hours, minutes and seconds

Example

To monitor the **pppoe-out1** connection:

```
[admin@MikroTik] interface pppoe-client> monitor pppoe-out1
        status: "connected"
        uptime: 6s
     idle-time: 6s
      encoding: "MPPE128 stateless"
  service-name: "testSN"
       ac-name: "MikroTik"
        ac-mac: 00:0C:42:04:00:73
           mtu: 1480
           mru: 1480

[admin@MikroTik] interface pppoe-client>
```

PPPoE Server Setup (Access Concentrator)

Home menu level: ***/interface pppoe-server server***

Description

The PPPoE server (access concentrator) supports multiple servers for each interface - with differing service names. Currently the throughput of the PPPoE server has been tested to 160 Mb/s on a Celeron 600 CPU. Using higher speed CPUs, throughput should increase proportionately.

The **access concentrator name** and PPPoE **service name** are used by clients to identity the access concentrator to register with. The **access concentrator name** is the same as the **identity** of the router displayed before the command prompt. The identity may be set within the **/system identity** submenu.

Note that if no service name is specified in WindowsXP, it will use only service with no name. So if you want to serve WindowsXP clients, leave your service name empty.

Property Description

authentication (*multiple choice: mschap2 | mschap1 | chap | pap*; default: **mschap2, mschap1, chap, pap**) - authentication algorithm

default-profile (*name*; default: **default**) - default user profile to use

interface (*name*) - interface, which the clients are connected to

keepalive-timeout (*time*; default: **10**) - defines the time period (in seconds) after which the router is starting to send keepalive packets every second. If no traffic and no keepalive responses has came for that period of time (i.e. 2 * keepalive-timeout), not responding client is proclaimed disconnected.

max-mru (*integer*; default: **1480**) - Maximum Receive Unit. The optimal value is the MTU of the interface the tunnel is working over decreased by 20 (so, for 1500-byte Ethernet link, set the MTU to 1480 to avoid fragmentation of packets)

max-mtu (*integer*; default: **1480**) - Maximum Transmission Unit. The optimal value is the MTU of the interface the tunnel is working over decreased by 20 (so, for 1500-byte Ethernet link, set the MTU to 1480 to avoid fragmentation of packets)

max-sessions (*integer*; default: **0**) - maximum number of clients that the AC can serve

- **0** - unlimited

mrru (*integer*: 512..65535; default: **disabled**) - maximum packet size that can be received on the link. If a packet is bigger than tunnel MTU, it will be split into multiple packets, allowing full size IP or Ethernet packets to be sent over the tunnel

- **disabled** - disable MRRU on this link

one-session-per-host (*yes* | *no*; default: **no**) - allow only one session per host (determined by MAC address). If a host will try to establish a new session, the old one will be closed

service-name (*text*) - the PPPoE service name

Notes

The default **keepalive-timeout** value of **10** is OK in most cases. If you set it to **0**, the router will not disconnect clients until they explicitly log out or the router is restarted. To resolve this problem, the **one-session-per-host** property can be used.

Security issue: do not assign an IP address to the interface you will be receiving the PPPoE requests on.

Specifying MRRU means enabling MP (Multilink PPP) over single link. This protocol is used to split big packets into smaller ones. Under Windows it can be enabled in Networking tag, Settings button, "Negotiate multi-link for single link connections". Their MRRU is hardcoded to 1614. This setting is usefull to overcome PathMTU discovery failures. The MP should be enabled on both peers.

Example

To add PPPoE server on **ether1** interface providing **ex** service and allowing only one connection per host:

```
[admin@MikroTik] interface pppoe-server server> add interface=ether1 \
\... service-name=ex one-session-per-host=yes
[admin@MikroTik] interface pppoe-server server> print
Flags: X - disabled
  0 X service-name="ex" interface=ether1 mtu=1480 mru=1480 mrru=disabled
      authentication=mschap2,mschap,chap,pap keepalive-timeout=10
      one-session-per-host=yes max-sessions=0 default-profile=default
[admin@MikroTik] interface pppoe-server server>
```

PPPoE Tunnel Interfaces

Home menu level: ***/interface pppoe-server***

Description

There are two types of interface (tunnel) items in PPTP server configuration - static users and dynamic connections. An interface is created for each tunnel established to the given server. Static interfaces are added administratively if there is a need to reference the particular interface name (in firewall rules or elsewhere) created for the particular user. Dynamic interfaces are added to this list automatically whenever a user is connected and its username does not match any existing static entry (or in case the entry is active already, as there can not be two separate tunnel interfaces referenced by the same name). Dynamic interfaces appear when a user connects and disappear once the user disconnects, so it is impossible to reference the tunnel created for that use in router configuration (for example, in firewall), so if you need a persistent rules for that user, create a static entry for him/her. Otherwise it is safe to use dynamic configuration. **Note** that in both cases PPP users must be configured properly - static entries do not replace PPP configuration.

Property Description

encoding (*read-only: text*) - encryption and encoding (if asymmetric, separated with '/') being used in this connection

mru (*read-only: integer*) - client's MRU

mtu (*read-only: integer*) - client's MTU

name (*name*) - interface name

remote-address (*read-only: MAC address*) - MAC address of the connected client

service (*name*) - name of the service the user is connected to

uptime (*read-only: time*) - shows how long the client is connected

user (*name*) - the name of the connected user (must be present in the user darabase anyway)

Example

To view the currently connected users:

```
[admin@MikroTik] interface pppoe-server> print
Flags: X - disabled, D - dynamic, R - running
 #     NAME        USER      SERVICE   REMOTE... ENCODING  UPTIME
 0  DR <pppoe-ex> user      ex        00:0C:... MPPE12... 40m45s
[admin@MikroTik] interface pppoe-server>
```

To disconnect the user **ex**:

```
[admin@MikroTik] interface pppoe-server> remove [find user=ex]
[admin@MikroTik] interface pppoe-server> print

[admin@MikroTik] interface pppoe-server>
```

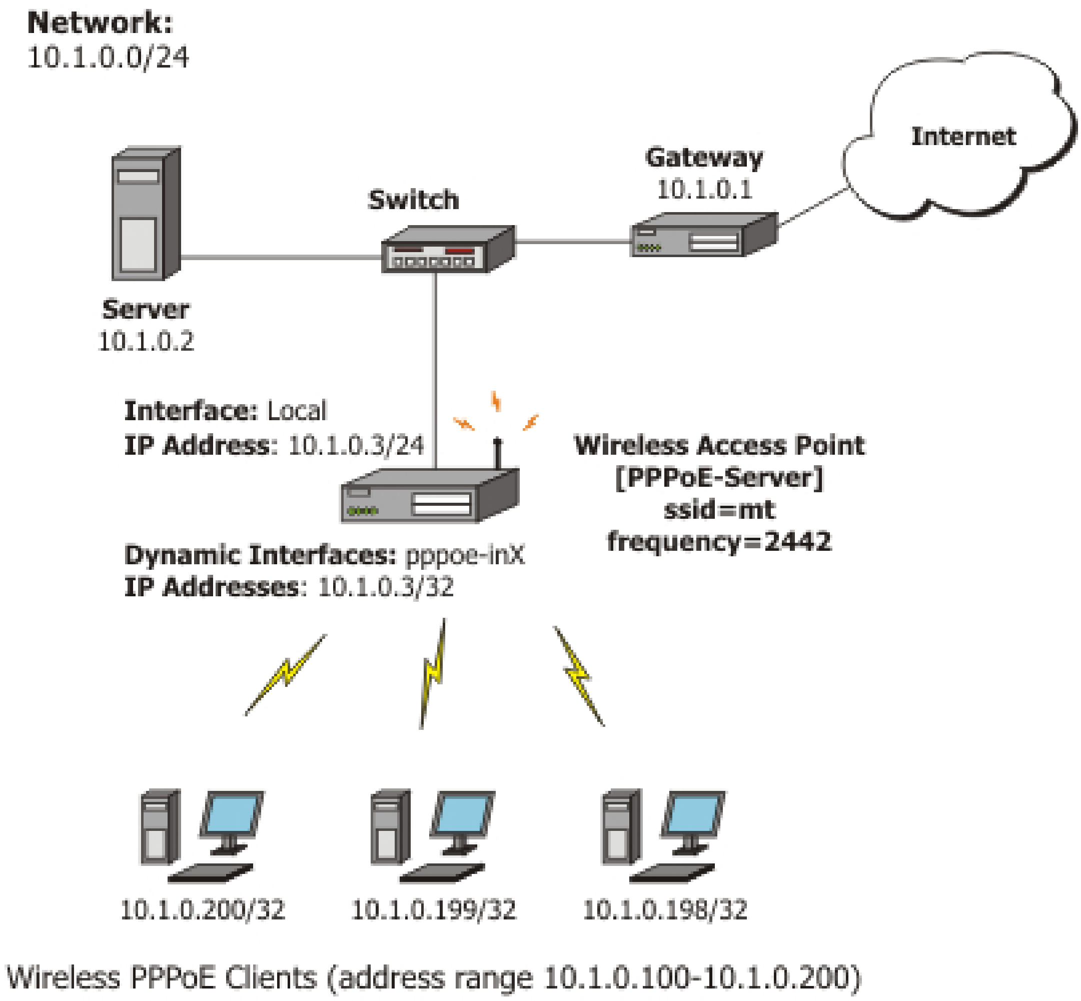
Application Examples

PPPoE in a multipoint wireless 802.11g network

In a wireless network, the PPPoE server may be attached to an Access Point (as well as to a regular station of wireless infrastructure). Either our RouterOS client or Windows PPPoE clients may connect to the Access Point for PPPoE authentication. Further, for RouterOS clients, the radio interface may be set to MTU 1600 so that the PPPoE interface may be set to MTU 1500. This optimizes the transmission of 1500 byte packets and avoids any problems associated with MTUs lower than 1500. It has not been determined how to change the

MTU of the Windows wireless interface at this moment.

Let us consider the following setup where the MikroTik Wireless AP offers wireless clients transparent access to the local network with authentication:



First of all, the wireless interface should be configured:

```
[admin@PPPoE-Server] interface wireless> set 0 mode=ap-bridge \
   frequency=2442 band=2.4ghz-b/g ssid=mt disabled=no
[admin@PPPoE-Server] interface wireless> print
Flags: X - disabled, R - running
 0 X  name="wlan1" mtu=1500 mac-address=00:0C:42:18:5C:3D arp=enabled
      interface-type=Atheros AR5413 mode=ap-bridge ssid="mt" frequency=2442
      band=2.4ghz-b/g scan-list=default antenna-mode=ant-a wds-mode=disabled
      wds-default-bridge=none wds-ignore-ssid=no default-authentication=yes
      default-forwarding=yes default-ap-tx-limit=0 default-client-tx-limit=0
      hide-ssid=no security-profile=default compression=no
[admin@PPPoE-Server] interface wireless>
```

Now, configure the Ethernet interface, add the IP address and set the default route:

```
[admin@PPPoE-Server] ip address> add address=10.1.0.3/24 interface=Local
[admin@PPPoE-Server] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   10.1.0.3/24        10.1.0.0        10.1.0.255      Local
[admin@PPPoE-Server] ip address> /ip route
```

```
[admin@PPPoE-Server] ip route> add gateway=10.1.0.1
[admin@PPPoE-Server] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DISTANCE INTER...
 0 ADC  10.1.0.0/24        10.1.0.3                          0        Local
 1 A S  0.0.0.0/0                          r 10.1.0.1        1        Local
[admin@PPPoE-Server] ip route> /interface ethernet
[admin@PPPoE-Server] interface ethernet> set Local arp=proxy-arp
[admin@PPPoE-Server] interface ethernet> print
Flags: X - disabled, R - running
 #    NAME                                 MTU   MAC-ADDRESS       ARP
 0  R Local                                1500  00:0C:42:03:25:53 proxy-arp
[admin@PPPoE-Server] interface ethernet>
```

We should add PPPoE server to the wireless interface:

```
[admin@PPPoE-Server] interface pppoe-server server> add interface=wlan1 \
   service-name=mt one-session-per-host=yes disabled=no
[admin@PPPoE-Server] interface pppoe-server server> print
Flags: X - disabled
 0   service-name="mt" interface=wlan1 max-mtu=1480 max-mru=1480 mrru=disabled
     authentication=pap,chap,mschap1,mschap2 keepalive-timeout=10
     one-session-per-host=yes max-sessions=0 default-profile=default
[admin@PPPoE-Server] interface pppoe-server server>
```

Finally, we can set up PPPoE clients:

```
[admin@PPPoE-Server] ip pool> add name=pppoe ranges=10.1.0.100-10.1.0.200
[admin@PPPoE-Server] ip pool> print
 # NAME                                         RANGES
 0 pppoe                                        10.1.0.100-10.1.0.200
[admin@PPPoE-Server] ip pool> /ppp profile
[admin@PPPoE-Server] ppp profile> set default use-encryption=yes \
   local-address=10.1.0.3 remote-address=pppoe
[admin@PPPoE-Server] ppp profile> print
Flags: * - default
 0 * name="default" local-address=10.1.0.3 remote-address=pppoe
     use-compression=no use-vj-compression=no use-encryption=yes only-one=no
     change-tcp-mss=yes

 1 * name="default-encryption" use-compression=default
     use-vj-compression=default use-encryption=yes only-one=default
     change-tcp-mss=default
[admin@PPPoE-Server] ppp profile> .. secret
[admin@PPPoE-Server] ppp secret> add name=w password=wkst service=pppoe
[admin@PPPoE-Server] ppp secret> add name=l password=ltp service=pppoe
[admin@PPPoE-Server] ppp secret> print
Flags: X - disabled
 #   NAME        SERVICE CALLER-ID PASSWORD  PROFILE            REMOTE-ADDRESS
 0   w           pppoe             wkst      default            0.0.0.0
 1   l           pppoe             ltp       default            0.0.0.0
[admin@PPPoE-Server] ppp secret>
```

Thus we have completed the configuration and added two users: **w** and **l** who are able to connect to Internet, using PPPoE client software.

Note that Windows XP built-in client supports encryption, but RASPPPOE does not. So, if it is planned not to support Windows clients older than Windows XP, it is recommended not to require encryption. In other case, the server will accept clients that do not encrypt data.

Troubleshooting

Description

- **I can connect to my PPPoE server. The ping goes even through it, but I still cannot open web pages**

  Make sure that you have specified a valid DNS server in the router (in **/ip dns** or in **/ppp profile** the **dns-server** parameter).

- **The PPPoE server shows more than one active user entry for one client, when the clients disconnect, they are still shown and active**

  Set the **keepalive-timeout** parameter (in the PPPoE server configuration) to **10** if You want clients to be considered logged off if they do not respond for 10 seconds.

  Note that if the **keepalive-timeout** parameter is set to **0** and the **only-one** parameter (in PPP profile settings) is set to **yes** then the clients might be able to connect only once. To resolve this problem **one-session-per-host** parameter in PPPoE server configuration should be set to **yes**

- **My Windows XP client cannot connect to the PPPoE server**

  You have to specify the "Service Name" in the properties of the XP PPPoE client. If the service name is not set, or it does not match the service name of the MikroTik PPPoE server, you get the "line is busy" errors, or the system shows "verifying password - unknown error"

- **I want to have logs for PPPoE connection establishment**

  Configure the logging feature under the **/system logging facility** and enable the PPP type logs

PPTP Tunnel

Document revision 1.7 (January 16, 2008, 9:10 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

PPTP (Point to Point Tunnel Protocol) supports encrypted tunnels over IP. The MikroTik RouterOS implementation includes support for both PPTP client and server.

General applications of PPTP tunnels:

- secure router-to-router tunnels over the Internet

- linking (bridging) local Intranets or LANs
- accessing an Intranet/LAN of a company for remote (mobile) clients (employees)

Each PPTP connection is composed of a server and a client. The MikroTik RouterOS may function as a server or client or, for various configurations, it may be the server for some connections and client for other connections. For example, the client created below could connect to a Windows 2000 server, another MikroTik Router, or another router which supports a PPTP server.

Quick Setup Guide

To make a PPTP tunnel between 2 MikroTik routers with IP addresses **10.5.8.104** (PPTP server) and **10.1.0.172** (PPTP client), follow the next steps.

- Configuration on PPTP server router:
  1. Add a user:

```
[admin@PPTP-Server] ppp secret> add name=user password=passwd \
\... local-address=10.0.0.1 remote-address=10.0.0.2
```

  2. Enable the PPTP server:

```
[admin@PPTP-Server] interface pptp-server server> set enabled=yes
```

- Configuration on PPTP client router:
  1. Add the PPTP client:

```
[admin@PPTP-Client] interface pptp-client> add user=user password=passwd \
\... connect-to=10.5.8.104 disabled=no
```

Specifications

Packages required: ***ppp***
License required: ***level1 (limited to 1 tunnel), level3 (limited to 200 tunnels), level5***
Home menu level: ***/interface pptp-server, /interface pptp-client***
Standards and Technologies: ***PPTP (RFC 2637)***
Hardware usage: ***Not significant***

Description

PPTP is a secure tunnel for transporting IP traffic using PPP. PPTP encapsulates PPP in virtual lines that run over IP. PPTP incorporates PPP and MPPE (Microsoft Point to Point Encryption) to make encrypted links. The purpose of this protocol is to make well-managed secure connections between routers as well as between routers and PPTP clients (clients are available for and/or included in almost all OSs including Windows).

Multilink PPP (MP) is supported in order to provide MRRU (the ability to transmit full-sized 1500 and larger packets) and bridging over PPP links (using Bridge Control Protocol (BCP) that allows to send raw Ethernet frames over PPP links). This way it is possible to setup bridging without EoIP. The bridge should either have an administratively set MAC address or an Ethernet-like interface in it, as PPP links do not have MAC addresses.

PPTP includes PPP authentication and accounting for each PPTP connection. Full authentication and accounting of each connection may be done through a RADIUS client or locally.

MPPE 40bit RC4 and MPPE 128bit RC4 encryption are supported.

PPTP traffic uses TCP port 1723 and IP protocol GRE (Generic Routing Encapsulation, IP protocol ID 47), as assigned by the Internet Assigned Numbers Authority (IANA). PPTP can be used with most firewalls and routers by enabling traffic destined for TCP port 1723 and protocol 47 traffic to be routed through the firewall or router.

PPTP connections may be limited or impossible to setup though a masqueraded/NAT IP connection. Please see the Microsoft and RFC links listed below for more information.

Additional Documents

- http://msdn.microsoft.com/library/backgrnd/html/understanding_pptp.htm
- http://support.microsoft.com/support/kb/articles/q162/8/47.asp
- http://www.ietf.org/rfc/rfc2637.txt?number=2637
- http://www.ietf.org/rfc/rfc3078.txt?number=3078
- http://www.ietf.org/rfc/rfc3079.txt?number=3079

PPTP Client Setup

Home menu level: ***/interface pptp-client***

Property Description

add-default-route (*yes* | *no*; default: **no**) - whether to use the server which this client is connected to as its default router (gateway)

allow (*multiple choice: mschap2*, *mschap1*, *chap*, *pap*; default: **mschap2, mschap1, chap, pap**) - the protocol to allow the client to use for authentication

connect-to (*IP address*) - The IP address of the PPTP server to connect to

max-mru (*integer*; default: **1460**) - Maximum Receive Unit. The optimal value is the MRU of the interface the tunnel is working over decreased by 40 (so, for 1500-byte Ethernet link, set the MRU to 1460 to avoid fragmentation of packets)

max-mtu (*integer*; default: **1460**) - Maximum Transmission Unit. The optimal value is the MTU of the interface the tunnel is working over decreased by 40 (so, for 1500-byte Ethernet link, set the MTU to 1460 to avoid fragmentation of packets)

mrru (*integer*: 512..65535; default: **disabled**) - maximum packet size that can be received on the link. If a packet is bigger than tunnel MTU, it will be split into multiple packets, allowing full size IP or Ethernet packets to be sent over the tunnel

- **disabled** - disable MRRU on this link

name (*name*; default: **pptp-outN**) - interface name for reference

password (*text*; default: **""**) - user password to use when logging to the remote server

profile (*name*; default: **default**) - profile to use when connecting to the remote server

user (*text*) - user name to use when logging on to the remote server

Notes

Specifying MRRU means enabling MP (Multilink PPP) over single link. This protocol is used to split big packets into smaller ones. Under Windows it can be enabled in Networking tag, Settings button, "Negotiate multi-link for single link connections". Their MRRU is hardcoded to 1614. This setting is usefull to overcome PathMTU discovery failures. The MP should be enabled on both peers.

Example

To set up PPTP client named **test2** using username **john** with password **john** to connect to the **10.1.1.12** PPTP server and use it as the default gateway:

```
[admin@MikroTik] interface pptp-client> add name=test2 connect-to=10.1.1.12 \
\... user=john add-default-route=yes password=john
[admin@MikroTik] interface pptp-client> print
Flags: X - disabled, R - running
  0 X  name="test2" max-mtu=1460 max-mru=1460 mrru=disabled connect-to=10.1.1.12
       user="john" password="john" profile=default add-default-route=yes
       allow=pap,chap,mschap1,mschap2
[admin@MikroTik] interface pptp-client> enable 0
```

Monitoring PPTP Client

Command name: ***/interface pptp-client monitor***

Property Description

encoding (*text*) - encryption and encoding (if asymmetric, separated with '/') being used in this connection

idle-time (*read-only: time*) - time since the last packet has been transmitted over this link

mru (*read-only: integer*) - effective MRU of the link

mtu (*read-only: integer*) - effective MTU of the link

status (*text*) - status of the client
- **dialing** - attempting to make a connection
- **verifying password...** - connection has been established to the server, password verification in progress
- **connected** - self-explanatory
- **terminated** - interface is not enabled or the other side will not establish a connection

uptime (*time*) - connection time displayed in days, hours, minutes and seconds

Example

Example of an established connection:

```
[admin@MikroTik] interface pptp-client> monitor test2
     status: "connected"
     uptime: 6h44m9s
  idle-time: 6h44m9s
   encoding: "MPPE128 stateless"
```

```
        mtu: 1460
        mru: 1460
[admin@MikroTik] interface pptp-client>
```

PPTP Server Setup

Home menu level: ***/interface pptp-server server***

Description

The PPTP server creates a dynamic interface for each connected PPTP client. The PPTP connection count from clients depends on the license level you have. Level1 license allows 1 PPTP client, Level3 or Level4 licenses up to 200 clients, and Level5 or Level6 licenses do not have PPTP client limitations.

Property Description

authentication (*multiple choice: pap | chap | mschap1 | mschap2*; default: **mschap2**) - authentication algorithm

default-profile - default profile to use

enabled (*yes* | *no*; default: **no**) - defines whether PPTP server is enabled or not

keepalive-timeout (*time*; default: **30**) - defines the time period (in seconds) after which the router is starting to send keepalive packets every second. If no traffic and no keepalive responses has came for that period of time (i.e. 2 * keepalive-timeout), not responding client is proclaimed disconnected

max-mru (*integer*; default: **1460**) - Maximum Receive Unit. The optimal value is the MRU of the interface the tunnel is working over decreased by 40 (so, for 1500-byte ethernet link, set the MRU to 1460 to avoid fragmentation of packets)

max-mtu (*integer*; default: **1460**) - Maximum Transmission Unit. The optimal value is the MTU of the interface the tunnel is working over decreased by 40 (so, for 1500-byte ethernet link, set the MTU to 1460 to avoid fragmentation of packets)

mrru (*integer*: 512..65535; default: **disabled**) - maximum packet size that can be received on the link. If a packet is bigger than tunnel MTU, it will be split into multiple packets, allowing full size IP or Ethernet packets to be sent over the tunnel

- **disabled** - disable MRRU on this link

Notes

Specifying MRRU means enabling MP (Multilink PPP) over single link. This protocol is used to split big packets into smaller ones. Under Windows it can be enabled in Networking tag, Settings button, "Negotiate multi-link for single link connections". Their MRRU is hardcoded to 1614. This setting is usefull to overcome PathMTU discovery failures. The MP should be enabled on both peers.

Example

To enable PPTP server:

```
[admin@MikroTik] interface pptp-server server> set enabled=yes
[admin@MikroTik] interface pptp-server server> print
            enabled: yes
```

```
             max-mtu: 1460
             max-mru: 1460
                mrru: disabled
      authentication: mschap2,mschap1
   keepalive-timeout: 30
     default-profile: default
[admin@MikroTik] interface pptp-server server>
```

PPTP Tunnel Interfaces

Home menu level: ***/interface pptp-server***

Description

There are two types of interface (tunnel) items in PPTP server configuration - static users and dynamic connections. An interface is created for each tunnel established to the given server. Static interfaces are added administratively if there is a need to reference the particular interface name (in firewall rules or elsewhere) created for the particular user. Dynamic interfaces are added to this list automatically whenever a user is connected and its username does not match any existing static entry (or in case the entry is active already, as there can not be two separate tunnel interfaces referenced by the same name). Dynamic interfaces appear when a user connects and disappear once the user disconnects, so it is impossible to reference the tunnel created for that use in router configuration (for example, in firewall), so if you need a persistent rules for that user, create a static entry for him/her. Otherwise it is safe to use dynamic configuration. **Note** that in both cases PPP users must be configured properly - static entries do not replace PPP configuration.

Property Description

client-address (*read-only: IP address*) - shows the IP address of the connected client

encoding (*read-only: text*) - encryption and encoding (if asymmetric, separated with '/') being used in this connection

mru (*read-only: integer*) - client's MRU

mtu (*read-only: integer*) - client's MTU

name (*name*) - interface name

uptime (*read-only: time*) - shows how long the client is connected

user (*name*) - the name of the user that is configured statically or added dynamically

Example

To add a static entry for **ex1** user:

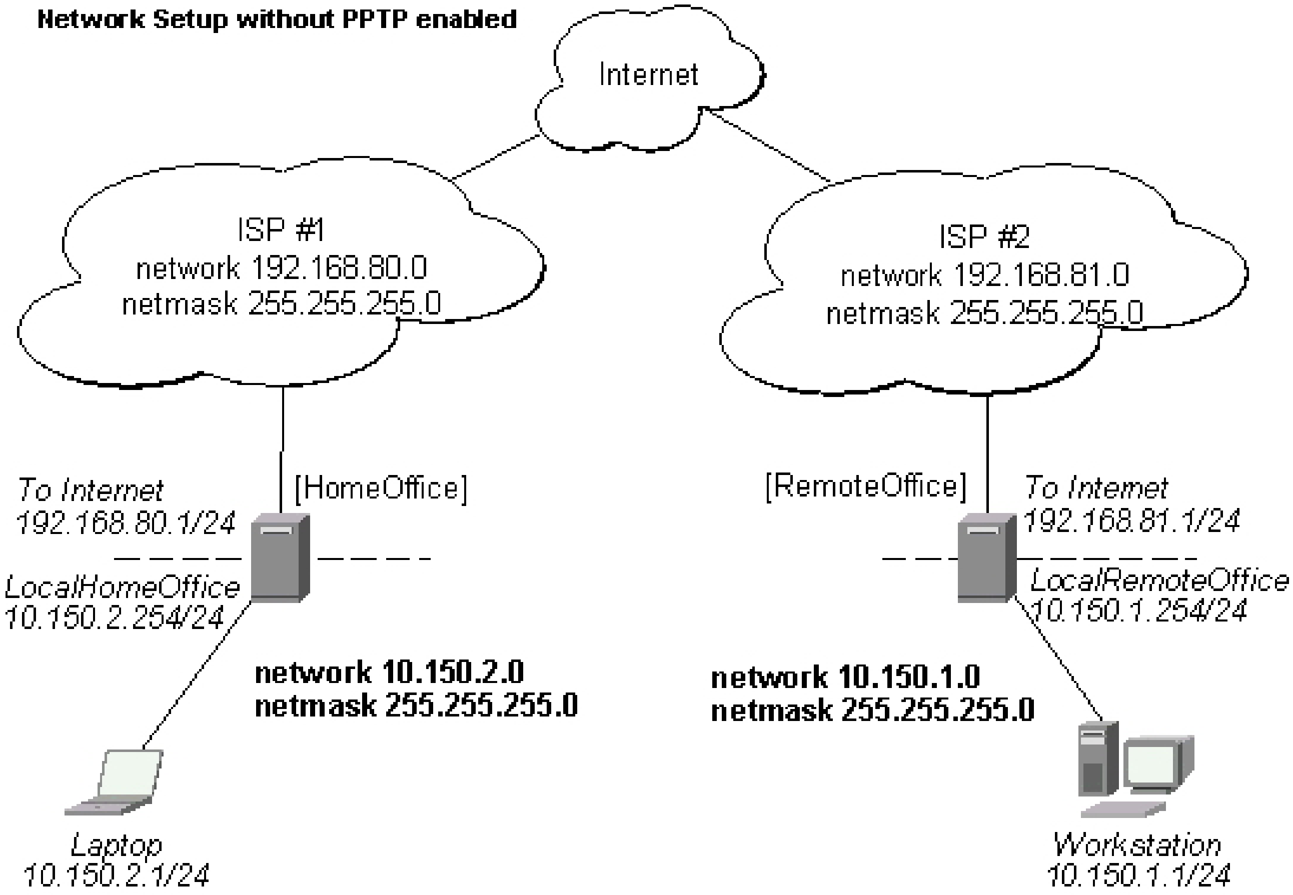
```
[admin@MikroTik] interface pptp-server> add user=ex1
[admin@MikroTik] interface pptp-server> print
Flags: X - disabled, D - dynamic, R - running
  #     NAME                 USER         MTU   CLIENT-ADDRESS  UPTIME   ENC...
  0  DR <pptp-ex>            ex           1460  10.0.0.202      6m32s    none
  1     pptp-in1             ex1
[admin@MikroTik] interface pptp-server>
```

In this example an already connected user **ex** is shown besides the one we just added. Now the interface named **pptp-in1** can be referenced from anywhere in RouterOS configuration like a regular interface.

PPTP Application Examples

Router-to-Router Secure Tunnel Example

The following is an example of connecting two Intranets using an encrypted PPTP tunnel over the Internet.



There are two routers in this example:

- [HomeOffice]
  Interface LocalHomeOffice 10.150.2.254/24
  Interface ToInternet 192.168.80.1/24
- [RemoteOffice]
  Interface ToInternet 192.168.81.1/24
  Interface LocalRemoteOffice 10.150.1.254/24

Each router is connected to a different ISP. One router can access another router through the Internet.

On the PPTP server a user must be set up for the client:

```
[admin@HomeOffice] ppp secret> add name=ex service=pptp password=lkjrht \
\... local-address=10.0.103.1 remote-address=10.0.103.2
[admin@HomeOffice] ppp secret> print detail
Flags: X - disabled
  0   name="ex" service=pptp caller-id="" password="lkjrht" profile=default
      local-address=10.0.103.1 remote-address=10.0.103.2 routes==""
```

```
[admin@HomeOffice] ppp secret>
```

Then the user should be added in the PPTP server list:

```
[admin@HomeOffice] interface pptp-server> add user=ex
[admin@HomeOffice] interface pptp-server> print
Flags: X - disabled, D - dynamic, R - running
  #     NAME                 USER         MTU   CLIENT-ADDRESS  UPTIME   ENC...
  0     pptp-in1             ex
[admin@HomeOffice] interface pptp-server>
```

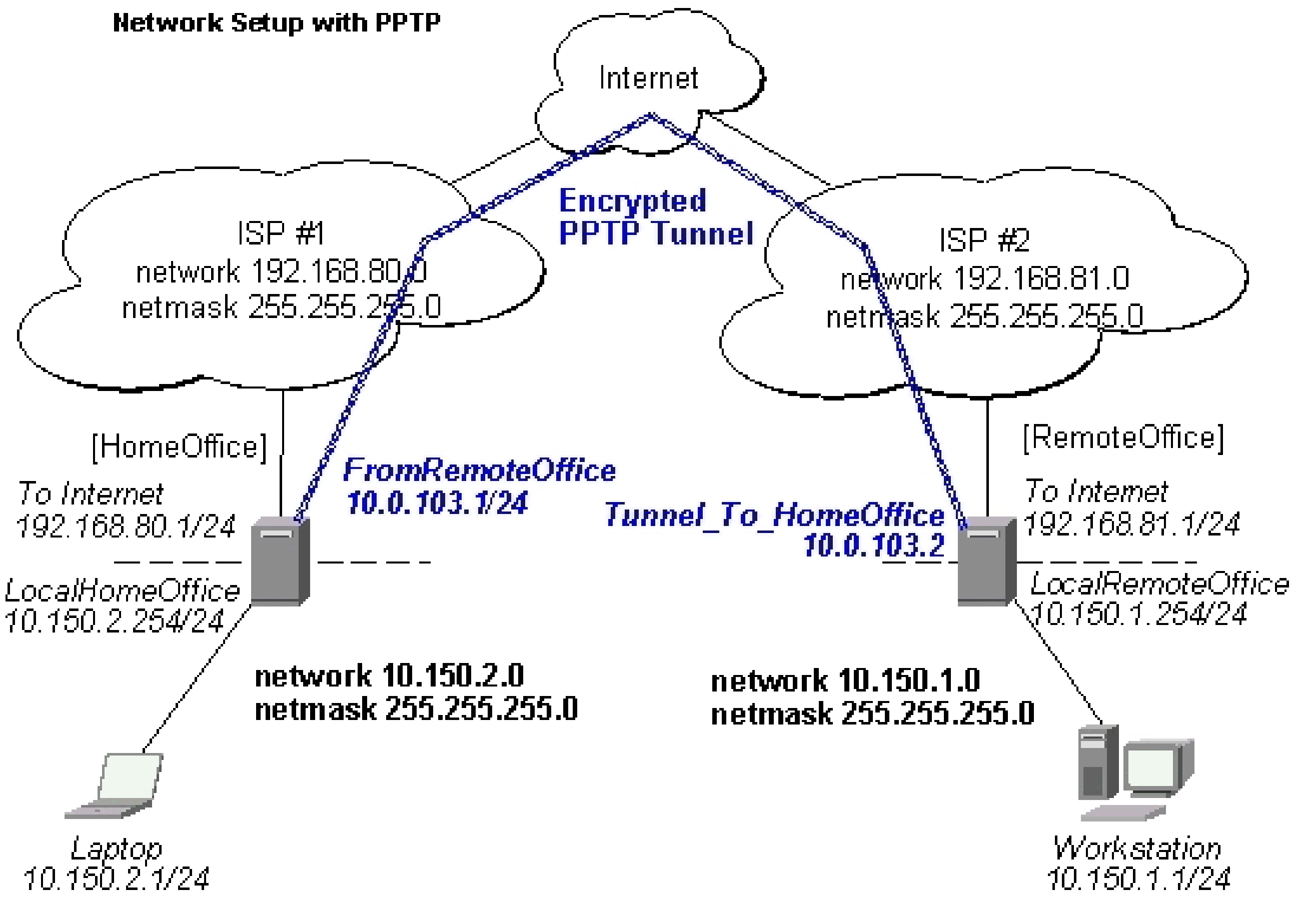
And finally, the server must be enabled:

```
[admin@HomeOffice] interface pptp-server server> set enabled=yes
[admin@HomeOffice] interface pptp-server server> print
            enabled: yes
            max-mtu: 1460
            max-mru: 1460
               mrru: disabled
     authentication: mschap2
  keepalive-timeout: 30
    default-profile: default
[admin@HomeOffice] interface pptp-server server>
```

Add a PPTP client to the RemoteOffice router:

```
[admin@RemoteOffice] interface pptp-client> add connect-to=192.168.80.1 user=ex \
\... password=lkjrht disabled=no
[admin@RemoteOffice] interface pptp-client> print
Flags: X - disabled, R - running
  0  R name="pptp-out1" mtu=1460 mru=1460 mrru=disabled connect-to=192.168.80.1
       user="ex" password="lkjrht" profile=default add-default-route=no
       allow=pap,chap,mschap1,mschap2
[admin@RemoteOffice] interface pptp-client>
```

Thus, a PPTP tunnel is created between the routers. This tunnel is like an Ethernet point-to-point connection between the routers with IP addresses 10.0.103.1 and 10.0.103.2 at each router. It enables 'direct' communication between the routers over third party networks.



To route the local Intranets over the PPTP tunnel you need to add these routes:

```
[admin@HomeOffice] > ip route add dst-address 10.150.1.0/24 gateway 10.0.103.2
[admin@RemoteOffice] > ip route add dst-address 10.150.2.0/24 gateway 10.0.103.1
```

On the PPTP server it can alternatively be done using **routes** parameter of the user configuration:

```
[admin@HomeOffice] ppp secret> print detail
Flags: X - disabled
  0   name="ex" service=pptp caller-id="" password="lkjrht" profile=default
      local-address=10.0.103.1 remote-address=10.0.103.2 routes==""

[admin@HomeOffice] ppp secret> set 0 routes="10.150.1.0/24 10.0.103.2 1"
[admin@HomeOffice] ppp secret> print detail
Flags: X - disabled
  0   name="ex" service=pptp caller-id="" password="lkjrht" profile=default
      local-address=10.0.103.1 remote-address=10.0.103.2
      routes="10.150.1.0/24 10.0.103.2 1"

[admin@HomeOffice] ppp secret>
```

Test the PPTP tunnel connection:

```
[admin@RemoteOffice]> /ping 10.0.103.1
10.0.103.1 pong: ttl=255 time=3 ms
10.0.103.1 pong: ttl=255 time=3 ms
10.0.103.1 pong: ttl=255 time=3 ms
ping interrupted
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 3/3.0/3 ms
```

Test the connection through the PPTP tunnel to the LocalHomeOffice interface:

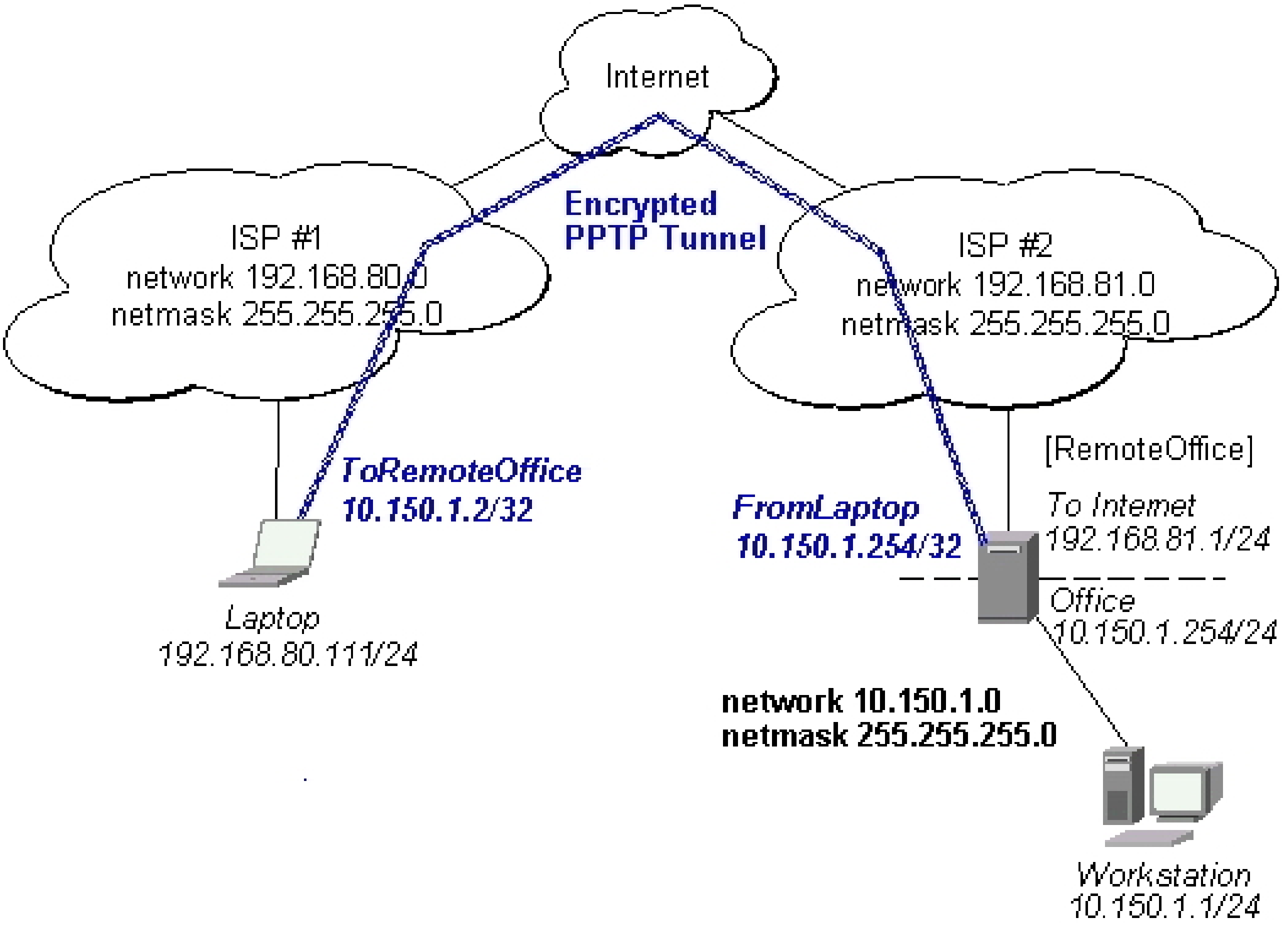
```
[admin@RemoteOffice]> /ping 10.150.2.254
10.150.2.254 pong: ttl=255 time=3 ms
10.150.2.254 pong: ttl=255 time=3 ms
10.150.2.254 pong: ttl=255 time=3 ms
ping interrupted
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 3/3.0/3 ms
```

To bridge a LAN over this secure tunnel, please see the example in the 'EoIP' section of the manual. To set the maximum speed for traffic over this tunnel, please consult the 'Queues' section.

Connecting a Remote Client via PPTP Tunnel

The following example shows how to connect a computer to a remote office network over PPTP encrypted tunnel giving that computer an IP address from the same network as the remote office has (without need of bridging over EoIP tunnels)

Please, consult the respective manual on how to set up a PPTP client with the software You are using.



The router in this example:

- [RemoteOffice]
  Interface ToInternet 192.168.81.1/24
  Interface Office 10.150.1.254/24

The client computer can access the router through the Internet.

On the PPTP server a user must be set up for the client:

```
[admin@RemoteOffice] ppp secret> add name=ex service=pptp password=lkjrht
local-address=10.150.1.254 remote-address=10.150.1.2
[admin@RemoteOffice] ppp secret> print detail
Flags: X - disabled
  0   name="ex" service=pptp caller-id="" password="lkjrht" profile=default
      local-address=10.150.1.254 remote-address=10.150.1.2 routes==""

[admin@RemoteOffice] ppp secret>
```

Then the user should be added in the PPTP server list:

```
[admin@RemoteOffice] interface pptp-server> add name=FromLaptop user=ex
[admin@RemoteOffice] interface pptp-server> print
Flags: X - disabled, D - dynamic, R - running
  #     NAME                 USER          MTU   CLIENT-ADDRESS  UPTIME   ENC...
  0     FromLaptop           ex
[admin@RemoteOffice] interface pptp-server>
```

And the server must be enabled:

```
[admin@RemoteOffice] interface pptp-server server> set enabled=yes
[admin@RemoteOffice] interface pptp-server server> print
            enabled: yes
            max-mtu: 1460
            max-mru: 1460
               mrru: disabled
     authentication: mschap2
  keepalive-timeout: 30
    default-profile: default
[admin@RemoteOffice] interface pptp-server server>
```

Finally, the proxy APR must be enabled on the 'Office' interface:

```
[admin@RemoteOffice] interface ethernet> set Office arp=proxy-arp
[admin@RemoteOffice] interface ethernet> print
Flags: X - disabled, R - running
  #    NAME                 MTU   MAC-ADDRESS       ARP
  0  R ToInternet           1500  00:30:4F:0B:7B:C1 enabled
  1  R Office               1500  00:30:4F:06:62:12 proxy-arp
[admin@RemoteOffice] interface ethernet>
```

PPTP Setup for Windows

Microsoft provides PPTP client support for Windows NT, 2000, ME, 98SE, and 98. Windows 98SE, 2000, and ME include support in the Windows setup or automatically install PPTP. For 95, NT, and 98, installation requires a download from Microsoft. Many ISPs have made help pages to assist clients with Windows PPTP installation.

- http://www.real-time.com/Customer_Support/PPTP_Config/pptp_config.html
- http://www.microsoft.com/windows95/downloads/contents/WUAdminTools/S_WUNetworkingTools/W95WinsockU

Sample instructions for PPTP (VPN) installation and client setup - Windows 98SE

If the VPN (PPTP) support is installed, select 'Dial-up Networking' and 'Create a new connection'. The option to create a 'VPN' should be selected. If there is no 'VPN' options, then follow the installation instructions below. When asked for the 'Host name or IP address of the VPN server', type the IP address of the router. Double-click on the 'new' icon and type the correct user name and password (must also be in the user database on the router or RADIUS server used for authentication).

The setup of the connections takes nine seconds after selection the 'connect' button. It is suggested that the connection properties be edited so that 'NetBEUI', 'IPX/SPX compatible', and 'Log on to network' are unselected. The setup time for the connection will then be two seconds after the 'connect' button is selected.

To install the 'Virtual Private Networking' support for Windows 98SE, go to the 'Setting' menu from the main 'Start' menu. Select 'Control Panel', select 'Add/Remove Program', select the 'Windows setup' tab, select the 'Communications' software for installation and 'Details'. Go to the bottom of the list of software and select 'Virtual Private Networking' to be installed.

Troubleshooting

Description

- **I use firewall and I cannot establish PPTP connection**
  Make sure the TCP connections to port 1723 can pass through both directions between your sites. Also, IP protocol 47 should be passed through

VLAN

Document revision 1.3 (October 11, 2007, 17:38 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

VLAN is an implementation of the 802.1Q VLAN protocol for MikroTik RouterOS. It allows you to have multiple Virtual LANs on a single ethernet or wireless interface, giving the ability to segregate LANs efficiently. It supports up to 4095 VLAN interfaces, each with a unique VLAN ID, per ethernet device. VLAN priorites may also be used and manipulated. Many routers, including Cisco and Linux based, and many Layer 2 switches use VLAN to enable multiple independent, isolated networks to exist on the same physical network.

A VLAN is a logical grouping that allows end users to communicate as if they were physically connected to a single isolated LAN, independent of the physical configuration of the network. VLAN support adds a new dimension of security and cost savings permitting the sharing of a physical network infrastructure and interfaces/ports while logically maintaining separation among unrelated users.

Specifications

Packages required: ***system***
License required: ***level1 (limited to 1 vlan), level3***
Home menu level: ***/interface vlan***
Standards and Technologies: ***VLAN (IEEE 802.1Q)***
Hardware usage: ***Not significant***

Description

VLANs are simply a way of grouping a set of switch ports together so that they form a logical network, separate from any other such group. It may also be understood as breaking one physical switch into several independent parts. Within a single switch this is straightforward local configuration. When the VLAN extends over more than one switch, the inter-switch links have to become trunks, on which packets are tagged to indicate which VLAN they belong to.

You can use MikroTik RouterOS (as well as Cisco IOS, Linux and other router systems) to mark these packets as well as to accept and route marked ones.

As VLAN works on OSI Layer 2, it can be used just as any other network interface without any restrictions. VLAN successfully passes through regular Ethernet bridges.

You can also transport VLANs over wireless links and put multiple VLAN interfaces on a single wireless interface. Note that as VLAN is not a full tunnel protocol (i.e., it does not have additional fields to transport MAC addresses of sender and recipient), the same limitation applies to bridging over VLAN as to bridging plain wireless interfaces. In other words, while wireless clients may participate in VLANs put on wireless interfaces, it is not possible to have VLAN put on a wireless interface in **station** mode bridged with any other interface.

Currently supported Ethernet interfaces

This is a list of network interfaces on which VLAN was tested and worked. Note that there might be many other interfaces that support VLAN, but they just were not checked. Most modern Ethernet interfaces support VLAN.

- Realtek 8139
- Intel PRO/100
- Intel PRO1000 server adapter
- National Semiconductor DP83816 based cards (RouterBOARD200 onboard Ethernet, RouterBOARD 24 card)
- National Semiconductor DP83815 (Soekris onboard Ethernet)
- VIA VT6105M based cards (RouterBOARD 44 card)
- VIA VT6105
- VIA VT6102 (VIA EPIA onboard Ethernet)

This is a list of network interfaces on which VLAN was tested and worked, but **WITHOUT LARGE PACKET (>1496 bytes) SUPPORT**:

- 3Com 3c59x PCI
- DEC 21140 (tulip)

Additional Documents

- http://www.csd.uwo.ca/courses/CS457a/reports/handin/jpbojtos/A2/trunking.htm
- http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121newft/121t/121t3/dtbridge.htm#xtocid114533
- http://www.cisco.com/warp/public/473/27.html#tagging
- http://www.cisco.com/warp/public/538/7.html

- http://www.nwfusion.com/news/tech/2001/0305tech.html
- http://www.intel.com/network/connectivity/resources/doc_library/tech_brief/virtual_lans.htm

VLAN Setup

Home menu level: ***/interface vlan***

Property Description

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; default: **enabled**) - Address Resolution Protocol mode

- **disabled** - the interface will not use ARP protocol
- **enabled** - the interface will fully use ARP protocol
- **proxy-arp** - the interface will be an ARP proxy
- **reply-only** - the interface will only reply to the requests for to its own IP addresses, but neighbor MAC addresses will be gathered from /ip arp statically set table only

interface (*name*) - physical interface to the network where the VLAN is put

mtu (*integer*; default: **1500**) - Maximum Transmission Unit

name (*name*) - interface name for reference

vlan-id (*integer*; default: **1**) - Virtual LAN identifier or tag that is used to distinguish VLANs. Must be equal for all computers that belong to the same VLAN.

Notes

MTU should be set to 1500 bytes as on Ethernet interfaces. But this may not work with some Ethernet cards that do not support receiving/transmitting of full size Ethernet packets with VLAN header added (1500 bytes data + 4 bytes VLAN header + 14 bytes Ethernet header). In this situation MTU 1496 can be used, but note that this will cause packet fragmentation if larger packets have to be sent over interface. At the same time remember that MTU 1496 may cause problems if path MTU discovery is not working properly between source and destination.

Example

To add and enable a VLAN interface named **test** with **vlan-id**=1 on interface **ether1**:

```
[admin@MikroTik] interface vlan> add name=test vlan-id=1 interface=ether1
[admin@MikroTik] interface vlan> print
Flags: X - disabled, R - running
  #    NAME                 MTU  ARP       VLAN-ID INTERFACE
  0 X  test                 1500 enabled   1       ether1
[admin@MikroTik] interface vlan> enable 0
[admin@MikroTik] interface vlan> print
Flags: X - disabled, R - running
  #    NAME                 MTU  ARP       VLAN-ID INTERFACE
  0  R test                 1500 enabled   1       ether1
[admin@MikroTik] interface vlan>
```

Application Example

VLAN example on MikroTik Routers

Let us assume that we have two or more MikroTik RouterOS routers connected with a hub. Interfaces to the physical network, where the VLAN is to be created is **ether1** for all of them (it is needed only for example simplification, it is NOT a must).

To connect computers through VLAN they must be connected physically and unique IP addresses should be assigned them so that they could ping each other. Then on each of them the VLAN interface should be created:

```
[admin@MikroTik] interface vlan> add name=test vlan-id=32 interface=ether1
[admin@MikroTik] interface vlan> print
Flags: X - disabled, R - running
  #    NAME                 MTU  ARP       VLAN-ID INTERFACE
  0  R test                 1500 enabled   32      ether1
[admin@MikroTik] interface vlan>
```

If the interfaces were successfully created, both of them will be **running**. If computers are connected incorrectly (through network device that does not retransmit or forward VLAN packets), either both or one of the interfaces will not be **running**.

When the interface is running, IP addresses can be assigned to the VLAN interfaces.

On the Router 1:

```
[admin@MikroTik] ip address> add address=10.10.10.1/24 interface=test
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.204/24      10.0.0.0        10.0.0.255      ether1
  1   10.20.0.1/24       10.20.0.0       10.20.0.255     pc1
  2   10.10.10.1/24      10.10.10.0      10.10.10.255    test
[admin@MikroTik] ip address>
```

On the Router 2:

```
[admin@MikroTik] ip address> add address=10.10.10.2/24 interface=test
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.201/24      10.0.0.0        10.0.0.255      ether1
  1   10.10.10.2/24      10.10.10.0      10.10.10.255    test
[admin@MikroTik] ip address>
```

If it set up correctly, then it is possible to ping Router 2 from Router 1 and vice versa:

```
[admin@MikroTik] ip address> /ping 10.10.10.1
10.10.10.1 64 byte pong: ttl=255 time=3 ms
10.10.10.1 64 byte pong: ttl=255 time=4 ms
10.10.10.1 64 byte pong: ttl=255 time=10 ms
10.10.10.1 64 byte pong: ttl=255 time=5 ms
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 3/10.5/10 ms
[admin@MikroTik] ip address> /ping 10.10.10.2
10.10.10.2 64 byte pong: ttl=255 time=10 ms
10.10.10.2 64 byte pong: ttl=255 time=11 ms
10.10.10.2 64 byte pong: ttl=255 time=10 ms
10.10.10.2 64 byte pong: ttl=255 time=13 ms
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 10/11/13 ms
[admin@MikroTik] ip address>
```

Graphing

Document revision 1.3 (February 6, 2008, 1:44 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Graphing is a tool which is used for monitoring various RouterOS parameters over a period of time.

Specifications

Packages required: ***system, routerboard (optional)***
License required: ***level1***
Home menu level: ***/tool graphing***
Hardware usage: ***Not significant***

Description

The Graphing tool can display graphics for:

- Routerboard health (voltage and temperature)
- Resource usage (CPU, Memory and Disk usage)
- Traffic which is passed through interfaces
- Traffic which is passed through simple queues

Graphing consists of two parts - first part collects information and other part displays data in a Web page. To access the graphics, type **http://[Router_IP_address]/graphs/** and choose a graphic to display in your Web browser.

Data from the router is gathered every 5 minutes, but saved on the system drive every **store-every** time. After rebooting the router, graphing will display information that was last time saved on the disk before the reboot.

RouterOS generates four graphics for each item:

- "Daily" Graph (5 Minute Average)
- "Weekly" Graph (30 Minute Average)
- "Monthly" Graph (2 Hour Average)
- "Yearly" Graph (1 Day Average)

To access each graphic from a network, specify this network in **allow-address** parameter for the respective item.

General Options

Home menu level: ***/tool graphing***

Property Description

store-every (*5min* | *hour* | *24hours*; default: **5min**) - how often to store information on system drive

Example

To store information on system drive every hour:

```
/tool graphing set store-every=hour
[admin@MikroTik] tool graphing> print
    store-every: hour
[admin@MikroTik] tool graphing>
```

Health Graphing

Home menu level: ***/tool graphing health***

Description

This submenu provides information about RouterBoard's 'health' - voltage and temperature. For this option, you have to install the **routerboard** package:

Property Description

allow-address (*IP addressnetmask*; default: **0.0.0.0/0**) - network which is allowed to view graphs of router health

store-on-disk (yes | no; default: **yes**) - whether to store information about traffic on system drive or not. If not, the information will be stored in RAM and will be lost after a reboot

Interface Graphing

Home menu level: ***/tool graphing interface***

Description

Shows how much traffic is passed through an interface over a period of time.

Property Description

allow-address (*IP addressnetmask*; default: **0.0.0.0/0**) - network which is allowed to view graphs of router health

interface (*name*; default: **all**) - name of the interface which will be monitored

store-on-disk (yes | no; default: **yes**) - whether to store information about traffic on system drive or not. If not, the information will be stored in RAM and will be lost after a reboot

Example

To monitor traffic which is passed through interface **ether1** only from local network **192.168.0.0/24**, and write information on disk:

```
[admin@MikroTik] tool graphing interface> add interface=ether1 \
\... allow-address=192.168.0.0/24 store-on-disk=yes
[admin@MikroTik] tool graphing interface> print
Flags: X - disabled
 #   INTERFACE ALLOW-ADDRESS      STORE-ON-DISK
 0   ether1    192.168.0.0/24     yes
[admin@MikroTik] tool graphing interface>
```

Simple Queue Graphing

Home menu level: ***/tool graphing queue***

Description

In this submenu you can specify a queue from the **/queue simple** list to make a graphic for it.

Property Description

allow-address (*IP addressnetmask*; default: **0.0.0.0/0**) - network which is allowed to view graphs of router health

allow-target (yes | no; default: **yes**) - whether to allow access to web graphing from IP range that is specified in /queue simple target-address

simple-queue (*name*; default: **all**) - name of simple queue which will be monitored

store-on-disk (yes | no; default: **yes**) - whether to store information about traffic on hard drive or not. If not, the information will be stored in RAM and will be lost after a reboot

Example

Add a simple queue to Grapher list with simple-queue name **queue1**, allow limited clients to access Grapher from web, store information about traffic on disk:

```
[admin@MikroTik] tool graphing queue> add simple-queue=queue1 allow-address=yes \
\... store-on-disk=yes
```

Resource Graphing

Home menu level: ***/tool graphing resource***

Description

Provides with router resource usage information over a period of time:

- CPU usage
- Memory usage
- Disk usage

Property Description

allow-address (*IP addressnetmask*; default: **0.0.0.0/0**) - network which is allowed to view graphs of router health

store-on-disk (yes | no; default: **yes**) - whether to store information about traffic on hard drive or not. If not, the information will be stored in RAM and will be lost after a reboot

Example

Add IP range **192.168.0.0/24** from which users are allowed to monitor Grapher's resource usage:

```
[admin@MikroTik] tool graphing resource> add allow-address=192.168.0.0/24 \
\... store-on-disk=yes
[admin@MikroTik] tool graphing resource> print
Flags: X - disabled
 #   ALLOW-ADDRESS      STORE-ON-DISK
 0   192.168.0.0/24     yes
[admin@MikroTik] tool graphing resource>
```

HotSpot User AAA

Document revision 2.4 (February 6, 2008, 1:40 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

This document provides information on authentication, authorization and accounting parameters and configuration for HotSpot gateway system.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip hotspot user***
Standards and Technologies: ***RADIUS***
Hardware usage: *Local traffic accounting requires additional memory*

Description

HotSpot User Profiles

Home menu level: ***/ip hotspot user profile***

Description

HotSpot User profiles are used for common user settings. Profiles are like user groups, they are grouping users with the same limits.

Property Description

address-pool (*namenone*; default: **none**) - the IP pool name which the users will be given IP addresses from. This works like dhcp-pool method in earlier versions of MikroTik RouterOS, except that it does not use DHCP, but rather the embedded one-to-one NAT

- **none** - do not reassign IP addresses to the users of this profile

advertise (yes | no; default: **no**) - whether to enable forced advertisement popups for this profile

advertise-interval (*multiple choice: time*; default: **30m,10m**) - set of intervals between showing advertisement popups. After the list is done, the last value is used for all further advertisements

advertise-timeout (*timeimmediately* | *never*; default: **1m**) - how long to wait for advertisement to be shown, before blocking network access with walled-garden

advertise-url (*multiple choice: text*; default: **http://www.mikrotik.com/,http://www.routerboard.com/**) - list of URLs to show as advertisement popups. The list is cyclic, so when the last item reached, next time the first is shown

idle-timeout (*timenone*; default: **none**) - idle timeout (maximal period of inactivity) for authorized clients. It is used to detect, that client is not using outer networks (e.g. Internet), i.e., there is NO TRAFFIC coming from that client and going through the router. Reaching the timeout, user will be logged out, dropped of the host list, the address used by the user will be freed, and the session time accounted will be decreased by this value

- **none** - do not timeout idle users

incoming-filter (*name*) - name of the firewall chain applied to incoming packets from the users of this profile

incoming-packet-mark (*name*) - packet mark put on all the packets from every user of this profile automatically

keepalive-timeout (*timenone*; default: **00:02:00**) - keepalive timeout for authorized clients. Used to detect, that the computer of the client is alive and reachable. If check will fail during this period, user will be logged out, dropped of the host list, the address used by the user will be freed, and the session time accounted will be decreased by this value

- **none** - do not timeout unreachable users

name (*name*) - profile reference name

on-login (*text*; default: **""**) - script name to launch after a user has logged in

on-logout (*text*; default: **""**) - script name to launch after a user has logged out

open-status-page (*always* | *http-login*; default: **always**) - whether to show status page also for users authenticated using mac login method. Useful if you want to put some information (for example, banners or popup windows) in the alogin.html page so that all users would see it

- **http-login** - open status page only in case of HTTP login (including cookie and https login methods)
- **always** - open the status page in case of mac login as well once the user opens any web page

outgoing-filter (*name*) - name of the firewall chain applied to outgoing packets to the users of this profile

outgoing-packet-mark (*name*) - packet mark put on all the packets to every user of this profile automatically

rate-limit (*text*; default: **""**) - Rate limitation in form of rx-rate[/tx-rate] [rx-burst-rate[/tx-burst-rate] [rx-burst-threshold[/tx-burst-threshold] [rx-burst-time[/tx-burst-time] [priority] [rx-rate-min[/tx-rate-min]]]] from the point of view of the router (so "rx" is client upload, and "tx" is client download). All rates should be numbers with optional 'k' (1,000s) or 'M' (1,000,000s). If tx-rate is not specified, rx-rate is as tx-rate too. Same goes for tx-burst-rate and tx-burst-threshold and tx-burst-time. If both rx-burst-threshold and tx-burst-threshold are not specified (but burst-rate is specified), rx-rate and tx-rate is used as burst thresholds. If both rx-burst-time and tx-burst-time are not specified, 1s is used as default. Priority takes values 1..8, where 1 implies the highest priority, but 8 - the lowest. If rx-rate-min and tx-rate-min are not specified rx-rate and tx-rate values are used. The rx-rate-min and tx-rate-min values can not exceed rx-rate and tx-rate values.

session-timeout (*time*; default: **0s**) - session timeout (maximal allowed session time) for client. After this time, the user will be logged out unconditionally

- **0** - no timeout

shared-users (*integer*; default: **1**) - maximal number of simultaneously logged in users with the same username

status-autorefresh (*timenone*; default: **none**) - HotSpot servlet status page autorefresh interval

transparent-proxy (yes | no; default: **yes**) - whether to use transparent HTTP proxy for the authorized users of this profile

Notes

When idle-timeout or keepalive is reached, session-time for that user is reduced by the actual period of inactivity in order to prevent the user from being overcharged.

Example

HotSpot Users

Home menu level: ***/ip hotspot user***

Property Description

address (*IP address*; default: **0.0.0.0**) - static IP address. If not 0.0.0.0, client will always get the same IP address. A configured address implies, that only one simultaneous login for that user is allowed. Any existing address will be replaced with this one using the embedded one-to-one NAT

bytes-in (*read-only: integer*) - total amount of bytes received from user

bytes-out (*read-only: integer*) - total amount of bytes sent to user

email (*text*) - e-mail address. Only basic syntax checking is done to ensure validity of this field

limit-bytes-in (*integer*; default: **0**) - maximum amount of bytes user can transmit (i.e., bytes received from the user)

- **0** - no limit

limit-bytes-out (*integer*; default: **0**) - maximum amount of bytes user can receive (i.e., bytes sent to the user)

- **0** - no limit

limit-bytes-total (*integer*; default: **0**) - maximum aggregate amount of bytes user can receive and send (i.e., the sum of the amount of bytes sent to the user and received from it)

- **0** - no limit

limit-uptime (*time*; default: **0s**) - total uptime limit for user (pre-paid time)

- **0s** - no limit

mac-address (*MAC address*; default: **00:00:00:00:00:00**) - static MAC address. If not 00:00:00:00:00:00, client is allowed to login only from that MAC address

name (*name*) - user name. If authentication method is trial, then user name will be set automaticly after following pattern "T-MAC_adress", where MAC_address is trial user Mac address

packets-in (*read-only: integer*) - total amount of packets received from user (i.e., packets received from the user)

packets-out (*read-only: integer*) - total amount of packets sent to user (i.e., packets sent to the user)

password (*text*) - user password

profile (*name*; default: **default**) - user profile

routes (*text*) - routes that are to be registered on the HotSpot gateway when the client is connected. The route format is: dst-address [[gateway] [metric]] (for example, 10.1.0.0/24 10.0.0.1 1). Several routes may be specified separated with commas. If gateway is not specified, the remote address is used. If metric is not speciefied, the metric of 1 is used

server (*nameall*; default: **all**) - which HotSpot server is this user allowed to log in to

uptime (*read-only: time*) - total time user has been logged in

Notes

In case of **mac** authentication method, clients' MAC addresses can be used as usernames (without password)

The byte limits are total limits for each user (not for each session as at **/ip hotspot active**). So, if a user has already downloaded something, then session limit will show the total limit - (minus) already downloaded. For example, if download limit for a user is 100MB and the user has already downloaded 30MB, then session download limit after login at **/ip hotspot active** will be 100MB - 30MB = 70MB.

Should a user reach his/her limits (bytes-in >= limit-bytes-in or bytes-out >= limit-bytes-out), he/she will not be able to log in anymore.

The statistics is updated if a user is authenticated via local user database each time he/she logs out. It means, that if a user is currently logged in, then the statistics will not show current total values. Use **/ip hotspot active** submenu to view the statistics on the current user sessions.

If the user has IP address specified, only one simultaneous login is allowed. If the same credentials are used again when the user is still active, the active one will be automatically logged off.

Trial users will have dynamic records here with their **name** written as "T-[mac]" (where [mac] is the user's MAC address, without the brackets), **email** set to the password the user has supplied, **mac-address** - the client's MAC address, **profile** and **limit-uptime** - the respective values of **trial-user-profile** and **trial-uptime** limit properties of the HotSpot server profile. The entries will be automatically removed once the trial user times out (after **trial-uptime** reset time).

Example

To add user **ex** with password **ex** that is allowed to log in only with **01:23:45:67:89:AB** MAC address and is limited to 1 hour of work:

```
[admin@MikroTik] ip hotspot user> add name=ex password=ex \
\... mac-address=01:23:45:67:89:AB limit-uptime=1h
[admin@MikroTik] ip hotspot user> print
Flags: X - disabled
 #   SERVER     NAME                       ADDRESS         PROFILE UPTIME
 0              ex                                         default 00:00:00
[admin@MikroTik] ip hotspot user> print detail
Flags: X - disabled, D - dynamic
 0   name="ex" password="ex" mac-address=01:23:45:67:89:AB profile=default
limit-uptime=1h uptime=0s bytes-in=0 bytes-out=0 packets-in=0 packets-out=0
[admin@MikroTik] ip hotspot user>
```

HotSpot Active Users

Home menu level: ***/ip hotspot active***

Description

The active user list shows the list of currently logged in users. Nothing can be changed here, except user can be logged out with the **remove** command.

Property Description

address (*read-only: IP address*) - IP address of the user

blocked (*read-only: flag*) - whether the user is blocked by advertisement (i.e., usual due advertisement is pending)

bytes-in (*read-only: integer*) - how many bytes did the router receive from the client

bytes-out (*read-only: integer*) - how many bytes did the router send to the client

domain (*read-only: text*) - domain of the user (if split from username)

idle-time (*read-only: time*) - the amount of time has the user been idle

idle-timeout (*read-only: time*) - the exact value of idle-timeout that applies to this user. This property shows how long should the user stay idle for it to be logged off automatically

keepalive-timeout (*read-only: time*) - the exact value of keepalive-timeout that applies to this user. This property shows how long should the user's computer stay out of reach for it to be logged off automatically

limit-bytes-in (*read-only: integer*) - maximal amount of bytes the user is allowed to send to the router

limit-bytes-out (*read-only: integer*) - maximal amount of bytes the router is allowed to send to the client

limit-bytes-total (*read-only: integer*) - maximal aggregate amount of bytes the router is allowed to send to the client and receive form it

login-by (*multiple choice, read-only: cookie* | *http-chap* | *http-pap* | *https* | *mac* | *trial*) - authentication method used by user

mac-address (*read-only: MAC address*) - actual MAC address of the user

packets-in (*read-only: integer*) - how many packets did the router receive from the client

packets-out (*read-only: integer*) - how many packets did the router send to the client

radius (*read-only: flag*) - whether the user was authenticated via RADIUS

server (*read-only: name*) - the particular HotSpot server the used is logged on at.

session-time-left (*read-only: time*) - the exact value of session-time-left that applies to this user. This property shows how long should the user stay logged-in (see uptime) for it to be logged off automatically

uptime (*read-only: time*) - current session time of the user (i.e., how long has the user been logged in)

user (*read-only: name*) - name of the user

Example

To get the list of active users:

```
[admin@MikroTik] ip hotspot active> print
Flags: R - radius, B - blocked
 #    USER         ADDRESS         UPTIME       SESSION-TIME-LEFT IDLE-TIMEOUT
 0    ex           10.0.0.144      4m17s        55m43s
[admin@MikroTik] ip hotspot active>
```

IP accounting

Document revision 2.2 (February 6, 2008, 1:40 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Authentication, Authorization and Accounting feature provides a possibility of local and/or remote (on RADIUS server) Point-to-Point and HotSpot user management and traffic accounting (all IP traffic passing the router is accounted; local traffic acocunting is an option).

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/user, /ppp, /ip accounting, /radius***
Standards and Technologies: ***RADIUS***
Hardware usage: *Traffic accounting requires additional memory*

Local IP Traffic Accounting

Home menu level: ***/ip accounting***

Description

As each packet passes through the router, the packet source and destination addresses are matched against an IP pair list in the accounting table and the traffic for that pair is increased. The traffic of PPP, PPTP, PPPoE, ISDN and HotSpot clients can be accounted on per-user basis too. Both the number of packets and the number of bytes are accounted.

If no matching IP or user pair exists, a new entry will be added to the table.

Only the packets that enter and leave the router are accounted. Packets that are dropped in the router are not counted. Packets that are NATted on the router will be accounted for with the actual IP addresses on each side. Packets that are going through bridged interfaces (i.e. inside the bridge interface) are also counted correctly.

Traffic, generated by the router itself, and sent to it, may as well be accounted.

Property Description

account-local-traffic (yes | no; default: **no**) - whether to account the traffic to/from the router itself

enabled (yes | no; default: **no**) - whether local IP traffic accounting is enabled

threshold (*integer*; default: **256**) - maximum number of IP pairs in the accounting table (maximal value is 8192)

Notes

For bidirectional connections two entries will be created.

Each IP pair uses approximately 100 bytes

When the threshold limit is reached, no new IP pairs will be added to the accounting table. Each packet that is not accounted in the accounting table will then be added to the **uncounted** counter!

Example

Enable IP accounting:

```
[admin@MikroTik] ip accounting> set enabled=yes
[admin@MikroTik] ip accounting> print
                enabled: yes
  account-local-traffic: no
              threshold: 256
[admin@MikroTik] ip accounting>
```

Local IP Traffic Accounting Table

Home menu level: ***/ip accounting snapshot***

Description

When a snapshot is made for data collection, the accounting table is cleared and new IP pairs and traffic data are added. The more frequently traffic data is collected, the less likelihood that the IP pairs thereshold limit will be reached.

Property Description

bytes (*read-only: integer*) - total number of bytes, matched by this entry

dst-address (*read-only: IP address*) - destination IP address

dst-user (*read-only: text*) - recipient's name (if aplicable)

packets (*read-only: integer*) - total number of packets, matched by this entry

src-address (*read-only: IP address*) - source IP address

src-user (*read-only: text*) - sender's name (if aplicable)

Notes

Usernames are shown only if the users are connected to the router via a PPP tunnel or are authenticated by HotSpot.

You should "take" snapshot in order to review the current state of the table by issueing the **take** command. Before the first snapshot has been taken, the table is empty.

Example

To take a new snapshot:

```
[admin@MikroTik] ip accounting snapshot> take
[admin@MikroTik] ip accounting snapshot> print
 # SRC-ADDRESS     DST-ADDRESS     PACKETS    BYTES      SRC-USER    DST-USER
 0 192.168.0.2     159.148.172.197 474        19130
 1 192.168.0.2     10.0.0.4        3          120
 2 192.168.0.2     192.150.20.254  32         3142
 3 192.150.20.254  192.168.0.2     26         2857
 4 10.0.0.4        192.168.0.2     2          117
 5 159.148.147.196 192.168.0.2     2          136
 6 192.168.0.2     159.148.147.196 1          40
 7 159.148.172.197 192.168.0.2     835        1192962
[admin@MikroTik] ip accounting snapshot>
```

Web Access to the Local IP Traffic Accounting Table

Home menu level: ***/ip accounting web-access***

Description

The web page report make it possible to use the standard Unix/Linux tool wget to collect the traffic data and save it to a file or to use MikroTik shareware Traffic Counter to display the table. If the web report is enabled and the web page is viewed, the **snapshot** will be made when connection is initiated to the web page. The **snapshot** will be displayed on the web page. TCP protocol, used by http connections with the wget tool guarantees that none of the traffic data will be lost. The **snapshot** image will be made when the connection from wget is initiated. Web browsers or wget should connect to URL: **http://routerIP/accounting/ip.cgi**

Property Description

accessible-via-web (yes | no; default: **no**) - wheather the snapshot is available via web

address (*IP addressnetmask*; default: **0.0.0.0**) - IP address range that is allowed to access the snapshot

Example

To enable web access from **10.0.0.1** server only:

```
[admin@MikroTik] ip accounting web-access> set accessible-via-web=yes \
\... address=10.0.0.1/32
[admin@MikroTik] ip accounting web-access> print
    accessible-via-web: yes
               address: 10.0.0.1/32
[admin@MikroTik] ip accounting web-access>
```

Uncounted Connections

Home menu level: ***/ip accounting uncounted***

Description

In case no more IP pairs can be added to the accounting table (the accounting threshold has been reached), all traffic that does not belong to any of the known IP pairs is summed together and totals are shown in this menu

Property Description

bytes (*read-only: integer*) - byte count

packets (*read-only: integer*) - packet count

Example

See the uncounted packets:

```
[admin@MikroTik] ip accounting uncounted> print
    packets: 0
      bytes: 0
[admin@MikroTik] ip accounting uncounted>
```

PPP User AAA

Document revision 2.6 (February 6, 2008, 1:40 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

This documents provides summary, configuration reference and examples on PPP user management. This includes asynchronous PPP, PPTP, L2TP, OpenVPN, PPPoE and ISDN users.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ppp***

Description

The MikroTik RouterOS provides scalable Authentication, Athorization and Accounting (AAA) functionality.

Local authentication is performed using the User Database and the Profile Database. The actual configuration for the given user is composed using respective user record from the User Database, associated item from the Profile Database and the item in the Profile database which is set as default for a given service the user is authenticating to. Default profile settings from the Profile database have lowest priority while the user access record settings from the User Database have highest priority with the only exception being particular IP addresses take precedence over IP pools in the **local-address** and **remote-address** settings, which described later on.

Support for RADIUS authentication gives the ISP or network administrator the ability to manage PPP user access and accounting from one server throughout a large network. The MikroTik RouterOS has a RADIUS client which can authenticate for PPP, PPPoE, PPTP, L2TP, OpenVPN and ISDN connections. The attributes received from RADIUS server override the ones set in the default profile, but if some parameters are not received they are taken from the respective default profile.

Local PPP User Profiles

Home menu level: ***/ppp profile***

Description

PPP profiles are used to define default values for user access records stored under **/ppp secret** submenu. Settings in **/ppp secret** User Database override corresponding **/ppp profile** settings except that single IP addresses always take precedence over IP pools when specified as **local-address** or **remote-address** parameters.

Property Description

bridge (*name*) - bridge interface name, which the PPP tunnel will automatically be added in case BCP negotiation will be successful (i.e., in case both peers support BCP and have this parameter configured)

change-tcp-mss (*yes* | *no* | *default*; default: **default**) - modifies TCP connection MSS settings

- **yes** - adjust connection MSS value
- **no** - do not atjust connection MSS value
- **default** - derive this value from the interface default profile; same as no if this is the interface default profile

dns-server (*IP address*) - IP address of the DNS server to supply to clients

idle-timeout (*time*) - specifies the amount of time after which the link will be terminated if there was no activity present. There is no timeout set by default

- **0s** - no link timeout is set

incoming-filter (*name*) - firewall chain name for incoming packets. Specified chain gets control for each packet coming from the client. The ppp chain should be manually added and rules with action=jump jump-target=ppp should be added to other relevant chains in order for this feature to work. For more information look at the Examples section

local-address (*IP addressname*) - IP address or IP address pool name for PPP server

name (*name*) - PPP profile name

only-one (*yes* | *no* | *default*; default: **default**) - defines whether a user is allowed to have more then one connection at a time

- **yes** - a user is not allowed to have more than one connection at a time

- **no** - the user is allowed to have more than one connection at a time
- **default** - derive this value from the interface default profile; same as no if this is the interface default profile

outgoing-filter (*name*) - firewall chain name for outgoing packets. Specified chain gets control for each packet going to the client. The ppp chain should be manually added and rules with action=jump jump-target=ppp should be added to other relevant chains in order for this feature to work. For more information look at the Examples section

rate-limit (*text*; default: **""**) - rate limitation in form of rx-rate[/tx-rate] [rx-burst-rate[/tx-burst-rate] [rx-burst-threshold[/tx-burst-threshold] [rx-burst-time[/tx-burst-time] [priority] [rx-rate-min[/tx-rate-min]]]] from the point of view of the router (so "rx" is client upload, and "tx" is client download). All rates are measured in bits per second, unless followed by optional 'k' suffix (kilobits per second) or 'M' suffix (megabits per second). If tx-rate is not specified, rx-rate serves as tx-rate too. The same applies for tx-burst-rate, tx-burst-threshold and tx-burst-time. If both rx-burst-threshold and tx-burst-threshold are not specified (but burst-rate is specified), rx-rate and tx-rate are used as burst thresholds. If both rx-burst-time and tx-burst-time are not specified, 1s is used as default. Priority takes values 1..8, where 1 implies the highest priority, but 8 - the lowest. If rx-rate-min and tx-rate-min are not specified rx-rate and tx-rate values are used. The rx-rate-min and tx-rate-min values can not exceed rx-rate and tx-rate values.

remote-address (*IP addressname*) - IP address or IP address pool name for PPP clients

session-timeout (*time*) - maximum time the connection can stay up. By default no time limit is set

- **0s** - no connection timeout

use-compression (*yes* | *no* | *default*; default: **default**) - specifies whether to use data compression or not

- **yes** - enable data compression
- **no** - disable data compression
- **default** - derive this value from the interface default profile; same as no if this is the interface default profile

use-encryption (*yes* | *no* | *required* | *default*; default: **default**) - specifies whether to use data encryption or not

- **yes** - enable data encryption
- **no** - disable data encryption
- **requided** - enable and require encryption
- **default** - derive this value from the interface default profile; same as no if this is the interface default profile

use-vj-compression (*yes* | *no* | *default*; default: **default**) - specifies whether to use Van Jacobson header compression algorithm

- **yes** - enable Van Jacobson header compression
- **no** - disable Van Jacobson header compression
- **default** - derive this value from the interface default profile; same as no if this is the interface default profile

wins-server (*IP address*) - IP address of the WINS server to supply to Windows clients

Notes

There are two default profiles that cannot be removed:

```
[admin@rb13] ppp profile> print
Flags: * - default
 0 * name="default" use-compression=default use-vj-compression=default
     use-encryption=default only-one=default change-tcp-mss=yes

 1 * name="default-encryption" use-compression=default
     use-vj-compression=default use-encryption=yes only-one=default
     change-tcp-mss=yes
[admin@rb13] ppp profile>
```

Use Van Jacobson compression only if you have to because it may slow down the communications on bad or congested channels.

incoming-filter and **outgoing-filter** arguments add dynamic **jump** rules to chain **ppp**, where the **jump-target** argument will be equal to **incoming-filter** or **outgoing-filter** argument in **/ppp profile**. Therefore, chain **ppp** should be manually added before changing these arguments.

only-one parameter is ignored if RADIUS authentication is used.

If there are more that 10 simultaneous PPP connections planned, it is recommended to turn the **change-mss** property off, and use one general MSS changing rule in mangle table instead, to reduce CPU utilization.

By configuring **bridge** property you enable the BCP protocol on the link. It is useful to enable MRRU as well in order for the link to be capable of transmitting full-size Ethernet frames. If the BCP negotiation is successful, the link will automatically be added to the specified bridge. Note that the bridge must have either a valid administrative MAC address, or another Ethernet-like port with a valid MAC address, as the PPP link do not have any MAC address.

Client will use a fake IP address (10.112.112.x) as a remote end address if no remote address is known. It won't be possible to ping this address, and it should be used as a gateway for routes (like default route) only. This helps GSM/GPRS setups where dial-in servers for some reason does not advertise their ip address.

Example

To add the profile **ex** that assigns the router itself the **10.0.0.1** address, and the addresses from the **ex** pool to the clients, filtering traffic coming from clients through **mypppclients** chain:

```
[admin@rb13] ppp profile> add name=ex local-address=10.0.0.1 remote-address=ex
incoming-filter=mypppclients
[admin@rb13] ppp profile> print
Flags: * - default
 0 * name="default" use-compression=default use-vj-compression=default
     use-encryption=default only-one=default change-tcp-mss=yes

 1 * name="default-encryption" use-compression=default
     use-vj-compression=default use-encryption=yes only-one=default
     change-tcp-mss=yes
 2   name="ex" local-address=10.0.0.1 remote-address=ex use-compression=default
     use-vj-compression=default use-encryption=default only-one=default
     change-tcp-mss=default incoming-filter=mypppclients
[admin@rb13] ppp profile>
```

Local PPP User Database

Home menu level: ***/ppp secret***

Description

PPP User Database stores PPP user access records with PPP user profile assigned to each user.

Property Description

caller-id (*text*; default: **""**) - for PPTP and L2TP it is the IP address a client must connect from. For PPPoE it is the MAC address (written in CAPITAL letters) a client must connect from. For ISDN it is the caller's number (that may or may not be provided by the operator) the client may dial-in from

- **""** - no restrictions on where clients may connect from

limit-bytes-in (*integer*; default: **0**) - maximal amount a client can upload, in bytes, for a session

limit-bytes-out (*integer*; default: **0**) - maximal amount a client can download, in bytes, for a session

local-address (*IP addressname*) - IP address or IP address pool name for PPP server

name (*name*) - user's name used for authentication

password (*text*; default: **""**) - user's password used for authentication

profile (*name*; default: **default**) - profile name to use together with this access record for user authentication

remote-address (*IP addressname*) - IP address or IP address pool name for PPP clients

routes (*text*) - routes that appear on the server when the client is connected. The route format is: dst-address [[gateway] [metric]] (for example, 10.1.0.0/24 10.0.0.1 1). Several routes may be specified separated with commas. If gateway is not specified, the remote address is used. If metric is not speciefied, the metric of 1 is used

service (*any* | *async* | *l2tp* | *ovpn* | *pppoe* | *pptp*; default: **any**) - specifies the services available to a particular user

Example

To add the user **ex** with password **lkjrht** and profile **ex** available for PPTP service only, enter the following command:

```
[admin@rb13] ppp secret> add name=ex password=lkjrht service=pptp profile=ex
[admin@rb13] ppp secret> print
Flags: X - disabled
 #   NAME                   SERVICE CALLER-ID PASSWORD PROFILE REMOTE-ADDRESS
 0   ex                     pptp              lkjrht   ex      0.0.0.0
[admin@rb13] ppp secret>
```

Monitoring Active PPP Users

Command name: ***/ppp active print***

Property Description

address (*read-only: IP address*) - IP address the client got from the server

bytes (*read-only: integerinteger*) - amount of bytes transfered through this connection. First figure represents amount of transmitted traffic from the router's point of view, while the second one shows amount of received traffic

caller-id (*read-only: text*) - for PPTP and L2TP it is the IP address the client connected from. For PPPoE it is the MAC address the client connected from. For ISDN it is the caller's number the client dialed-in from

- **""** - no restrictions on where clients may connect from

encoding (*read-only: text*) - shows encryption and encoding (separated with '/' if asymmetric) being used in this connection

limit-bytes-in (*read-only: integer*) - maximal amount of bytes the user is allowed to send to the router

limit-bytes-out (*read-only: integer*) - maximal amount of bytes the router is allowed to send to the client

name (*read-only: name*) - user name supplied at authentication stage

packets (*read-only: integerinteger*) - amount of packets transfered through tis connection. First figure represents amount of transmitted traffic from the router's point of view, while the second one shows amount of received traffic

service (*read-only: async* | *l2tp* | *ovpn* | *pppoe* | *pptp*) - the type of service the user is using

session-id (*read-only: text*) - shows unique client identifier

uptime (*read-only: time*) - user's uptime

Example

```
[admin@rb13] > /ppp active print
Flags: R - radius
 #   NAME         SERVICE CALLER-ID         ADDRESS         UPTIME   ENCODING
 0   ex           pptp    10.0.11.12        10.0.0.254      1m16s    MPPE128...
[admin@rb13] > /ppp active print detail
Flags: R - radius
 0   name="ex" service=pptp caller-id="10.0.11.12" address=10.0.0.254
     uptime=1m22s encoding="MPPE128 stateless" session-id=0x8180002B
     limit-bytes-in=200000000 limit-bytes-out=0
[admin@rb13] > /ppp active print stats
Flags: R - radius
 #   NAME         BYTES                  PACKETS
 0   ex           10510/159690614        187/210257
[admin@rb13] >
```

PPP User Remote AAA

Home menu level: ***/ppp aaa***

Property Description

accounting (yes | no; default: **yes**) - enable RADIUS accounting

interim-update (*time*; default: **0s**) - Interim-Update time interval

use-radius (yes | no; default: **no**) - enable user authentication via RADIUS

Notes

RADIUS user database is consulted only if the required username is not found in local user database.

Example

To enable RADIUS AAA:

```
[admin@MikroTik] ppp aaa> set use-radius=yes
[admin@MikroTik] ppp aaa> print
        use-radius: yes
        accounting: yes
    interim-update: 0s
[admin@MikroTik] ppp aaa>
```

Router User AAA

Document revision 2.4 (February 6, 2008, 1:40 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

This documents provides summary, configuration reference and examples on router user management.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/user***
Hardware usage: ***Not significant***

Description

MikroTik RouterOS router user facility manage the users connecting the router from the local console, via serial terminal, telnet, SSH or Winbox. The users are authenticated using either local database or designated RADIUS server.

Each user is assigned to a user group, which denotes the rights of this user. A group policy is a combination of individual policy items.

In case the user authentication is performed using RADIUS, the RADIUS client should be previously configured under the **/radius** submenu.

Router User Groups

Home menu level: ***/user group***

Description

The router user groups provide a convenient way to assign different permissions and access rights to different user classes.

Property Description

name (*name*) - the name of the user group

policy (*multiple choice: local* | *telnet* | *ssh* | *ftp* | *reboot* | *read* | *write* | *policy* | *test* | *winbox* | *password* | *web* | *sniff*) - group policy item set

- **local** - policy that grants rights to log in locally via local console
- **telnet** - policy that grants rights to log in remotely via telnet
- **ssh** - policy that grants rights to log in remotely via secure shell protocol
- **ftp** - policy that grants remote rights to log in remotely via FTP and to transfer files from and to the router. Keep in mind that the user allowed to transfer files, may also upload a new RouterOS version that will be applied upon the next reboot
- **reboot** - policy that allows rebooting the router
- **read** - policy that grants read access to the router's configuration. All console commands that do not alter router's configuration are allowed
- **write** - policy that grants write access to the router's configuration, except for user management. This policy does not allow to read the configuration, so make sure to enable read policy as well
- **policy** - policy that grants user management rights. Should be used together with write policy
- **test** - policy that grants rights to run ping, traceroute, bandwidth-test and wireless scan, sniffer and snooper commands
- **winbox** - policy that grants rights to connect to the router remotely using WinBox interface
- **password** - policy that grants user option to change own password
- **web** - policy that grants rights to log in remotely via WebBox
- **sniff** - policy that grants access to the packet sniffer facility

Notes

There are three system groups which cannot be deleted:

```
[admin@rb13] > /user group print
 0 name="read" policy=local,telnet,ssh,reboot,read,test,winbox,password,web,
                    sniff,!ftp,!write,!policy

 1 name="write" policy=local,telnet,ssh,reboot,read,write,test,winbox,password,
                     web,sniff,!ftp,!policy

 2 name="full" policy=local,telnet,ssh,ftp,reboot,read,write,policy,test,winbox,
                    password,web,sniff
[admin@rb13] >
```

Exclamation sign '**!**' just before policy item name means **NOT**.

Example

To add **reboot** group that is allowed to reboot the router locally or using telnet, as well as read the router's configuration, enter the following command:

```
[admin@rb13] user group> add name=reboot policy=telnet,reboot,read,local
[admin@rb13] user group> print
 0 name="read" policy=local,telnet,ssh,reboot,read,test,winbox,password,web,
                    sniff,!ftp,!write,!policy

 1 name="write" policy=local,telnet,ssh,reboot,read,write,test,winbox,password,
                     web,sniff,!ftp,!policy

 2 name="full" policy=local,telnet,ssh,ftp,reboot,read,write,policy,test,winbox,
                    password,web,sniff
 3 name="reboot" policy=local,telnet,reboot,read,!ssh,!ftp,!write,!policy,!test,
                      !winbox,!password,!web,!sniff
[admin@rb13] user group>
```

Router Users

Home menu level: ***/user***

Description

Router user database stores the information such as username, password, allowed access addresses and group about router management personnel.

Property Description

address (*IP addressnetmask*; default: **0.0.0.0/0**) - host or network address from which the user is allowed to log in

group (*name*) - name of the group the user belongs to

name (*name*) - user name. Although it must start with an alphanumeric character, it may contain "*", "_", "." and "@" symbols

password (*text*; default: **""**) - user password. If not specified, it is left blank (hit [Enter] when logging in). It conforms to standard Unix characteristics of passwords and may contain letters, digits, "*" and "_" symbols

Notes

There is one predefined user with full access rights:

```
[admin@MikroTik] user> print
Flags: X - disabled
  #   NAME                                             GROUP ADDRESS
  0   ;;; system default user
      admin                                            full  0.0.0.0/0

[admin@MikroTik] user>
```

There always should be at least one user with full access rights. If the user with full access rights is the only one, it cannot be removed.

Example

To add user **joe** with password **j1o2e3** belonging to **write** group, enter the following command:

```
[admin@MikroTik] user> add name=joe password=j1o2e3 group=write
[admin@MikroTik] user> print
Flags: X - disabled
  0   ;;; system default user
      name="admin" group=full address=0.0.0.0/0

  1   name="joe" group=write address=0.0.0.0/0


[admin@MikroTik] user>
```

Monitoring Active Router Users

Command name: ***/user active print***

Description

This command shows the currently active users along with respective statisics information.

Property Description

address (*read-only: IP address*) - host IP address from which the user is accessing the router

- **0.0.0.0** - the user is logged in locally from the console

name (*read-only: name*) - user name

radius (*read-only: flag*) - the user has been authenticated through a RADIUS server

via (*read-only: console* | *telnet* | *ssh* | *winbox*) - user's access method

- **console** - user is logged in locally
- **telnet** - user is logged in remotely via telnet
- **ssh** - user is logged in remotely via secure shell protocol
- **winbox** - user is logged in remotely via WinBox tool

when (*read-only: date*) - log in date and time

Example

To print currently active users, enter the following command:

```
[admin@rb13] user> active print
Flags: R - radius
 #   WHEN                 NAME                                               ADDRESS
VIA
 0   feb/27/2004 00:41:41 admin                                              1.1.1.200
ssh
 1   feb/27/2004 01:22:34 admin                                              1.1.1.200
winbox
[admin@rb13] user>
```

Router User Remote AAA

Home menu level: ***/user aaa***

Description

Router user remote AAA enables router user authentication and accounting via RADIUS server.

Property Description

accounting (yes | no; default: **yes**) - whether to use RADIUS accounting

default-group (*name*; default: **read**) - user group used for the users authenticated via a RADIUS server by default (if the server did not specify a different user group)

interim-update (*time*; default: **0s**) - RADIUS Interim-Update interval

use-radius (yes | no; default: **no**) - specifies whether a user database on a RADIUS server should be consulted

Notes

The RADIUS user database is consulted only if the required username is not found in the local user database

Example

To enable RADIUS AAA, enter the following command:

```
[admin@MikroTik] user aaa> set use-radius=yes
[admin@MikroTik] user aaa> print
        use-radius: yes
        accounting: yes
    interim-update: 0s
     default-group: read
[admin@MikroTik] user aaa>
```

SSH keys

Home menu level: ***/user ssh-keys***

Description

Remote users may be allowed to log in without using password authentication and even ever entering their password, but by using pregenerated DSA openssh SSH keys instead. Note that if you use puttygen, convert generated keys to right type.

Property Description

key-owner (*read-only: text*) - emote user, as specified in the key file

user (*name*) - the user that is allowed to log in using this key (must exist in the user list)

Command Description

import - import the uploaded DSA key

- **user** - the user the imported key is linked to
- **file** - filename of the DSA key to import

Example

Generating key on a linux machine:

```
sh-3.00$ ssh-keygen -t dsa -f ./id_dsa
Generating public/private dsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ./id_dsa.
Your public key has been saved in ./id_dsa.pub.
The key fingerprint is:
91:d7:08:be:b6:a1:67:5e:81:02:cb:4d:47:d6:a0:3b admin-ssh@test
```

Importing the generated (ang uploaded) key:

```
[admin@MikroTik] user ssh-keys> print
 # USER                 KEY-OWNER
[admin@MikroTik] user ssh-keys> import file=id_dsa.pub user=admin-ssh
[admin@MikroTik] user ssh-keys> print
 # USER                 KEY-OWNER
 0 admin-ssh            admin-ssh@test
[admin@MikroTik] user ssh-keys>
```

Traffic Flow

Document revision 1.1 (February 6, 2008, 1:40 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip traffic-flow***
Hardware usage: ***Not significant***

Related Documents

- Cisco NetFlow
- NTop
- Integrating ntop with NetFlow

Description

MikroTik Traffic-Flow is a system that provides statistic information about packets which pass through the router. Besides network monitoring and accounting, system administrators can identify various problems that may occur in the network. With help of Traffic-Flow, it is possible to analyze and optimize the overall network performance. As Traffic-Flow is compatible with Cisco NetFlow, it can be used with various utilities which are designed for Cisco's NetFlow.

Traffic-Flow supports the following NetFlow formats:

- **version 1** - the first version of NetFlow data format, do not use it, unless you have to
- **version 5** - in addition to version 1, version 5 has the BGP AS and flow sequence number

information included

- **version 9** - a new format which can be extended with new fields and record types, thanks to its template-style design

General Configuration

Description

This section describes the basic configuration of Traffic-Flow.

Property Description

active-flow-timeout (*time*; default: **30m**) - maximum life-time of a flow

cache-entries (*1k* | *2k* | *4k* | *8k* | *16k* | *32k* | *64k* | *128k* | *256k* | *512k*; default: **1k**) - number of flows which can reside in the router's memory simultaneously

enabled (yes | no) - whether to enable traffic-flow service or not

inactive-flow-timeout (*time*; default: **15s**) - how long to keep the flow active, if it is idle

interfaces (*name*) - names of those interfaces which will be used to gather statistics for traffic-flow. To specify more than one interface, separate them with a comma (",")

Traffic-Flow Target

Home menu level: ***/ip traffic-flow target***

Description

With Traffic-Flow targets we specify those hosts which will gather the Traffic-Flow information from router.

Property Description

address (*IP addressport*) - IP address and UDP port of the host which receives Traffic-Flow statistics packets from the router

v9-template-refresh (*integer*; default: **20**) - number of packets after which the template is sent to the receiving host (only for NetFlow version 9)

v9-template-timeout - after how long to send the template, if it has not been sent

version (*1* | *5* | *9*) - which version format of NetFlow to use

Application Examples

Traffic-Flow Example

This example shows how to configure Traffic-Flow on a router

1. Enable Traffic-Flow on the router:

```
[admin@MikroTik] ip traffic-flow> set enabled=yes
```

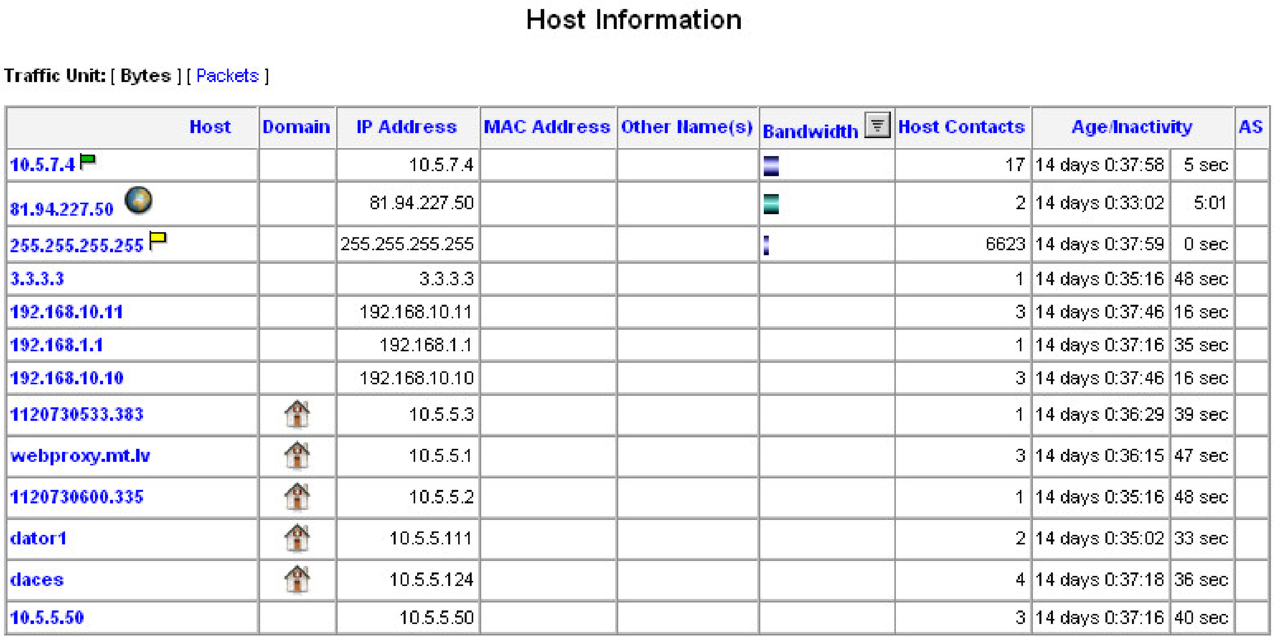
```
[admin@MikroTik] ip traffic-flow> print
                enabled: yes
             interfaces: all
          cache-entries: 1k
    active-flow-timeout: 30m
  inactive-flow-timeout: 15s
[admin@MikroTik] ip traffic-flow>
```

2. Specify IP address and port of the host, which will receive Traffic-Flow packets:

```
[admin@MikroTik] ip traffic-flow target> add address=192.168.0.2:2055 \
\... version=9
[admin@MikroTik] ip traffic-flow target> print
Flags: X - disabled
 #   ADDRESS                VERSION
 0   192.168.0.2:2055       9
[admin@MikroTik] ip traffic-flow target>
```

Now the router starts to send packets with Traffic-Flow information.

Some screenshots from NTop program, which has gathered Traffic-Flow information from our router and displays it in nice graphs and statistics. For example, where what kind of traffic has flown:

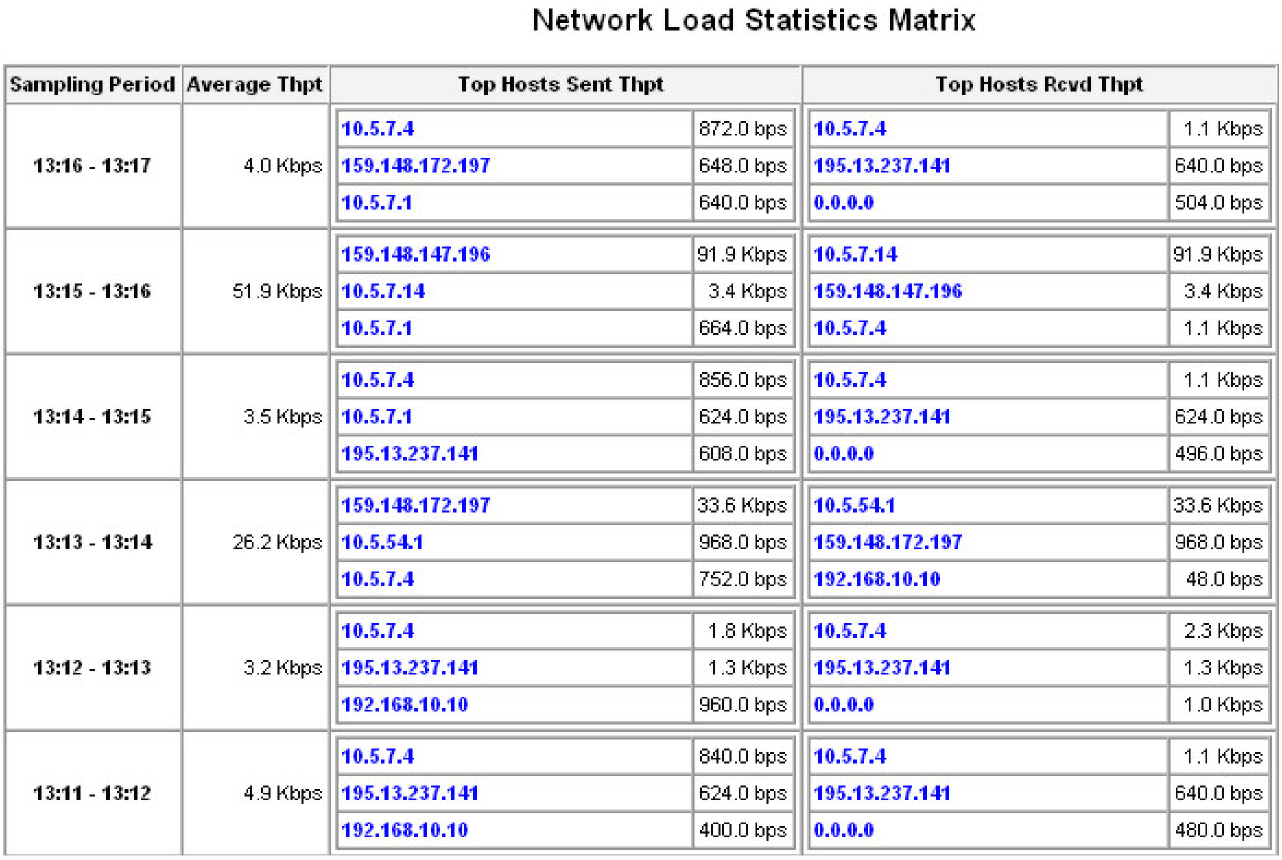


Host Information

Traffic Unit: [Bytes] [Packets]

Host	Domain	IP Address	MAC Address	Other Name(s)	Bandwidth	Host Contacts	Age/Inactivity		AS
10.5.7.4 ⚑		10.5.7.4			▮	17	14 days 0:37:58	5 sec	
81.94.227.50		81.94.227.50			▮	2	14 days 0:33:02	5:01	
255.255.255.255 ⚐		255.255.255.255			▮	6623	14 days 0:37:59	0 sec	
3.3.3.3		3.3.3.3				1	14 days 0:35:16	48 sec	
192.168.10.11		192.168.10.11				3	14 days 0:37:46	16 sec	
192.168.1.1		192.168.1.1				1	14 days 0:37:16	35 sec	
192.168.10.10		192.168.10.10				3	14 days 0:37:46	16 sec	
1120730533.383	🏠	10.5.5.3				1	14 days 0:36:29	39 sec	
webproxy.mt.lv	🏠	10.5.5.1				3	14 days 0:36:15	47 sec	
1120730600.335	🏠	10.5.5.2				1	14 days 0:35:16	48 sec	
dator1	🏠	10.5.5.111				2	14 days 0:35:02	33 sec	
daces	🏠	10.5.5.124				4	14 days 0:37:18	36 sec	
10.5.5.50		10.5.5.50				3	14 days 0:37:16	40 sec	

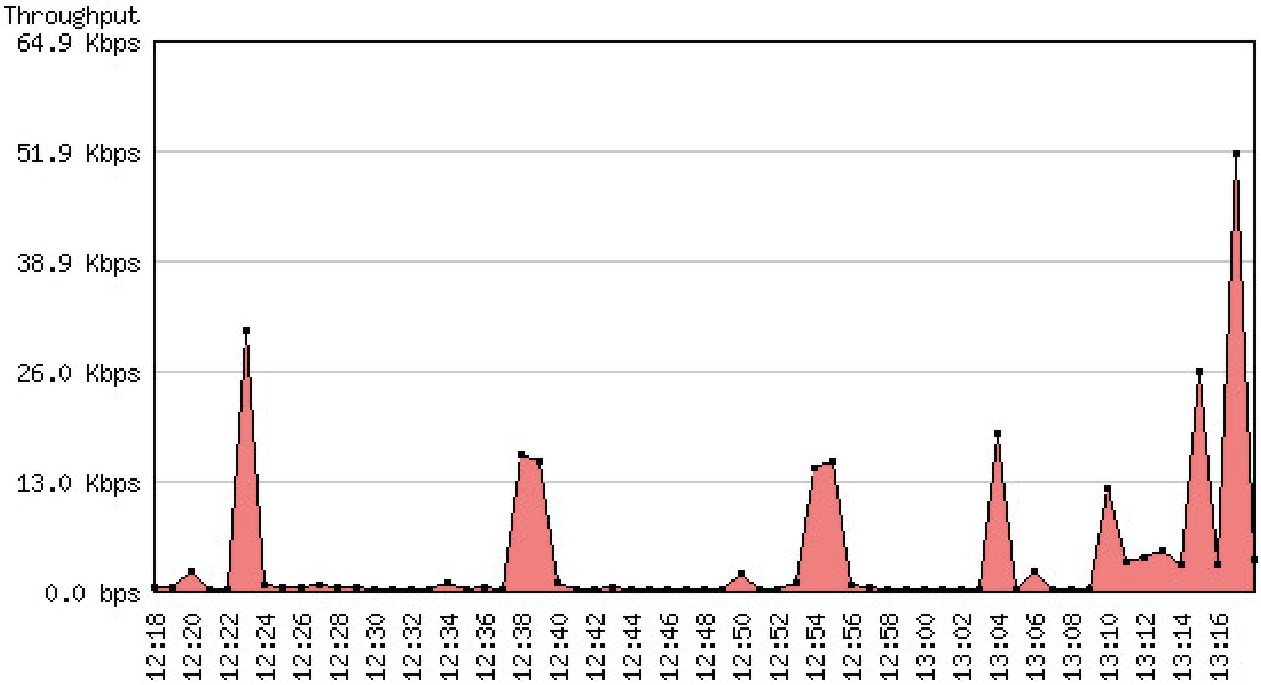
Top three hosts by upload and download each minute:



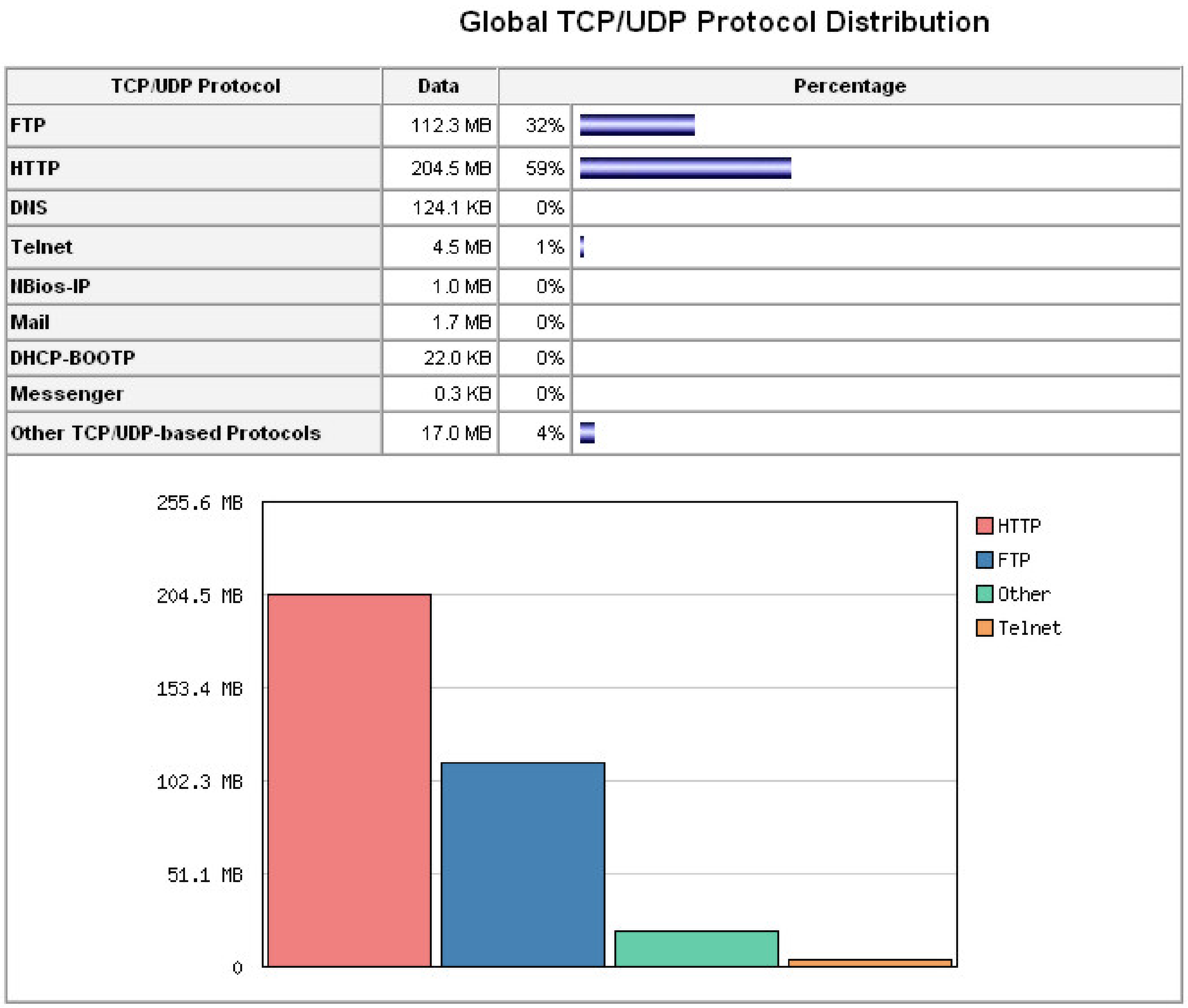
Network Load Statistics Matrix

Sampling Period	Average Thpt	Top Hosts Sent Thpt		Top Hosts Rcvd Thpt	
13:16 - 13:17	4.0 Kbps	10.5.7.4	872.0 bps	10.5.7.4	1.1 Kbps
		159.148.172.197	648.0 bps	195.13.237.141	640.0 bps
		10.5.7.1	640.0 bps	0.0.0.0	504.0 bps
13:15 - 13:16	51.9 Kbps	159.148.147.196	91.9 Kbps	10.5.7.14	91.9 Kbps
		10.5.7.14	3.4 Kbps	159.148.147.196	3.4 Kbps
		10.5.7.1	664.0 bps	10.5.7.4	1.1 Kbps
13:14 - 13:15	3.5 Kbps	10.5.7.4	856.0 bps	10.5.7.4	1.1 Kbps
		10.5.7.1	624.0 bps	195.13.237.141	624.0 bps
		195.13.237.141	608.0 bps	0.0.0.0	496.0 bps
13:13 - 13:14	26.2 Kbps	159.148.172.197	33.6 Kbps	10.5.54.1	33.6 Kbps
		10.5.54.1	968.0 bps	159.148.172.197	968.0 bps
		10.5.7.4	752.0 bps	192.168.10.10	48.0 bps
13:12 - 13:13	3.2 Kbps	10.5.7.4	1.8 Kbps	10.5.7.4	2.3 Kbps
		195.13.237.141	1.3 Kbps	195.13.237.141	1.3 Kbps
		192.168.10.10	960.0 bps	0.0.0.0	1.0 Kbps
13:11 - 13:12	4.9 Kbps	10.5.7.4	840.0 bps	10.5.7.4	1.1 Kbps
		195.13.237.141	624.0 bps	195.13.237.141	640.0 bps
		192.168.10.10	400.0 bps	0.0.0.0	480.0 bps

Overall network load each minute:



Traffic usage by each protocol:



Global TCP/UDP Protocol Distribution

TCP/UDP Protocol	Data	Percentage	
FTP	112.3 MB	32%	
HTTP	204.5 MB	59%	
DNS	124.1 KB	0%	
Telnet	4.5 MB	1%	
NBios-IP	1.0 MB	0%	
Mail	1.7 MB	0%	
DHCP-BOOTP	22.0 KB	0%	
Messenger	0.3 KB	0%	
Other TCP/UDP-based Protocols	17.0 MB	4%	

Log Management

Document revision 2.4 (February 6, 2008, 1:40 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Various system events and status information can be logged. Logs can be saved in local routers file, displayed in console, sent to an email or to a remote server running a syslog daemon. MikroTik provides a shareware Windows Syslog daemon, which can be downloaded from www.mikrotik.com

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/system logging, /log***
Standards and Technologies: ***Syslog***
Hardware usage: ***Not significant***

Description

Logs have different groups or topics. Logs from each topic can be configured to be discarded, logged locally or remotely. Locally log files can be stored in memory (default; logs are lost on reboot or power outage) or on hard drive (not enabled by default as is harmful for flash disks).

General Settings

Home menu level: ***/system logging***

Property Description

action (*name*; default: **memory**) - specifies one of the system default actions or user specified action listed in /system logging action

prefix (*text*) - local log prefix

topics (*info* | *critical* | *firewall* | *keepalive* | *packet* | *read* | *timer* | *write* | *ddns* | *hotspot* | *l2tp* | *ppp* | *route* | *update* | *account* | *debug* | *ike* | *manager* | *pppoe* | *script* | *warning* | *async* | *dhcp* | *notification* | *pptp* | *state* | *watchdog* | *bgp* | *error* | *ipsec* | *radius* | *system* | *web-proxy* | *calc* | *event* | *isdn* | *ospf* | *raw* | *telephony* | *wireless* | *e-mail* | *gsm* | *mme* | *ntp* | *open* | *ovpn* | *pim* | *radvd* | *rip* | *sertcp* | *ups*; default: **info**) - specifies log group or log message type

Example

To log messages that are generated by firewall by saving them in local buffer

```
[admin@MikroTik] system logging> add topics=firewall action=memory
[admin@MikroTik] system logging> print
 Flags: X - disabled, I - invalid
 #   TOPICS                                                   ACTION PREFIX
 0   info                                                     memory
 1   error                                                    memory
 2   warning                                                  memory
 3   critical                                                 echo
 4   firewall                                                 memory
[admin@MikroTik] system logging>
```

Actions

Home menu level: ***/system logging action***

Property Description

disk-lines (*integer*; default: **100**) - number of records in log file saved on the disk (only if action target is set to disk)

disk-stop-on-full (yes | no; default: **no**) - whether to stop to save log messages on disk after the specified disk-lines number is reached

email-to (*name*) - email address logs are sent to (only if action target is set to email)

memory-lines (*integer*; default: **100**) - number of records in local memory buffer (only if action target is set to memory)

memory-stop-on-full (yes | no; default: **no**) - whether to stop to save log messages in local buffer after the specified memory-lines number is reached

name (*name*) - name of an action

remember (yes | no; default: **yes**) - whether to keep log messages, which have not yet been displayed in console (only if action target is set to echo)

remote (*IP addressport*; default: **0.0.0.0:514**) - remote logging server's IP address and UDP port (only if action target is set to remote)

target (*disk* | *echo* | *email* | *memory* | *remote*; default: **memory**) - log storage facility or target

- **disk** - logs are saved to the hard drive
- **echo** - logs are displayed on the console screen

- **email** - logs are sent by email
- **memory** - logs are saved to the local memory buffer
- **remote** - logs are sent to a remote host

Notes

You cannot delete or rename default actions.

Example

To add a new action with name short, that will save logs in local buffer, if number of records in buffer are less than 50:

```
[admin@MikroTik] system logging action> add name=short \
\... target=memory memory-lines=50 memory-stop-on-full=yes
[admin@MikroTik] system logging action> print
Flags: * - default
 #   NAME                                TARGET REMOTE
 0 * memory                              memory
 1 * disk                                disk
 2 * echo                                echo
 3 * remote                              remote 0.0.0.0:514
 4   short                               memory
[admin@MikroTik] system logging action>
```

Log Messages

Home menu level: ***/log***

Description

Displays locally stored log messages

Property Description

message (*read-only: text*) - message text

time (*read-only: text*) - date and time of the event

topics (*read-only: text*) - topic list the message belongs to

Example

To view the local logs:

```
[admin@MikroTik] > log print
 TIME                 MESSAGE
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:20:36 log configuration changed by admin
-- [Q quit|D dump]
```

To monitor the system log:

```
[admin@MikroTik] > log print follow
 TIME                 MESSAGE
 dec/24/2003 08:20:36 log configuration changed by admin
 dec/24/2003 08:24:34 log configuration changed by admin
 dec/24/2003 08:24:51 log configuration changed by admin
 dec/24/2003 08:25:59 log configuration changed by admin
 dec/24/2003 08:25:59 log configuration changed by admin
 dec/24/2003 08:30:05 log configuration changed by admin
 dec/24/2003 08:30:05 log configuration changed by admin
 dec/24/2003 08:35:56 system started
 dec/24/2003 08:35:57 isdn-out1: initializing...
 dec/24/2003 08:35:57 isdn-out1: dialing...
 dec/24/2003 08:35:58 Prism firmware loading: OK
 dec/24/2003 08:37:48 user admin logged in from 10.1.0.60 via telnet
-- Ctrl-C to quit. New entries will appear at bottom.
```

Bandwidth Control

Document revision 2.2 (November 28, 2007, 10:45 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Bandwidth Control is a set of mechanisms that control data rate allocation, delay variability, timely delivery, and delivery reliability. The MikroTik RouterOS supports the following queuing disciplines:

- **PFIFO** - Packets First-In First-Out
- **BFIFO** - Bytes First-In First-Out
- **SFQ** - Stochastic Fairness Queuing
- **RED** - Random Early Detect
- **PCQ** - Per Connection Queue
- **HTB** - Hierarchical Token Bucket

Specifications

Packages required: ***system***
License required: ***level1 (limited to 1 queue), level3***
Home menu level: ***/queue***
Standards and Technologies: ***None***
Hardware usage: *significant*

Description

Quality of Service (QoS) means that the router should prioritize and shape network traffic. QoS is not so much about limiting, it is more about providing quality service to the network users. Some features of MikroTik RouterOS traffic control mechanism are listed below:

- limit data rate for certain IP adresses, subnets, protocols, ports, and other parameters
- limit peer-to-peer traffic
- prioritize some packet flows over others
- use queue bursts for faster web browsing
- apply queues on fixed time intervals
- share available traffic among users equally, or depending on the load of the channel

The queuing is applied on packets leaving the router through a real interface (i.e., the queues are applied on the outgoing interface, regarding the traffic flow), or any of the 3 additional virtual interfaces (global-in, global-out, global-total).

The QoS is performed by means of dropping packets. In case of TCP protocol, the dropped packets will be resent on a slower rate, so there is no need to worry that with shaping we lose some TCP information.

The main terms used to describe the level of QoS for network applications, are:

- **queuing discipline (qdisc)** - an algorithm that holds and maintains a queue of packets. It accumulates the packets and decides the order of the outgoing packets (it means that queuing discipline can reorder packets). Qdisc also decides which packets to drop if there is no space for them.
- **CIR (Committed Information Rate)** - the guaranteed data rate. It means that traffic rate, not exceeding this value should always be delivered
- **MIR (Maximal Information Rate)** - the maximal data rate router will provide
- **Priority** - the order of importance in what traffic will be processed. You can give priority to some traffic in order it to be handeled before some other traffic
- **Contention Ratio** - the ratio to which the defined data rate is shared among users (when a certain data rate is allocated to a number of subscribers). It is the number of subscribers that have a single speed limitation, applied to all of them together. For example, the contention ratio of 1:4 means that the allocated data rate may be shared between no more than 4 users

Before sending data over an interface, it is processed with a queuing discipline. There can be only one queueing discipline per interface, which, by default, is set under **/queue interface** for each physical interface (there is no default queuing discipline for virtual interfaces). Once we add a first queue (in **/queue tree** or **/queue simple**) to a physical interface, the interface default queue is replaced by HTB hierarchy with that queue, but the one defined in **/queue interface** for that particular interface, is no more active.

Scheduler and Shaper qdiscs

We can classify queuing disciplines by their influence to packet flow:

- **schedulers** - queuing disciplines only reschedule packets regarding their algorithm and drop packets which 'do not fit in the queue'. Scheduler queuing disciplines are: PFIFO, BFIFO, SFQ, PCQ (both scheduler and shaper), RED
- **shapers** - queuing disciplines that also perform the limitation. Shapers are PCQ (both scheduler and shaper) and HTB

Virtual Interfaces

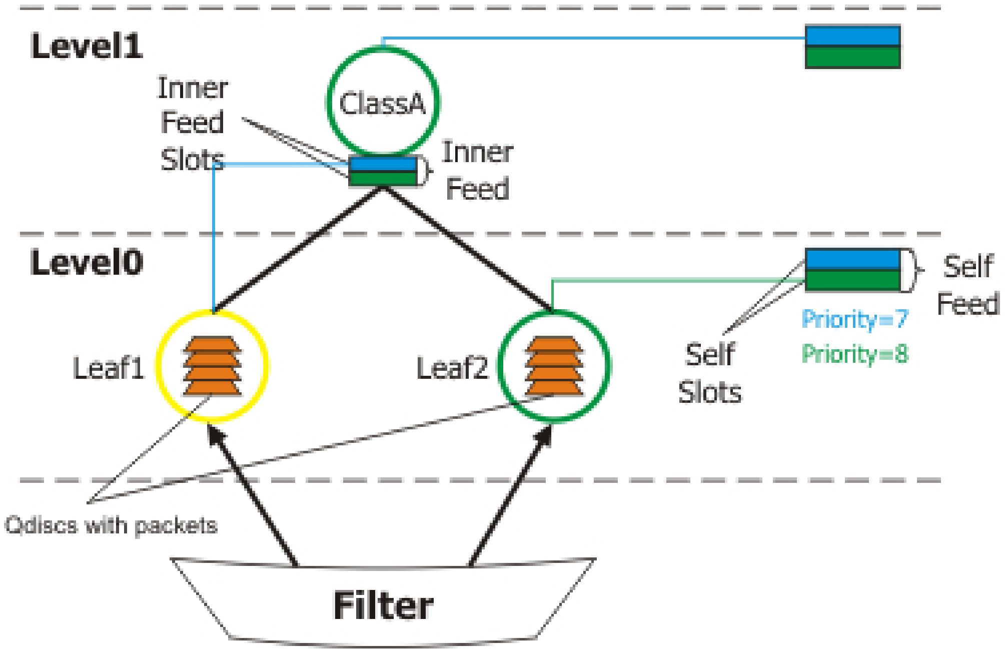
There are 3 virtual interfaces in RouterOS, in addition to real interfaces:

- **global-in** - represents all the input interfaces in general (INGRESS queue). Please note that queues attached to global-in apply to traffic that is received by the router, before the packet filtering. global-in queueing is executed just after mangle and dst-nat
- **global-out** - represents all the output interfaces in general (EGRESS queue). Queues attached to it apply before the ones attached to a specific interface
- **global-total** - represents a virtual interface through which all the data, going through the router, is passing. When attaching a qdisc to global-total, the limitation is done in both directions. For example, if we set a total-max-limit to 256000, we will get upload+download=256kbps (maximum)

Introduction to HTB

HTB (Hierarchical Token Bucket) is a classful queuing discipline that is useful for applying different handling for different kinds of traffic. The queues you add in **/queue simple** and **/queue tree** are attached to the main Hierarchical Token Bucket (HTB). For example, you can set a maximum data rate for a workgroup and then distribute that amount of traffic between the members of that workgroup.

HTB qdisc in detail:



HTB terms:

- **filter** - a procedure that classifies packets. The filters are responsible for classifying packets so that they are put in the corresponding qdiscs. All filters are applied at the HTB root and classify packets directly into the qdiscs, without traversing the HTB tree. If a packet is not classified into any of the qdiscs, it is sent out to the interface directly, traversing the HTB, so no HTB rules are applied to those packets (it would mean effective higher priority than of any packet flow managed by HTB).
- **level** - position of a class in the hierarchy.
- **class** - algorithm for limiting traffic flow to a certain rate. It does not store any packets (this function can only be performed by a queue). A class may contain either one or more subclasses (inner class), or one and only one qdisc (leaf class).
- **inner class** - a class that has one or more child class attached to it. As inner classes do not store any packets, qdiscs can not be attached to them (so their qdisc and filter settings are ignored, although may be still shown in RouterOS configuration), so they only do traffic shaping. Priority setting is ignored as well.
- **leaf class** - a class that has a parent but does not have any child classes. Leaf classes are always located at level 0 of the hierarchy. Each leaf class has one and only one qdisc attached to it, with a certain priority.
- **self feed** - an exit (out of the HTB tree, to the interface) for the packets from all the classes active on its level of the hierarchy. There is one self feed per level, each consisting of 8 self slots that represent priorities.
- **self slot** - an element of a self feed that corresponds to each particular priority. There is one self slot per priority per level. All classes, active at the same level, having the same priority are attached to one self slot that they are using to send packets out through.
- **active class (at a particular level)** - a class that is attached to a self slot at the given level.
- **inner feed** - similar to a self feed object, which consists of inner self slots, present on each inner class. There is one inner feed per inner class.

- **inner feed slot** - similar to self slot. Each inner feed consists of inner slots which represent a priority.

Each class has a parent and may have one or more children. Classes that do not have children, are put at level 0, where queues are maintained, and are called 'leaf classes'.

Each class in the hierarchy can prioritize and shape traffic. There are two main parameters, which refer to shaping and one - to prioritizing:

- **limit-at** - normal data rate that is guaranteed to a class (CIR)
- **max-limit** - maximal data rate that is allowed for a class to reach (MIR)
- **priority** - order in which classes are served at the same level (8 is the lowest priority, 1 is the highest)

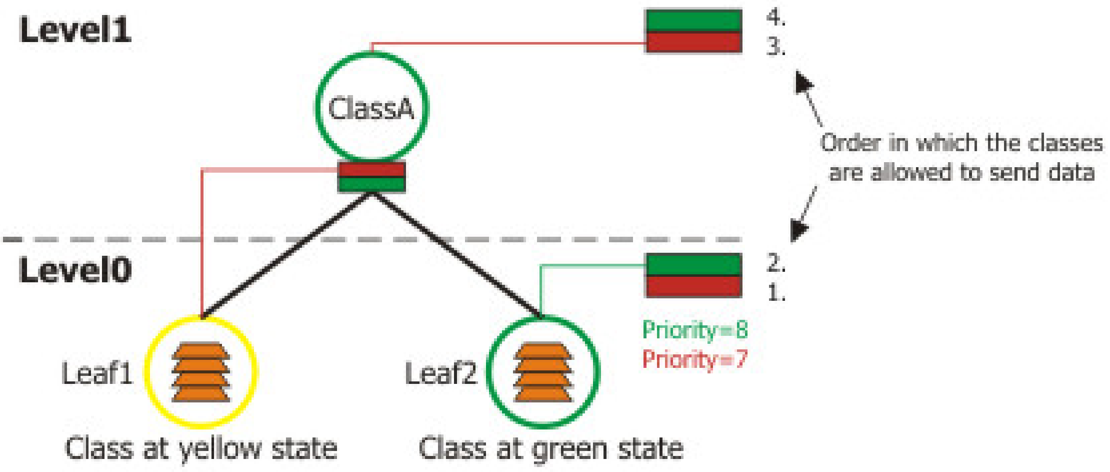
Each HTB class can be in one of 3 states, depending on data rate that it consumes:

- **green** - a class the actual rate of which is equal or less than limit-at. At this state, the class is attached to self slot at the corresponding priority at its level, and is allowed to satisfy its CIR limitation regardless of what limitations its parents have. For example, if we have a leaf class with limit-at=512000 and its parent has max-limit=limit-at=128000, the class will still get its 512kbps! All CIRs of a particular level are satisfied before all MIRs of the same level and any limitations of higher levels.
- **yellow** - a class the actual rate of which is greater than limit-at and equal or less than max-limit (or burst-limit if burst is active). At this state, the class is attached to the inner slot of the corresponding priority of its parent's inner feed, which, in turn, may be attached to either its parent's inner slot of the same priority (in case the parent is also yellow), or to its own level self slot of the same priority (in case the parent is green). Upon the transition to this state, the class 'disconnects' from self feed of its level, and 'connects' to its parent's inner feed.
- **red** - a class the actual rate of which exceeds max-limit (or burst-limit if burst is active). This class cannot borrow rate from its parent class.

Note: as CIRs are always satisfied before MIRs or other limitations of higher levels are consulted, you should always ensure that the **limit-at** property of any inner class is equal or greater than the sum of all **limit-at** parameter of the children of that inner class.

Priorities

When there is a possibility to send out a packet, HTB queries all its self slots in order of priority, starting with highest priority on the lowest level, till lowest priority on highest level. Each leaf class (packets are stored and enqueued only within qdiscs attached to each leaf class) is ultimately connected to a certain self slot, either directly or through a chain of parent classes:



As you can see from the picture, leaf-classes that are in the green state will always have a higher effective priority than those that are yellow (and, thus, borrowing their rate from parent classes), because their priority is at a lower level (level 0). In this picture, **Leaf1** will be served only after **Leaf2**, although it has a higher priority (priority 7) than **Leaf1** (priority 8).

In case of equal priorities and equal states, HTB serves these classes, using round robin algorithm.

HTB Examples

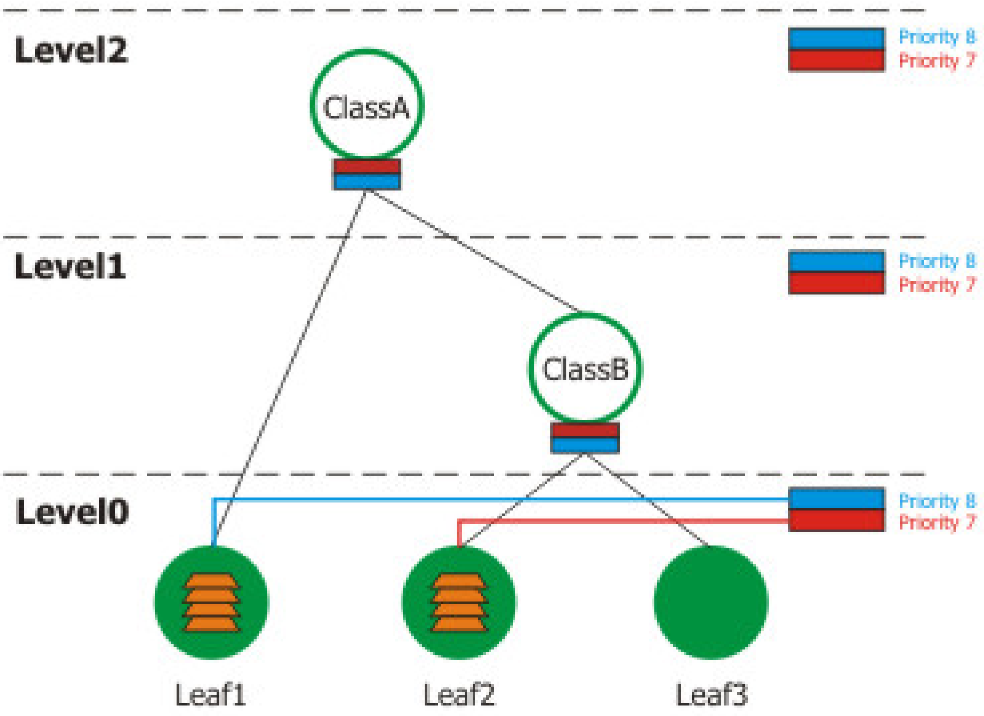
Here are some examples on how the HTB works.

Imagine the following scenario - we have 3 different kinds of traffic, marked in **/ip firewall mangle** (packet_mark1, packet_mark2 and packet_mark3), and now have bulit a HTB hierarchy:

```
[admin@MikroTik] queue tree> add name=ClassA parent=Local max-limit=2048000
[admin@MikroTik] queue tree> add name=ClassB parent=ClassA max-limit=1024000
[admin@MikroTik] queue tree> add name=Leaf1 parent=ClassA max-limit=2048000 \
\... limit-at=1024000 packet-mark=packet_mark1 priority=8
[admin@MikroTik] queue tree> add name=Leaf2 parent=ClassB max-limit=1024000 \
\... limit-at=256000 packet-mark=packet_mark2 priority=7
[admin@MikroTik] queue tree> add name=Leaf3 parent=ClassB max-limit=1024000 \
\... limit-at=768000 packet-mark=packet_mark3 priority=8
[admin@MikroTik] queue tree> print
Flags: X - disabled, I - invalid
 0   name="ClassA" parent=Local packet-mark="" limit-at=0 queue=default
     priority=8 max-limit=2048000 burst-limit=0 burst-threshold=0
     burst-time=0s

 1   name="ClassB" parent=ClassA packet-mark="" limit-at=0 queue=default
     priority=8 max-limit=1024000 burst-limit=0 burst-threshold=0
     burst-time=0s

 2   name="Leaf1" parent=ClassA packet-mark=packet_mark1 limit-at=1024000
     queue=default priority=8 max-limit=2048000 burst-limit=0
     burst-threshold=0 burst-time=0s

 3   name="Leaf2" parent=ClassB packet-mark=packet_mark2 limit-at=256000
     queue=default priority=7 max-limit=1024000 burst-limit=0
     burst-threshold=0 burst-time=0s

 4   name="Leaf3" parent=ClassB packet-mark=packet_mark3 limit-at=768000
     queue=default priority=8 max-limit=1024000 burst-limit=0
     burst-threshold=0 burst-time=0s
[admin@MikroTik] queue tree>
```

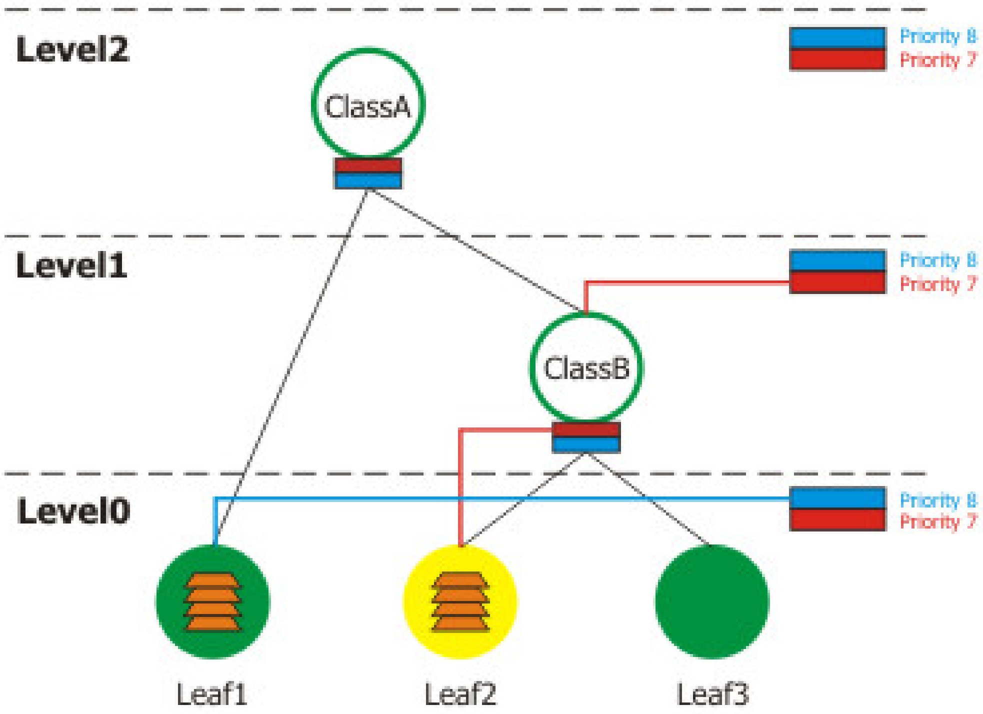
Now let us describe some scenarios, using this HTB hierarchy.

1. Imagine a situation when packets have arrived at Leaf1 and Leaf2. Because of this, Leaf1 attaches itself to this level's (Level 0) self slot with priority=8 and Leaf2 attaches to self slot with priority=7. Leaf3 has nothing to send, so it does nothing.



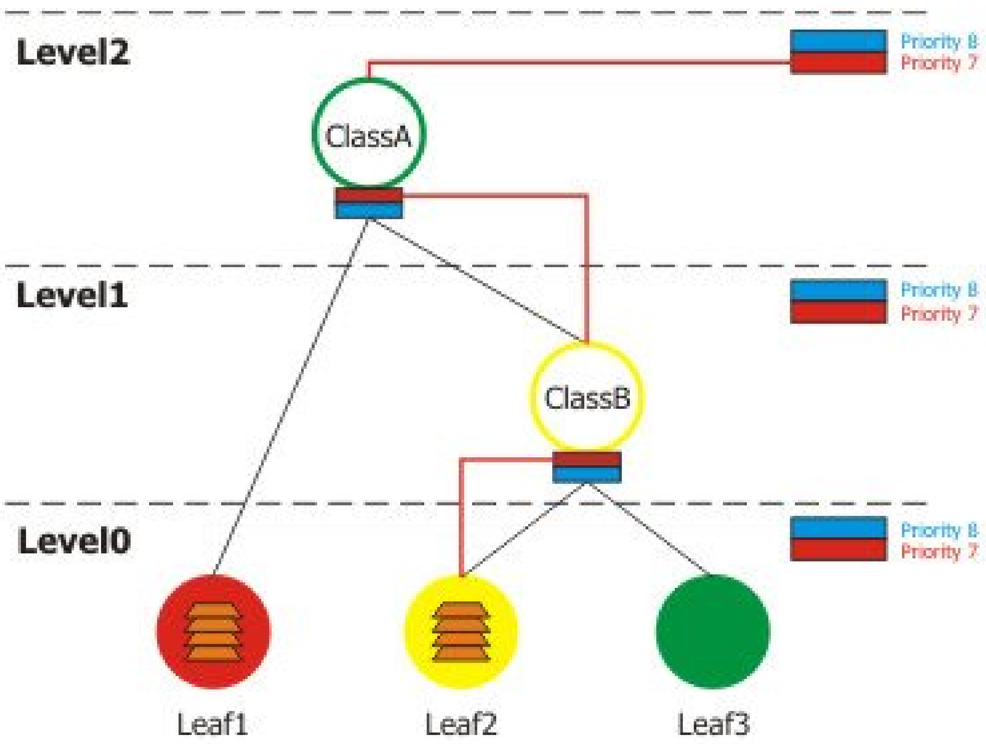
This is a simple situation: there are two active classes (Leaf1 and Leaf2) at Level 0, and as they both are in green state, they are processed in order of their priorities - at first, we serve Leaf2, then Leaf1.

2. Now assume that Leaf2 has to send more than 256kbps, so it needs to go over it's green limit. With the state change, it attaches itself to its parent's (ClassB) inner feed, which recursively attaches itself to Level1 self slot at priority 7. Leaf1 remains in green state - it has packets to send, but their rate is lower than 1Mbps. Leaf3 still has nothing to send.



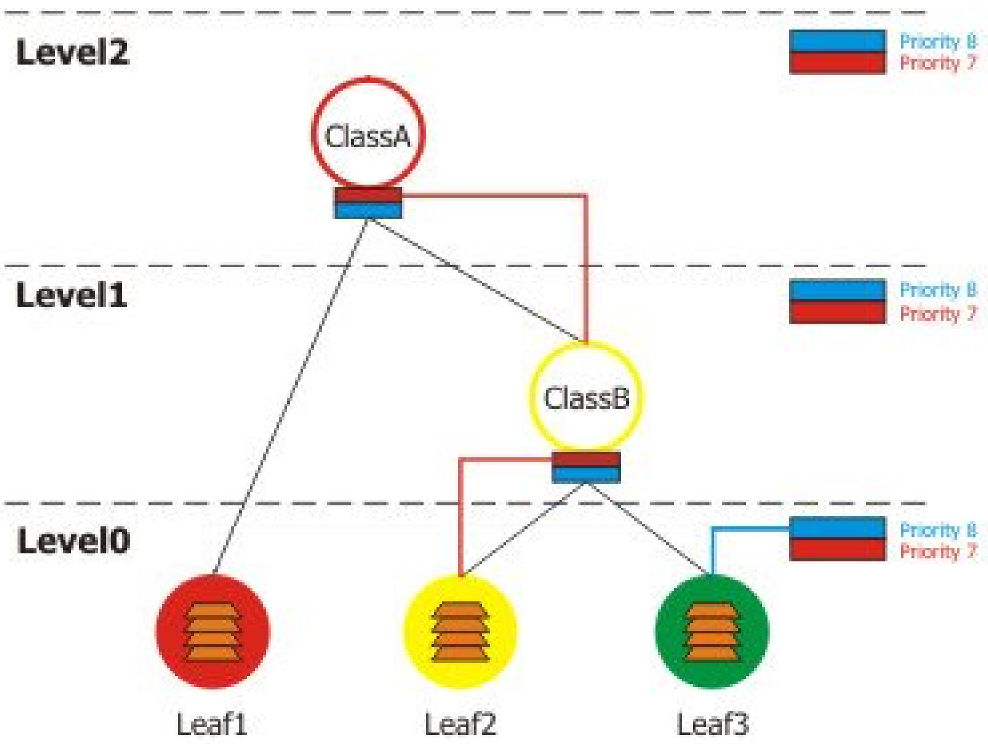
It is very important to understand that Leaf1 now has higher effective priority than Leaf2 (when it is in green state), although we have configured it for a lower priority (8) than Leaf2. It is because Leaf2 has disconnected itself from self feed at Level 0 and is now borrowing rate from its parent (ClassB), which, in turn, has attached to a self feed at Level 1. Thus, the priority of Leaf2 has jumped to Level1. Remember that lowest level is served first, than the next level, and so on, satisfying the attached classes in order of their priority.

3. Consider that Leaf1 has reached its max-limit and changed its state to red, and Leaf2 now uses more than 1Mbps (and less than 2Mbps), so its parent ClassB has to borrow from ClassA and becomes yellow. Leaf3 still has no packets to send.



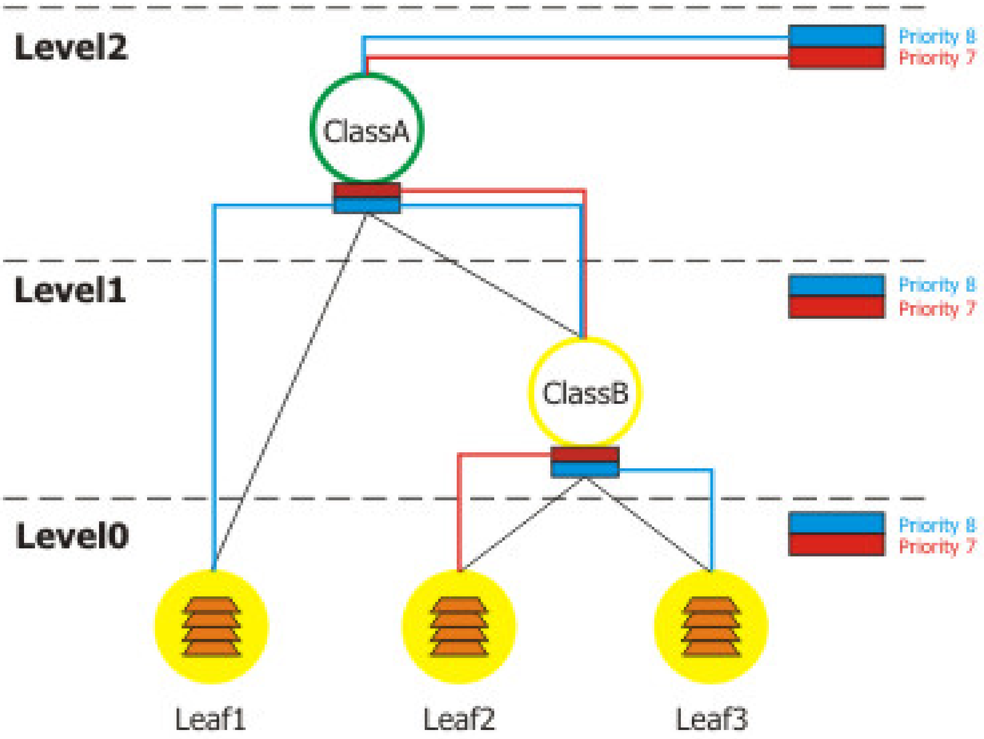
This scenario shows that Leaf1 has reached its max-limit and cannot even borrow from its parent (ClassA), so it is detached from all self slots and inner slots. Leaf2 has recursively reached Level 2, as it borrows from ClassB which, in turn, borrows from ClassA, as it does not have enough rate available. As Leaf3 has no packets to send, the only class that sends is Leaf2.

4. Assume that ClassA reaches its max-limit (2Mbps), so neither ClassB, nor Leaf2 can send as they only rely on borrowing rate, which is impossible as ClassA cannot send. But now, Leaf3 has some packets to send:



In this situation Leaf2 is in yellow state, but it cannot borrow (as Class B cannot borrow from Class A) and Leaf3 is the only class that can send. Note that even though no other calsses, including its parents is able to send, Leaf3 can send perfectly well while is is attached to the Level 0 self feed.

5. Finally, let's see what happens, if Leaf1, Leaf2, Leaf3 and ClassB are in the yellow state, and ClassA is green.

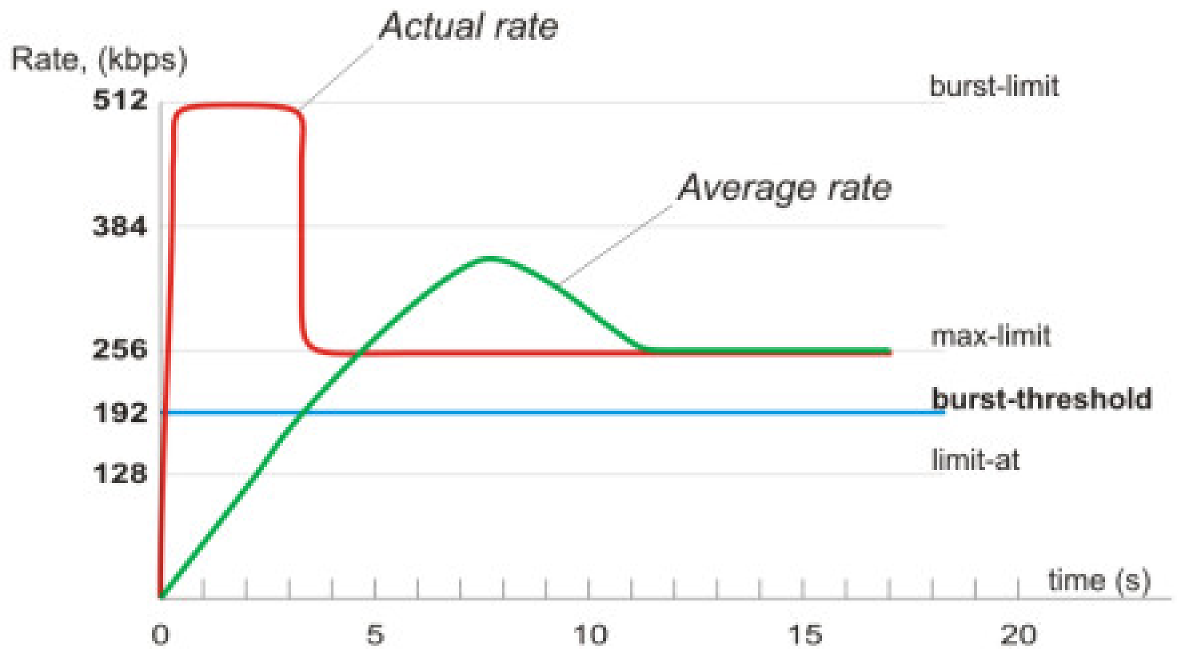


Leaf1 borrows from ClassA, Leaf2 and Leaf3 - from ClassB, and ClassB, in turn, borrows from ClassA. Now all the priorities have 'moved' to Level 2. So Leaf2 is on the highest priority and is served first. As Leaf1 and Leaf3 are of the same priority (8) on the same level (2), they are served using round robin algorithm.

Bursts

Bursts are used to allow higher data rates for a short period of time. Every 1/16 part of the **burst-time**, the router calculates the average data rate of each class over the last **burst-time** seconds. If this average data rate is less than **burst-threshold**, burst is enabled and the effective rate limit (transition to the red state) is set to **burst-limit** bps, otherwise the effective maximal limit falls to **max-limit**.

Let us consider the following setup: **max-limit**=256000, **burst-time**=8, **burst-threshold**=192000 and **burst-limit**=512000. When a user is starting to download a file via HTTP, we can observe such situation:



At the beginning the average data rate over the past 8 seconds is 0bps because no traffic has passed through this ruke before it has been created. Since this average data rate is less than **burst-threshold** (192kbps), burst is allowed. After the first second, the average data rate is (0+0+0+0+0+0+0+512)/8=64kbps, which is less than **burst-threshold**. After the second second, average data rate is (0+0+0+0+0+0+512+512)/8=128kbps. After the third second comes the breakpoint when the average data rate becomes larger than **burst-threshold**. At this moment burst is disabled and the effective data rate limitation falls down to **max-limit** (256kbps).

Note how the **burst-time** was used. The actual duration of burst does not depend of **burst-time** alone! It also depends on the **burst-threshold**/**burst-limit** ratio and the actual data rate passing through the bursty class. In this example the burst ratio was 192000/512000=3/8, the time was 8, and the queue has been trying to utilize all available rate the class was providing, so the burst was 3 seconds long.

Now you can easily see why the **burst-threshold** should be between **limit-at** and **max-limit** for normal operation. If you specify **burst-threshold** higher than **max-limit**, then the average rate will tend to **burst-threshold**, but the effective maximal limit will jump between **max-limit** and **burst-limit** constantly (depending on the actual traffic rate, it may happen even on each evaluation point (1/16th of **burst-time**)).

HTB in RouterOS

In addition to interface queues (one queue or HTB tree per interface), 3 virtual 4 HTB trees maintained by RouterOS:

- global-in
- global-total
- global-out

When adding a simple queue, it creates 3 HTB classes (in global-in, global-total and global-out), but it does not add any classes in interface queue. Queue tree is more flexible - you can add it to any of these HTB's.

When packet travels through the router, it passes 4 HTB trees - global-in, global-total, global-out and output interface queue. If it is directed to the router, it passes global-in and global-total HTB queues. If packets are sent from the router, they go through global-total, global-out and output interface queues

Additional Documents

- http://linux-ip.net/articles/Traffic-Control-HOWTO/overview.html
- http://luxik.cdi.cz/~devik/qos/htb/
- http://www.docum.org/docum.org/docs/

Queue Types

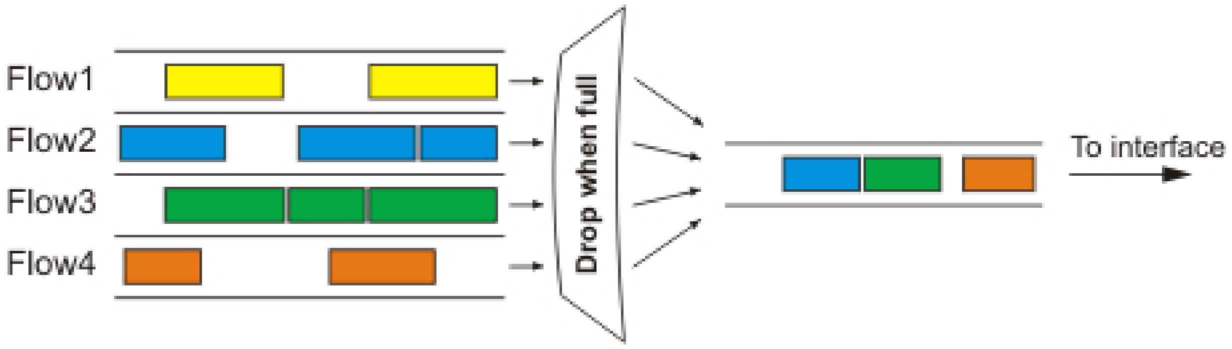
Home menu level: ***/queue type***

Description

You can create your custom queue types in this submenu. Afterwards, you will be able to use them in **/queue tree**, **/queue simple** or **/queue interface**. Note that these queueing disciplines can not limit data rate at all (except for PCQ) - they only reorganize (schedule) packets and drop excess ones (if the queue is getting too long and the managing class can not send the packets quickly enough), so you won't find any rate limitation parameters here (except for PCQ) - only storage limits. Note also that the scheduling is only taking place when the packets are being enqueued in the qdisc, and this only happens when the packets are coming in at the rate faster than the managing class can provide (so this is only a buffer). There are 5 kinds of qdiscs that can be used for storing packets:

PFIFO and BFIFO

These queuing disciplines are based on the FIFO algorithm (First-In First-Out). The difference between PFIFO and BFIFO is that one is measured in packets and the other one in bytes. There is only one parameter called **pfifo-limit** (or **bfifo-limit** in case of BFIFO) which defines how much data a FIFO queue can hold. Every packet that cannot be enqueued (if the queue is full), is dropped. Large queue sizes can increase latency, but utilize channel better.

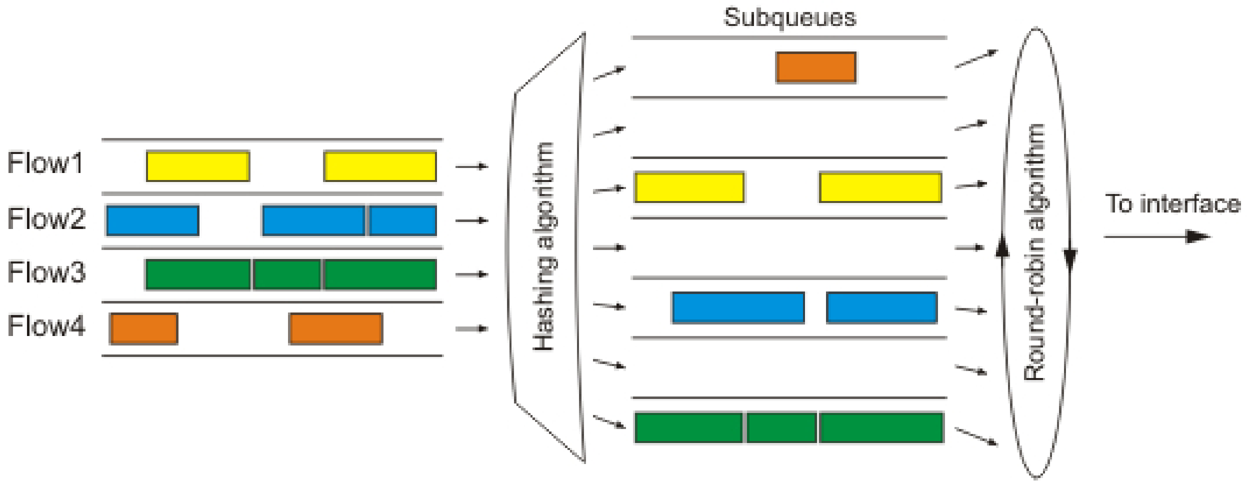


Use FIFO queuing disciplines if you have a noncongested link.

SFQ

Stochastic Fairness Queuing (SFQ) equalizes traffic flows (TCP sessions or UDP streams) when the link is completely full.

The fairness of SFQ is ensured by hashing and round-robin algorithms. Hashing algorithm divides the session traffic over a limited number of subqueues. A traffic flow may be uniquely identified by a tuple (src-address, dst-address, src-port and dst-port), so these parameters are used by SFQ hashing algorithm to classify packets into subqueues.



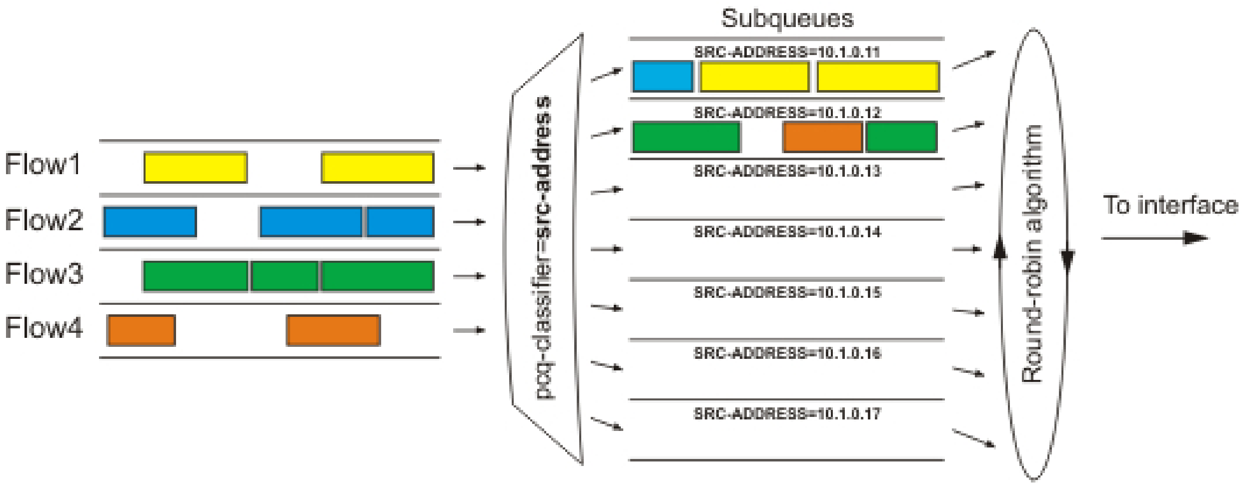
The whole SFQ queue can contain 128 packets and there are 1024 subqueues available for these packets. Each packet stored in a FIFO-like 128 packet buffer, belongs to a certain subqueue, determined by the hash function (a simple function of the tuple values with 10-bit output is used, hence the amount of subqueues is 1024). Stochastic nature of the queueing discipline is observed in that packets from an unpredictable number of flows may actually be classified in the same subqueue. After **sfq-perturb** seconds the hashing algorithm changes and divides the session traffic to other subqueues, so that no separate data flows will be associated with the same subqueue for a long time. The round-robin algorithm dequeues **pcq-allot** bytes from each subqueue in a turn.

Use SFQ for congested links to ensure that connections do not starve. SFQ is especially benefitial on wireless links.

PCQ

To solve some SFQ imperfectness, Per Connection Queuing (PCQ) was created. It is the only classless queuing type in RouterOS that can do rate limitation. It is an improved version of SFQ without its stohastic nature. PCQ also creates subqueues, based on the **pcq-classifier** parameter. Each subqueue has a data rate limit of **pcq-rate** and size of **pcq-limit** packets. The total size of a PCQ queue cannot be greater than **pcq-total-limit** packets.

The following example demonstrates the usage of PCQ with packets, classified by their source address.



If you classify the packets by **src-address** then all packets with different source IP addresses will be grouped into different subqueues. Now you can do the limitation or equalization for each subqueue with the **pcq-rate** parameter. Perhaps, the most significant part is to decide to which interface should we attach this queue. If we will attach it to the Local interface, all traffic from the Public interface will be grouped by src-address (probably it's not what we want), but if we attach it to the Public interface, all traffic from our clients will be grouped by src-address - so we can easily limit or equalize upload for clients. Same can be done for downloads, but in that case **dst-address** classifier will be used, and PCQ put on the locan interface.

To equalize rate among subqueues, classified by the **pcq-classifier**, set the **pcq-rate** to **0**! PCQ can be used to dynamically equalize or shape traffic for multiple users, using little administration. In fact, PCQ always equalizes the subqueues, so the **pcq-rate** is just a cap for equalization - a subqueue may get smaller rate, but will never get higher rate.

RED

Random Early Detection (also known as Random Early Drop, as this is how it actually works) is a queuing mechanism which tries to avoid network congestion by controlling the average queue size. When the average queue size reaches **red-min-threshold**, RED starts to drop packets randomly with linearly increasing probability as the average queue size grows up until the average queue size reaches the **red-max-threshold**. The effective queue size at any moment could be higher than the **red-max-threshold** as the probability does not grow very fast, so it is possible to specify a hard limit for the queue size. When the average queue size reaches **red-max-threshold** or becomes larger, all further packats are dropped until the average queue size does not drop below this valus (at which point probalistic calculations will be activated again).

The average queue size **avg** is *(1-W)*avg+W*q*, where

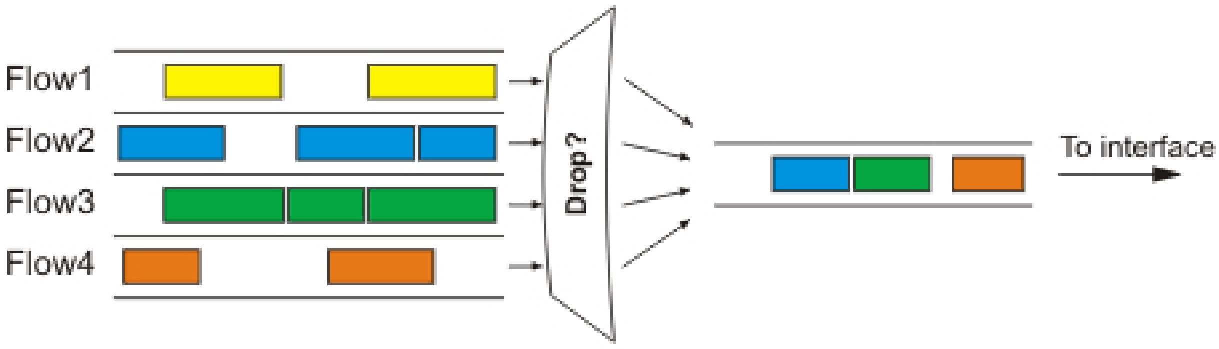
- **q** - current queue length
- **W** - queue weight defined as burst+1-min=(1-(1-W)^burst)/W. Note that log(W) value ir rounded to integer (so W can be 1, 0.1, 0.01, etc.). It is determined experimantally that in many generic cases, W is near to min/10*burst

The **pb** probability value is increasing linearly from 0% to 2% as the average queue size grows from **red-min-threshold** to **red-max-threshold**: *pb=0.02*(avg-min)/(max-min)*.

The packet dropping probability **pb** is increasing with **pb** and with enqueued packet count since the last packet was dropped: *pa=pb/(1-count*pb)*.

It is defined experimentally that a good **red-burst** value is *(min+2*max)/3*. And a good **red-max-threshold** is twice **red-min-threshold**.

Note that in the formulas above, **min** means **red-min-threshold**, **max** means **red-max-threshold** and **burst** means **red-burst**.



Used on congested links with high data rates, as it is fast and TCP-friendly.

Property Description

bfifo-limit (*integer*; default: **15000**) - maximum number of bytes that the BFIFO queue can hold

kind (*bfifo* | *pcq* | *pfifo* | *red* | *sfq*) - which queuing discipline to use

- **bfifo** - Bytes First-In, First-Out
- **pcq** - Per Connection Queue
- **pfifo** - Packets First-In, First-Out
- **red** - Random Early Detection
- **sfq** - Stohastic Fairness Queuing

name (*name*) - reference name of the queue type

pcq-classifier (*dst-address* | *dst-port* | *src-address* | *src-port*; default: **""**) - list classifiers for grouping packets into PCQ subqueues. Several classifiers can be used at once, e.g., src-address,src-port will group all packets with different source address and source-ports into separate subqueues

pcq-limit (*integer*; default: **50**) - number of packets that a single PCQ sub-queue can hold

pcq-rate (*integer*; default: **0**) - maximal data rate allowed for each PCQ sub-queue. This is a rate cap, as the subqueues will be equalized anyway

- **0** - no limitation set (only equalize rates between subqueues)

pcq-total-limit (*integer*; default: **2000**) - number of packets that the whole PCQ queue can hold

pfifo-limit (*integer*) - maximum number of packets that the PFIFO queue can hold

red-avg-packet (*integer*; default: **1000**) - average packet size, used for tuning average queue recalculation time

red-burst (*integer*) - a measure of how fast the average queue size will be influenced by the real queue size, given in bytes. Larger values will smooth the changes, so longer bursts will be allowed

red-limit (*integer*) - hard limit on queue size in bytes. If the real queue size (not average) exceeds this value then all further packets will be discarded until the queue size drops below. This should be higher than red-max-threshold+red-burst

red-max-threshold (*integer*) - upper limit for average queue size, in bytes. When the size reaches this value, all further packets shall be dropped

red-min-threshold (*integer*) - lower limit for average queue size, in bytes. When the size reaches this value, RED starts to drop packets randomly with a calculated probability

sfq-allot (*integer*; default: **1514**) - amount of bytes that a subqueue is allowed to send before the next subqueue gets a turn (amount of bytes which can be sent from a subqueue in a single round-robin turn), should be at least 1514 for links with 1500 byte MTU

sfq-perturb (*integer*; default: **5**) - how often to shake (perturb) SFQ's hashing algorithm, in seconds

Interface Default Queues

Home menu level: ***/queue interface***

Description

In order to send packets over an interface, they have to be enqueued in a queue even if you do not want to limit traffic at all. Here you can specify the queue type which will be used for transmitting data.

Note that once you configure tree queues for a listed interface, the interface default queue is no longer active for that particular interface, so you need to make sure all packets that goes out through this interface are filtered into some qdiscs inside the HTB tree. Otherwise the packets that are not filtered, are sent out directly (at effective higher priority than any of the packets in the HTB tree), and unbuffered, which ultimately lead to suboptimal performance.

Property Description

interface (*read-only: name*) - name of the interface

queue (*name*; default: **default**) - queue type which will be used for the interface

Example

Set the wireless interface to use **wireless-default** queue:

```
[admin@MikroTik] queue interface> set 0 queue=wireless-default
[admin@MikroTik] queue interface> print
 # INTERFACE QUEUE
 0 wlan1     wireless-default
[admin@MikroTik] queue interface>
```

Simple Queues

Description

The simpliest way to limit data rate for specific IP addresses and/or subnets, is to use simple queues.

You can also use simple queues to build advanced QoS applications. They have useful integrated features:

- Peer-to-peer traffic queuing
- Applying queue rules on chosen time intervals
- Priorities
- Using multiple packet marks from */ip firewall mangle*
- Shaping of bidirectional traffic (one limit for the total of upload + download)

Property Description

burst-limit (*integerinteger*) - maximum data rate which can be reached while the burst is active, in form of in/out (target upload/download)

burst-threshold (*integerinteger*) - average data rate limit, until which the burst is allowed. If the average data rate over the last burst-time seconds is less than burst-threshold, the actual data rate may reach burst-limit. Otherwise the hard limit is reset to max-limit. Set in form of in/out (target upload/download)

burst-time (*integerinteger*) - period of time, in seconds, over which the average data rate is calculated, in form of in/out (target upload/download)

direction (*none* | *both* | *upload* | *download*) - traffic flow directions from the targets' point of view, affected by this queue

- **none** - the queue is effectively inactive
- **both** - the queue limits both target upload and target download
- **upload** - the queue limits only target upload, leaving the download rates unlimited
- **download** - the queue limits only target download, leaving the upload rates unlimited

dst-address (*IP addressnetmask*) - destination address to match

dst-netmask (*netmask*) - netmask for dst-address

interface (*text*) - interface, this queue applies to (i.e., the interface the target is connected to)

limit-at (*integerinteger*) - CIR, in form of in/out (target upload/download)

max-limit (*integerinteger*) - MIR (in case burst is not active), in form of in/out (target upload/download)

name (*text*) - descriptive name of the queue

p2p (*all-p2p* | *bit-torrent* | *blubster* | *direct-connect* | *edonkey* | *fasttrack* | *gnutella* | *soulseek* | *winmx*) - which type of P2P traffic to match

- **all-p2p** - match all P2P traffic

packet-marks (*multiple choice: name*; default: "") - list of packet marks (set by /ip firewall mangle) to match. Multiple packet marks are separated by commas (",")

parent (*name*) - name of the parent queue in the hierarchy. Can only be another simple queue

priority (*integer*: 1..8) - priority of the queue. 1 is the highest, 8 - the lowest

queue (*namename*; default: **default/default**) - name of the queue from /queue type, in form of in/out

target-addresses (*multiple choice: IP addressnetmask*) - limitation target IP addresses (source addresses). Multiple addresses are separated by commas

time (*timetimesat* | *fri* | *thu* | *wed* | *tue* | *mon* | *sun*; default: "") - limit queue effect to a specified time period

total-burst-limit (*integer*) - burst limit for global-total (cumulative rate, upload + download) queue

total-burst-threshold (*integer*) - burst threshold for global-total (cumulative rate, upload + download) queue

total-burst-time (*time*) - burst time for global-total queue

total-limit-at (*integer*) - limit-at for global-total (cumulative rate, upload + download) queue

total-max-limit (*integer*) - max-limit for global-total (cumulative rate, upload + download) queue

total-queue (*name*) - queuing discipline to use for global-total queue

Queue Trees

Home menu level: ***/queue tree***

Description

The queue trees should be used when you want to use sophisticated data rate allocation based on protocols, ports, groups of IP addresses, etc. At first you have to mark packet flows with a mark under **/ip firewall mangle** and then use this mark as an identifier for packet flows in queue trees.

Property Description

burst-limit (*integer*) - maximum data rate which can be reached while the burst is active

burst-threshold (*integer*) - average data rate limit, until which the burst is allowed. If the average data rate over the last burst-time seconds is less than burst-threshold, the actual data rate may reach burst-limit. Otherwise the hard limit is reset to max-limit

burst-time (*time*) - period of time, in seconds, over which the average data rate is calculated

limit-at (*integer*) - CIR

max-limit (*integer*) - MIR (in case burst is not active)

name (*text*) - descriptive name for the queue

packet-mark (*text*) - packet flow mark (set by /ip firewall mangle) to match. This creates a filter that puts the packets with the given mark into this queue

parent (*text*) - name of the parent queue. The top-level parents are the available interfaces (actually, main HTB). Lower level parents can be other tree queues

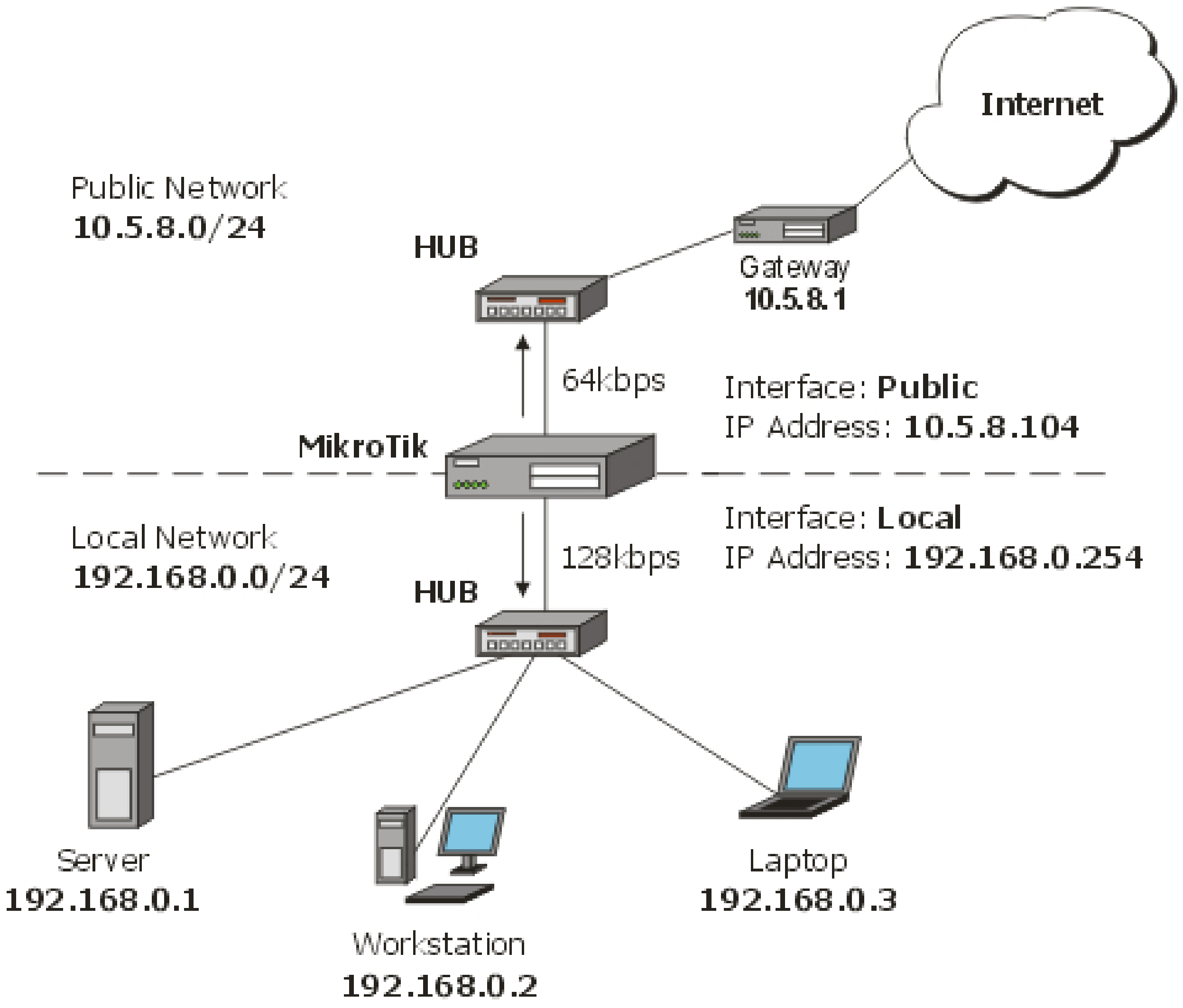
priority (*integer*: 1..8) - priority of the queue. 1 is the highest, 8 - the lowest

queue (*text*) - name of the queue type. Types are defined under /queue type

Application Examples

Example of emulating a 128Kibps/64Kibps Line

Assume, we want to emulate a 128Kibps download and 64Kibps upload line, connecting IP network **192.168.0.0/24**. The network is served through the Local interface of customer's router. The basic network setup is in the following diagram:



To solve this situation, we will use simple queues.

IP addresses on MikroTik router:

```
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   192.168.0.254/24   192.168.0.0     192.168.0.255   Local
 1   10.5.8.104/24      10.5.8.0        10.5.8.255      Public
[admin@MikroTik] ip address>
```

And routes:

```
[admin@MikroTik] ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        G GATEWAY         DIS INTE...
 0 A S  0.0.0.0/0                          r 10.5.8.1        1   Public
 1 ADC  10.5.8.0/24        10.5.8.104                        0   Public
 2 ADC  192.168.0.0/24     192.168.0.254                     0   Local
[admin@MikroTik] ip route>
```

Add a simple queue rule, which will limit the download traffic to 128Kib/s and upload to 64Kib/s for clients on the network **192.168.0.0/24**, served by the interface **Local**:

```
[admin@MikroTik] queue simple> add name=Limit-Local interface=Local \
\... target-address=192.168.0.0/24 max-limit=65536/131072
[admin@MikroTik] queue simple> print
Flags: X - disabled, I - invalid, D - dynamic
 0    name="Limit-Local" target-addresses=192.168.0.0/24 dst-address=0.0.0.0/0
      interface=Local parent=none priority=8 queue=default/default
      limit-at=0/0 max-limit=65536/131072 total-queue=default
[admin@MikroTik] queue simple>
```

The **max-limit** parameter cuts down the maximum available bandwidth. From the clients' point of view, the value **65536/131072** means that they will get maximum of 131072bps for download and 65536bps for upload. The **target-addresses** parameter defines the target network (or networks, separated by a comma) to which the queue rule will be applied.

Now see the traffic load:

```
[admin@MikroTik] interface> monitor-traffic Local
  received-packets-per-second: 7
       received-bits-per-second: 68kbps
        sent-packets-per-second: 13
           sent-bits-per-second: 135kbps

[admin@MikroTik] interface>
```

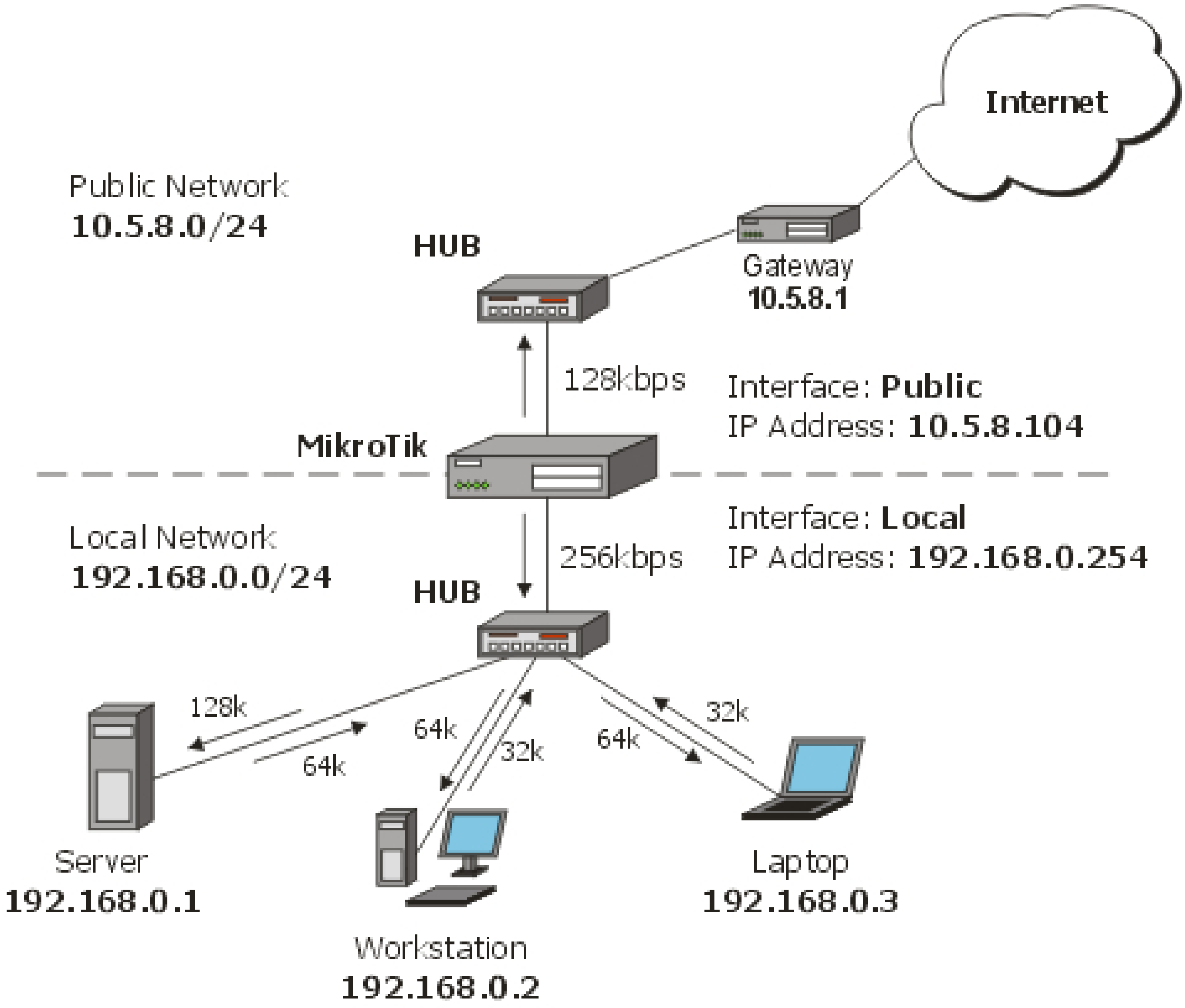
Probably, you want to exclude the server from being limited, if so, add a queue for it without any limitation (**max-limit=0/0** which means no limitation) and move it to the beginning of the list:

```
[admin@MikroTik] queue simple> add name=Server target-addresses=192.168.0.1/32 \
\... interface=Local
[admin@MikroTik] queue simple> print
Flags: X - disabled, I - invalid, D - dynamic
 0    name="Limit-Local" target-addresses=192.168.0.0/24 dst-address=0.0.0.0/0
      interface=Local parent=none priority=8 queue=default/default
      limit-at=0/0 max-limit=65536/131072 total-queue=default

 1    name="Server" target-addresses=192.168.0.1/32 dst-address=0.0.0.0/0
      interface=Local parent=none priority=8 queue=default/default
      limit-at=0/0 max-limit=0/0 total-queue=default
[admin@MikroTik] queue simple> mo 1 0
[admin@MikroTik] queue simple> print
Flags: X - disabled, I - invalid, D - dynamic
 0    name="Server" target-addresses=192.168.0.1/32 dst-address=0.0.0.0/0
      interface=Local parent=none priority=8 queue=default/default
      limit-at=0/0 max-limit=0/0 total-queue=default

 1    name="Limit-Local" target-addresses=192.168.0.0/24 dst-address=0.0.0.0/0
      interface=Local parent=none priority=8 queue=default/default
      limit-at=0/0 max-limit=65536/131072 total-queue=default
[admin@MikroTik] queue simple>
```

Queue Tree Example With Masquerading

In the previous example we dedicated 128Kib/s download and 64Kib/s upload traffic for the local network. In this example we will guarantee 256Kib/s download (128Kib/s for the server, 64Kib/s for the Workstation and also 64Kib/s for the Laptop) and 128Kib/s for upload (64/32/32Kib/s, respectivelly) for local network devices. Additionally, if there is spare bandwidth, share it among users equally. For example, if we turn off the laptop, share its 64Kib/s download and 32Kib/s upload to the Server and Workstation.

When using masquerading, you have to mark the outgoing connection with **new-connection-mark** and take the **mark-connection** action. When it is done, you can mark all packets which belong to this connection with the **new-packet-mark** and use the **mark-packet** action.



1. At first, mark the Server's download and upload traffic. With the first rule we will mark the outgoing connection and with the second one, all packets, which belong to this connection:

```
[admin@MikroTik] ip firewall mangle> add src-address=192.168.0.1/32 \
\... action=mark-connection new-connection-mark=server-con chain=prerouting
[admin@MikroTik] ip firewall mangle> add connection-mark=server-con \
\... action=mark-packet new-packet-mark=server chain=prerouting
[admin@MikroTik] ip firewall mangle> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=prerouting src-address=192.168.0.1 action=mark-connection
     new-connection-mark=server-con

 1   chain=prerouting connection-mark=server-con action=mark-packet
     new-packet-mark=server
[admin@MikroTik] ip firewall mangle>
```

2. The same for Laptop and Workstation:

```
[admin@MikroTik] ip firewall mangle> add src-address=192.168.0.2 \
\... action=mark-connection new-connection-mark=lap_works-con chain=prerouting
[admin@MikroTik] ip firewall mangle> add src-address=192.168.0.3 \
\... action=mark-connection new-connection-mark=lap_works-con chain=prerouting
[admin@MikroTik] ip firewall mangle> add connection-mark=lap_works-con \
\... action=mark-packet new-packet-mark=lap_work chain=prerouting
[admin@MikroTik] ip firewall mangle> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=prerouting src-address=192.168.0.1 action=mark-connection
```

```
     new-connection-mark=server-con

 1   chain=prerouting connection-mark=server-con action=mark-packet
     new-packet-mark=server

 2   chain=prerouting src-address=192.168.0.2 action=mark-connection
     new-connection-mark=lap_works-con

 3   chain=prerouting src-address=192.168.0.3 action=mark-connection
     new-connection-mark=lap_works-con

 4   chain=prerouting connection-mark=lap_works-con action=mark-packet
     new-packet-mark=lap_work
[admin@MikroTik] ip firewall mangle>
```

As you can see, we marked connections that belong for Laptop and Workstation with the same flow.

3. In **/queue tree** add rules that will limit Server's download and upload:

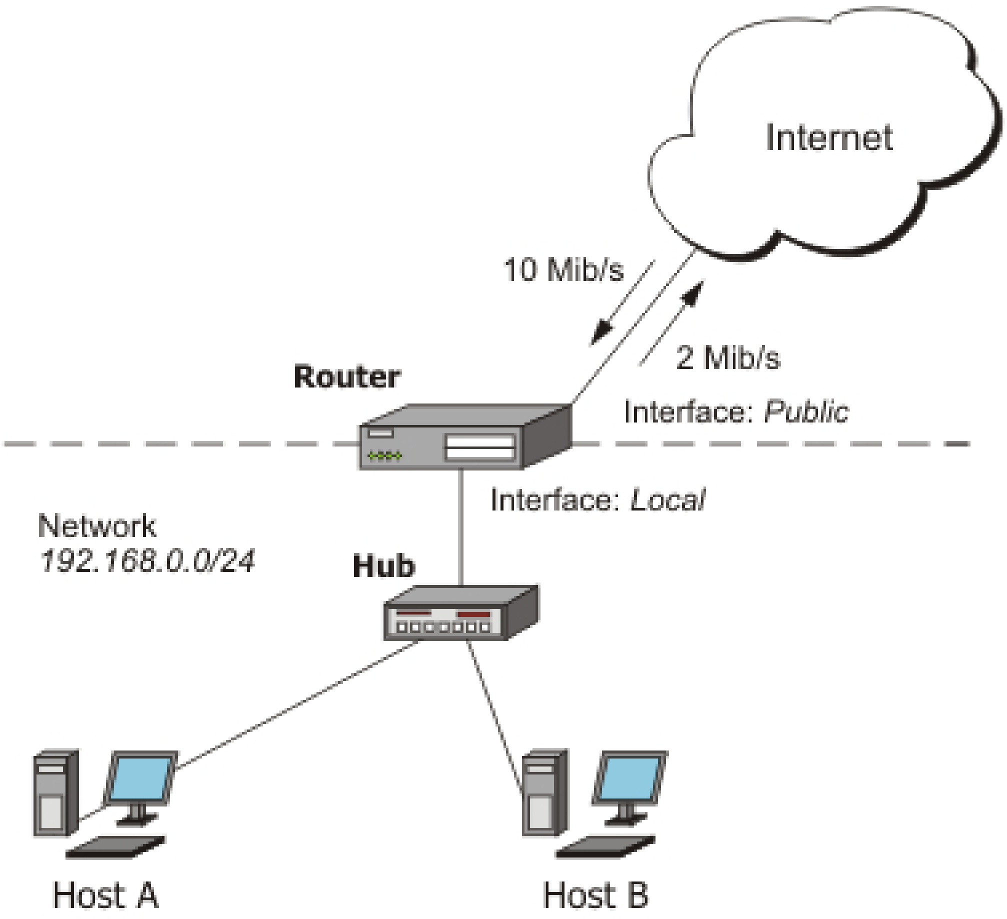
```
[admin@MikroTik] queue tree> add name=Server-Download parent=Local \
\... limit-at=131072 packet-mark=server max-limit=262144
[admin@MikroTik] queue tree> add name=Server-Upload parent=Public \
\... limit-at=65536 packet-mark=server max-limit=131072
[admin@MikroTik] queue tree> print
Flags: X - disabled, I - invalid
 0   name="Server-Download" parent=Local packet-mark=server limit-at=131072
     queue=default priority=8 max-limit=262144 burst-limit=0
     burst-threshold=0 burst-time=0s

 1   name="Server-Upload" parent=Public packet-mark=server limit-at=65536
     queue=default priority=8 max-limit=131072 burst-limit=0
     burst-threshold=0 burst-time=0s
[admin@MikroTik] queue tree>
```

And similar config for Laptop and Workstation:

```
[admin@MikroTik] queue tree> add name=Laptop-Wkst-Down parent=Local \
\... packet-mark=lap_work limit-at=65535 max-limit=262144
[admin@MikroTik] queue tree> add name=Laptop-Wkst-Up parent=Public \
\... packet-mark=lap_work limit-at=32768 max-limit=131072
[admin@MikroTik] queue tree> print
Flags: X - disabled, I - invalid
 0   name="Server-Download" parent=Local packet-mark=server limit-at=131072
     queue=default priority=8 max-limit=262144 burst-limit=0
     burst-threshold=0 burst-time=0s

 1   name="Server-Upload" parent=Public packet-mark=server limit-at=65536
     queue=default priority=8 max-limit=131072 burst-limit=0
     burst-threshold=0 burst-time=0s

 2   name="Laptop-Wkst-Down" parent=Local packet-mark=lap_work limit-at=65535
     queue=default priority=8 max-limit=262144 burst-limit=0
     burst-threshold=0 burst-time=0s

 3   name="Laptop-Wkst-Up" parent=Public packet-mark=lap_work limit-at=32768
     queue=default priority=8 max-limit=131072 burst-limit=0
     burst-threshold=0 burst-time=0s
[admin@MikroTik] queue tree>
```

Equal bandwidth sharing among users

This example shows how to equally share 10Mibps download and 2Mbps upload among active users in the network **192.168.0.0/24**. If **Host A** is downloading 2 Mbps, **Host B** gets 8 Mbps and vice versa. There might be situations when both hosts want to use maximum bandwidth (10 Mbps), then they will receive 5 Mbps each, the same goes for upload. This setup is also valid for more than 2 users.



At first, mark all traffic, coming from local network **192.168.0.0/24** with a mark **users**:

```
/ip firewall mangle add chain=forward src-address=192.168.0.0/24 \
   action=mark-connection new-connection-mark=users-con
/ip firewall mangle add connection-mark=users-con action=mark-packet \
   new-packet-mark=users chain=forward
```

Now we will add 2 new PCQ types. The first, called **pcq-download** will group all traffic by destination address. As we will attach this queue type to the **Local** interface, it will create a dynamic queue for each destination address (user) which is downloading to the network **192.168.0.0/24**. The second type, called **pcq-upload** will group the traffic by source address. We will attach this queue to the **Public** interface so it will make one dynamic queue for each user who is uploading to Internet from the local network **192.168.0.0/24**.

```
/queue type add name=pcq-download kind=pcq pcq-classifier=dst-address
/queue type add name=pcq-upload kind=pcq pcq-classifier=src-address
```

Finally, make a queue tree for download traffic:

```
/queue tree add name=Download parent=Local max-limit=10240000
/queue tree add parent=Download queue=pcq-download packet-mark=users
```

And for upload traffic:

```
/queue tree add name=Upload parent=Public max-limit=2048000
/queue tree add parent=Upload queue=pcq-upload packet-mark=users
```

Note! If your ISP cannot guarantee you a fixed amount of traffic, you can use just one queue for upload and one for download, attached directly to the interface:

```
/queue tree add parent=Local queue=pcq-download packet-mark=users
/queue tree add parent=Public queue=pcq-upload packet-mark=users
```

Filter

Document revision 2.8 (February 11, 2008, 4:14 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The firewall implements packet filtering and thereby provides security functions that are used to manage data flow to, from and through the router. Along with the Network Address Translation it serve as a tool for preventing unauthorized access to directly attached networks and the router itself as well as a filter for outgoing traffic.

Quick Setup Guide

- To add a firewall rule which drops all **TCP** packets that are destined to port **135** and going through the router, use the following command:

```
/ip firewall filter add chain=forward dst-port=135 protocol=tcp action=drop
```

- To deny acces to the router via Telnet (protocol TCP, port 23), type the following command:

```
/ip firewall filter add chain=input protocol=tcp dst-port=23 action=drop
```

- To only allow not more than 5 simultaneous connections from each of the clients, do the following:

```
/ip firewall filter add chain=forward protocol=tcp tcp-flags=syn connection-limit=6,32
action=drop
```

Specifications

Packages required: ***system***
License required: ***level1 (P2P filters limited to 1), level3***
Home menu level: ***/ip firewall filter***
Standards and Technologies: ***IP***, ***RFC2113***

Hardware usage: *Increases with filtering rules count*

Firewall Filter

Home menu level: ***/ip firewall filter***

Description

Network firewalls keep outside threats away from sensitive data available inside the network. Whenever different networks are joined together, there is always a threat that someone from outside of your network will break into your LAN. Such break-ins may result in private data being stolen and distributed, valuable data being altered or destroyed, or entire hard drives being erased. Firewalls are used as a means of preventing or minimizing the security risks inherent in connecting to other networks. Properly configured firewall plays a key role in efficient and secure network infrastrure deployment.

MikroTik RouterOS has very powerful firewall implementation with features including:

- stateful packet inspection
- Layer-7 protocol detection
- peer-to-peer protocols filtering
- traffic classification by:
    - source MAC address
    - IP addresses (network or list) and address types (broadcast, local, multicast, unicast)
    - port or port range
    - IP protocols
    - protocol options (ICMP type and code fields, TCP flags, IP options and MSS)
    - interface the packet arrived from or left through
    - internal flow and connection marks
    - DSCP byte
    - packet content
    - rate at which packets arrive and sequence numbers
    - packet size
    - packet arrival time
    - and much more!

General Filtering Principles

The firewall operates by means of firewall rules. A rule is a definitive form expression that tells the router what to do with a particular IP packet. Each rule consists of two parts that are the matcher which matches traffic flow against given conditions and the action which defines what to do with the mathched packets. Rules are organized in chains for better management.

The filter facility has three default chains: **input**, **forward** and **output** that are responsible for traffic coming from, throurh and to the router, respectively. New user-defined chains can be added, as necessary. Since these chains have no default traffic to match, rules with **action=jump** and relevant **jump-target** should be added to one or more of the three default chains.

Filter Chains

As mentioned before, the firewall filtering rules are grouped together in chains. It allows a packet to be matched against one common criterion in one chain, and then passed over for processing against some other common criteria to another chain. For example a packet should be matched against the **IP address:port** pair. Of course, it could be achieved by adding as many rules with **IP address:port** match as required to the **forward** chain, but a better way could be to add one rule that matches traffic from a particular IP address, e.g.: `/ip firewall filter add src-address=1.1.1.2/32 jump-target="mychain"` and in case of successfull match passes control over the IP packet to some other chain, *id est* **mychain** in this example. Then rules that perform matching against separate ports can be added to **mychain** chain without specifying the IP addresses.

- **input** - used to process packets entering the router through one of the interfaces with the destination IP address which is one of the router's addresses. Packets passing through the router are not processed against the rules of the input chain
- **forward** - used to process packets passing through the router
- **output** - used to process packets originated from the router and leaving it through one of the interfaces. Packets passing through the router are not processed against the rules of the output chain

There are three predefined chains, which cannot be deleted:

When processing a chain, rules are taken from the chain in the order they are listed there from top to bottom. If a packet matches the criteria of the rule, then the specified action is performed on it, and no more rules are processed in that chain (the exception is the **passthrough** action). If a packet has not matched any rule within the chain, then it is accepted.

Property Description

action (*accept* | *add-dst-to-address-list* | *add-src-to-address-list* | *drop* | *jump* | *log* | *passthrough* | *reject* | *return* | *tarpit*; default: **accept**) - action to undertake if the packet matches the rule

- **accept** - accept the packet. No action is taken, i.e. the packet is passed through and no more rules are applied to it
- **add-dst-to-address-list** - adds destination address of an IP packet to the address list specified by address-list parameter
- **add-src-to-address-list** - adds source address of an IP packet to the address list specified by address-list parameter
- **drop** - silently drop the packet (without sending the ICMP reject message)
- **jump** - jump to the chain specified by the value of the jump-target parameter
- **log** - each match with this action will add a message to the system log
- **passthrough** - ignores this rule and goes on to the next one
- **reject** - reject the packet and send an ICMP reject message

- **return** - passes control back to the chain from where the jump took place
- **tarpit** - captures and holds incoming TCP connections (replies with SYN/ACK to the inbound TCP SYN packet)

address-list (*name*) - specifies the name of the address list to collect IP addresses from rules having action=add-dst-to-address-list or action=add-src-to-address-list actions. These address lists could be later used for packet matching

address-list-timeout (*time*; default: **00:00:00**) - time interval after which the address will be removed from the address list specified by address-list parameter. Used in conjunction with add-dst-to-address-list or add-src-to-address-list actions

- **00:00:00** - leave the address in the address list forever

chain (*forward* | *input* | *outputname*) - specifies the chain to put a particular rule into. As the different traffic is passed through different chains, always be careful in choosing the right chain for a new rule. If the input does not match the name of an already defined chain, a new chain will be created

comment (*text*) - a descriptive comment for the rule. A comment can be used to identify rules form scripts

connection-bytes (*integerinteger*) - matches packets only if a given amount of bytes has been transfered through the particular connection

- **0** - means infinity, exempli gratia: connection-bytes=2000000-0 means that the rule matches if more than 2MB has been transfered through the relevant connection

connection-limit (*integernetmask*) - restrict connection limit per address or address block

connection-mark (*name*) - matches packets marked via mangle facility with particular connection mark

connection-state (*estabilished* | *invalid* | *new* | *related*) - interprets the connection tracking analysis data for a particular packet

- **estabilished** - a packet which belongs to an existing connection, exempli gratia a reply packet or a packet which belongs to already replied connection
- **invalid** - a packet which could not be identified for some reason. This includes out of memory condition and ICMP errors which do not correspond to any known connection. It is generally advised to drop these packets
- **new** - a packet which begins a new TCP connection
- **related** - a packet which is related to, but not part of an existing connection, such as ICMP errors or a packet which begins FTP data connection (the later requires enabled FTP connection tracking helper under /ip firewall service-port)

connection-type (*ftp* | *gre* | *h323* | *irc* | *mms* | *pptp* | *quake3* | *tftp*) - matches packets from related connections based on information from their connection tracking helpers. A relevant connection helper must be enabled under /ip firewall service-port

content (*text*) - the text packets should contain in order to match the rule

dscp (*integer*: 0..63) - DSCP (ex-ToS) IP header field value

dst-address (*IP addressnetmaskIP addressIP address*) - specifies the address range an IP packet is destined to. Note that console converts entered address/netmask value to a valid network address, i.e.:1.1.1.1/24 is converted to 1.1.1.0/24

dst-address-list (*name*) - matches destination address of a packet against user-defined address list

dst-address-type (*unicast* | *local* | *broadcast* | *multicast*) - matches destination address type of the IP packet, one of the:

- **unicast** - IP addresses used for one point to another point transmission. There is only one sender and one receiver in this case
- **local** - matches addresses assigned to router's interfaces
- **broadcast** - the IP packet is sent from one point to all other points in the IP subnetwork
- **multicast** - this type of IP addressing is responsible for transmission from one or more points to a set of other points

dst-limit (*integertimeintegerdst-address* | *dst-port* | *src-addresstime*) - limits the packet per second (pps) rate on a per destination IP or per destination port base. As opposed to the limit match, every destination IP address / destination port has it's own limit. The options are as follows (in order of appearance):

- **count** - maximum average packet rate, measured in packets per second (pps), unless followed by time option
- **time** - specifies the time interval over which the packet rate is measured
- **burst** - number of packets to match in a burst
- **mode** - the classifier(-s) for packet rate limiting
- **expire** - specifies interval after which recorded IP addresses / ports will be deleted

dst-port (*integer*: 0..65535*integer*: 0..65535) - destination port number or range

fragment (yes | no) - whether the packet is a fragment of an IP packet. Starting packet (i.e., first fragment) does not count. Note that is the connection tracking is enabled, there will be no fragments as the system automatically assembles every packet

hotspot (*multiple choice: auth* | *from-client* | *http* | *local-dst* | *to-client*) - matches packets received from clients against various HotSpot conditions. All values can be negated

- **auth** - true, if a packet comes from an authenticted HotSpotclient
- **from-client** - true, if a packet comes from any HotSpot client
- **http** - true, if a HotSpot client sends a packet to the address and port previously detected as his proxy server (Universal Proxy technique) or if the destination port is 80 and transparent proxying is enabled for that particular client
- **local-dst** - true, if a packet has local destination IP address
- **to-client** - true, if a packet is sent to a client

icmp-options (*integerinteger*) - matches ICMP Type:Code fields

in-bridge-port (*name*) - actual interface the packet has entered the router through (if bridged, this property matches the actual bridge port, while in-interface - the bridge itself)

in-interface (*name*) - interface the packet has entered the router through (if the interface is bridged, then the packet will appear to come from the bridge interface itself)

ingress-priority (*integer*: 0..63) - INGRESS (received) priority of the packet, if set (0 otherwise). The priority may be derived from either VLAN or WMM priority

ipv4-options (*any* | *loose-source-routing* | *no-record-route* | *no-router-alert* | *no-source-routing* | *no-timestamp* | *none* | *record-route* | *router-alert* | *strict-source-routing* | *timestamp*) - match ipv4 header options

- **any** - match packet with at least one of the ipv4 options

- **loose-source-routing** - match packets with loose source routing option. This option is used to route the internet datagram based on information supplied by the source
- **no-record-route** - match packets with no record route option. This option is used to route the internet datagram based on information supplied by the source
- **no-router-alert** - match packets with no router alter option
- **no-source-routing** - match packets with no source routing option
- **no-timestamp** - match packets with no timestamp option
- **record-route** - match packets with record route option
- **router-alert** - match packets with router alter option
- **strict-source-routing** - match packets with strict source routing option
- **timestamp** - match packets with timestamp

jump-target (*forward* | *input* | *outputname*) - name of the target chain to jump to, if the action=jump is used

layer7-protocol (*name*) - Layer 7 filter name as set in the /ip firewall layer7-protocol menu. Caution: this matcher needs high computational power

limit (*integertimeinteger*) - restricts packet match rate to a given limit. Usefull to reduce the amount of log messages

- **count** - maximum average packet rate, measured in packets per second (pps), unless followed by time option
- **time** - specifies the time interval over which the packet rate is measured
- **burst** - number of packets to match in a burst

log-prefix (*text*) - all messages written to logs will contain the prefix specified herein. Used in conjunction with action=log

nth (*integerinteger*: 0..15*integer*) - match a particular Nth packet received by the rule. One of 16 available counters can be used to count packets

- **every** - match every every+1th packet. For example, if every=1 then the rule matches every 2nd packet
- **counter** - specifies which counter to use. A counter increments each time the rule containing nth match matches
- **packet** - match on the given packet number. The value by obvious reasons must be between 0 and every. If this option is used for a given counter, then there must be at least every+1 rules with this option, covering all values between 0 and every inclusively.

out-bridge-port (*name*) - actual interface the packet is leaving the router through (if bridged, this property matches the actual bridge port, while out-interface - the bridge itself)

out-interface (*name*) - interface the packet is leaving the router through (if the interface is bridged, then the packet will appear to leave through the bridge interface itself)

p2p (*all-p2p* | *bit-torrent* | *blubster* | *direct-connect* | *edonkey* | *fasttrack* | *gnutella* | *soulseek* | *warez* | *winmx*) - matches packets from various peer-to-peer (P2P) protocols

packet-mark (*text*) - matches packets marked via mangle facility with particular packet mark

packet-size (*integer*: 0..65535*integer*: 0..65535) - matches packet of the specified size or size range in bytes

- **min** - specifies lower boundary of the size range or a standalone value

- **max** - specifies upper boundary of the size range

port (*port*) - matches if any (source or destination) port matches the specified list of ports or port ranges (note that the protocol must still be selected, just like for the regular src-port and dst-port matchers)

protocol (*ddp* | *egp* | *encap* | *ggp* | *gre* | *hmp* | *icmp* | *idrp-cmtp* | *igmp* | *ipencap* | *ipip* | *ipsec-ah* | *ipsec-esp* | *iso-tp4* | *ospf* | *pup* | *rdp* | *rspf* | *st* | *tcp* | *udp* | *vmtp* | *xns-idp* | *xtpinteger*) - matches particular IP protocol specified by protocol name or number. You should specify this setting if you want to specify ports

psd (*integertimeintegerinteger*) - attempts to detect TCP and UDP scans. It is advised to assign lower weight to ports with high numbers to reduce the frequency of false positives, such as from passive mode FTP transfers

- **WeightThreshold** - total weight of the latest TCP/UDP packets with different destination ports coming from the same host to be treated as port scan sequence
- **DelayThreshold** - delay for the packets with different destination ports coming from the same host to be treated as possible port scan subsequence
- **LowPortWeight** - weight of the packets with privileged (<=1024) destination port
- **HighPortWeight** - weight of the packet with non-priviliged destination port

random (*integer*: 1..99) - matches packets randomly with given propability

reject-with (*icmp-admin-prohibited* | *icmp-echo-reply* | *icmp-host-prohibited* | *icmp-host-unreachable* | *icmp-net-prohibited* | *icmp-network-unreachable* | *icmp-port-unreachable* | *icmp-protocol-unreachable* | *tcp-resetinteger*) - alters the reply packet of reject action

routing-mark (*name*) - matches packets marked by mangle facility with particular routing mark

src-address (*IP addressnetmaskIP addressIP address*) - specifies the address range an IP packet is originated from. Note that console converts entered address/netmask value to a valid network address, i.e.:1.1.1.1/24 is converted to 1.1.1.0/24

src-address-list (*name*) - matches source address of a packet against user-defined address list

src-address-type (*unicast* | *local* | *broadcast* | *multicast*) - matches source address type of the IP packet, one of the:

- **unicast** - IP addresses used for one point to another point transmission. There is only one sender and one receiver in this case
- **local** - matches addresses assigned to router's interfaces
- **broadcast** - the IP packet is sent from one point to all other points in the IP subnetwork
- **multicast** - this type of IP addressing is responsible for transmission from one or more points to a set of other points

src-mac-address (*MAC address*) - source MAC address

src-port (*integer*: 0..65535*integer*: 0..65535) - source port number or range

tcp-flags (*ack* | *cwr* | *ece* | *fin* | *psh* | *rst* | *syn* | *urg*) - tcp flags to match

- **ack** - acknowledging data
- **cwr** - congestion window reduced
- **ece** - ECN-echo flag (explicit congestion notification)
- **fin** - close connection
- **psh** - push function

- **rst** - drop connection
- **syn** - new connection
- **urg** - urgent data

tcp-mss (*integer*: 0..65535) - matches TCP MSS value of an IP packet

time (*timetimesat* | *fri* | *thu* | *wed* | *tue* | *mon* | *sun*) - allows to create filter based on the packets' arrival time and date or, for locally generated packets, departure time and date

Notes

Because the NAT rules are applied first, it is important to hold this in mind when setting up firewall rules, since the original packets might be already modified by the NAT

Filter Applications

Protect your RouterOS router

To protect your router, you should not only change admin's password but also set up packet filtering. All packets with destination to the router are processed against the ip firewall input chain. Note, that the input chain does not affect packets which are being transferred through the router.

```
/ ip firewall filter
add chain=input connection-state=invalid action=drop \
        comment="Drop Invalid connections"
add chain=input connection-state=established action=accept \
        comment="Allow Established connections"
add chain=input protocol=udp action=accept \
        comment="Allow UDP"
add chain=input protocol=icmp action=accept \
        comment="Allow ICMP"
add chain=input src-address=192.168.0.0/24 action=accept \
        comment="Allow access to router from known network"
add chain=input action=drop comment="Drop anything else"
```

Protecting the Customer's Network

To protect the customer's network, we should check all traffic which goes through router and block unwanted. For icmp, tcp, udp traffic we will create chains, where will be droped all unwanted packets:

```
/ip firewall filter
add chain=forward protocol=tcp connection-state=invalid \
         action=drop comment="drop invalid connections"
add chain=forward connection-state=established action=accept \
         comment="allow already established connections"
add chain=forward connection-state=related action=accept \
         comment="allow related connections"
```

Block IP addreses called "bogons":

```
add chain=forward src-address=0.0.0.0/8 action=drop
add chain=forward dst-address=0.0.0.0/8 action=drop
add chain=forward src-address=127.0.0.0/8 action=drop
add chain=forward dst-address=127.0.0.0/8 action=drop
add chain=forward src-address=224.0.0.0/3 action=drop
add chain=forward dst-address=224.0.0.0/3 action=drop
```

Make jumps to new chains:

```
add chain=forward protocol=tcp action=jump jump-target=tcp
add chain=forward protocol=udp action=jump jump-target=udp
add chain=forward protocol=icmp action=jump jump-target=icmp
```

Create tcp chain and deny some tcp ports in it:

```
add chain=tcp protocol=tcp dst-port=69 action=drop \
         comment="deny TFTP"
add chain=tcp protocol=tcp dst-port=111 action=drop \
         comment="deny RPC portmapper"
add chain=tcp protocol=tcp dst-port=135 action=drop \
         comment="deny RPC portmapper"
add chain=tcp protocol=tcp dst-port=137-139 action=drop \
         comment="deny NBT"
add chain=tcp protocol=tcp dst-port=445 action=drop \
         comment="deny cifs"
add chain=tcp protocol=tcp dst-port=2049 action=drop comment="deny NFS"
add chain=tcp protocol=tcp dst-port=12345-12346 action=drop comment="deny NetBus"
add chain=tcp protocol=tcp dst-port=20034 action=drop comment="deny NetBus"
add chain=tcp protocol=tcp dst-port=3133 action=drop comment="deny BackOriffice"
add chain=tcp protocol=tcp dst-port=67-68 action=drop comment="deny DHCP"
```

Deny udp ports in udp chain:

```
add chain=udp protocol=udp dst-port=69 action=drop comment="deny TFTP"
add chain=udp protocol=udp dst-port=111 action=drop comment="deny PRC portmapper"
add chain=udp protocol=udp dst-port=135 action=drop comment="deny PRC portmapper"
add chain=udp protocol=udp dst-port=137-139 action=drop comment="deny NBT"
add chain=udp protocol=udp dst-port=2049 action=drop comment="deny NFS"
add chain=udp protocol=udp dst-port=3133 action=drop comment="deny BackOriffice"
```

Allow only needed icmp codes in icmp chain:

```
add chain=icmp protocol=icmp icmp-options=0:0 action=accept \
         comment="drop invalid connections"
add chain=icmp protocol=icmp icmp-options=3:0 action=accept \
         comment="allow established connections"
add chain=icmp protocol=icmp icmp-options=3:1 action=accept \
         comment="allow already established connections"
add chain=icmp protocol=icmp icmp-options=4:0 action=accept \
         comment="allow source quench"
add chain=icmp protocol=icmp icmp-options=8:0 action=accept \
         comment="allow echo request"
add chain=icmp protocol=icmp icmp-options=11:0 action=accept \
         comment="allow time exceed"
add chain=icmp protocol=icmp icmp-options=12:0 action=accept \
         comment="allow parameter bad"
add chain=icmp action=drop comment="deny all other types"
```

Address Lists

Document revision 2.8 (February 11, 2008, 4:14 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Firewall address lists allow to create a list of IP addresses to be used for packet matching.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip firewall address-list***
Standards and Technologies: ***IP***
Hardware usage: ***Not significant***

Address Lists

Description

Firewall address lists allow user to create lists of IP addresses grouped together. Firewall filter, mangle and NAT facilities can use address lists to match packets against them.

The address list records could be updated dynamically via the **action=add-src-to-address-list** or **action=add-dst-to-address-list** items found in NAT mangle and filter facilities.

Property Description

address (*IP addressnetmaskIP addressIP address*) - specify the IP address or range to be added to the address list. Note that console converts entered address/netmask value to a valid network address, i.e.:1.1.1.1/24 is converted to 1.1.1.0/24

list (*name*) - specify the name of the address list to add IP address to

Example

The following example creates an address list of people thet are connecting to port 23 (telnet) on the router and drops all further traffic from them. Additionaly, the address list will contain one static entry of **address=192.0.34.166/32** (www.example.com):

```
[admin@MikroTik] > /ip firewall address-list add list=drop_traffic
address=192.0.34.166/32
[admin@MikroTik] > /ip firewall address-list print
Flags: X - disabled, D - dynamic
 #   LIST         ADDRESS
 0   drop_traffic 192.0.34.166
[admin@MikroTik] > /ip firewall mangle add chain=prerouting protocol=tcp dst-port=23 \
\... action=add-src-to-address-list address-list=drop_traffic
[admin@MikroTik] > /ip firewall filter add action=drop chain=input
src-address-list=drop_traffic
[admin@MikroTik] > /ip firewall address-list print
Flags: X - disabled, D - dynamic
 #   LIST         ADDRESS
 0   drop_traffic 192.0.34.166
 1 D drop_traffic 1.1.1.1
 2 D drop_traffic 10.5.11.8
[admin@MikroTik] >
```

As seen in the output of the last **print** command, two new dynamic entries appeared in the address list. Hosts with these IP addresses tried to initialize a telnet session to the router.

Mangle

Document revision .NaN (February 11, 2008, 4:14 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The mangle facility allows to mark IP packets with special marks. These marks are used by various other router facilities to identify the packets. Additionaly, the mangle facility is used to modify some fields in the IP header, like TOS (DSCP) and TTL fields.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip firewall mangle***
Standards and Technologies: ***IP***
Hardware usage: *Increases with count of mangle rules*

Mangle

Home menu level: ***/ip firewall mangle***

Description

Mangle is a kind of 'marker' that marks packets for future processing with special marks. Many other facilities in RouterOS make use of these marks, e.g. queue trees and NAT. They identify a packet based on its mark and process it accordingly. The mangle marks exist only within the router, they are not transmitted across the network.

Property Description

action (*accept* | *add-dst-to-address-list* | *add-src-to-address-list* | *change-dscp* | *change-mss* | *change-ttl* | *jump* | *log* | *mark-connection* | *mark-packet* | *mark-routing* | *passthrough* | *return* | *set-priority* | *strip-ipv4-options*; default: **accept**) - action to undertake if the packet matches the rule

- **accept** - accept the packet. No action, i.e., the packet is passed through and no more rules are applied to it
- **add-dst-to-address-list** - add destination address of an IP packet to the address list specified by address-list parameter
- **add-src-to-address-list** - add source address of an IP packet to the address list specified by address-list parameter
- **change-dscp** - change Differentiated Services Code Point (DSCP) field value specified by the new-dscp parameter
- **change-mss** - change Maximum Segment Size field value of the packet to a value specified by the new-mss parameter
- **change-ttl** - change Time to Live field value of the packet to a value specified by the new-ttl parameter
- **jump** - jump to the chain specified by the value of the jump-target parameter
- **log** - each match with this action will add a message to the system log
- **mark-connection** - place a mark specified by the new-connection-mark parameter on the entire connection that matches the rule
- **mark-packet** - place a mark specified by the new-packet-mark parameter on a packet that matches the rule
- **mark-routing** - place a mark specified by the new-routing-mark parameter on a packet. This kind of marks is used for policy routing purposes only
- **passthrough** - ignore this rule go on to the next one
- **return** - pass control back to the chain from where the jump took place
- **set-priority** - set priority speciefied by the new-priority parameter on the packets sent out through a link that is capable of transporting priority (VLAN or WMM-enabled wireless interface)
- **strip-ipv4-options** - strip IPv4 option fields from the IP packet

address-list (*name*) - specify the name of the address list to collect IP addresses from rules having action=add-dst-to-address-list or action=add-src-to-address-list actions. These address lists could be later used for packet matching

address-list-timeout (*time*; default: **00:00:00**) - time interval after which the address will be removed from the address list specified by address-list parameter. Used in conjunction with add-dst-to-address-list or add-src-to-address-list actions

- **00:00:00** - leave the address in the address list forever

chain (*forward* | *input* | *output* | *postrouting* | *prerouting*) - specify the chain to put a particular rule into. As the different traffic is passed through different chains, always be careful in choosing the right chain for a new rule. If the input does not match the name of an already defined chain, a new chain will be created

comment (*text*) - free form textual comment for the rule. A comment can be used to refer the particular rule from scripts

connection-bytes (*integerinteger*) - match packets only if a given amount of bytes has been transfered through the particular connection

- **0** - means infinity, exempli gratia: connection-bytes=2000000-0 means that the rule matches if more than 2MB has been transfered through the relevant connection

connection-limit (*integernetmask*) - restrict connection limit per address or address block

connection-mark (*name*) - match packets marked via mangle facility with particular connection mark

connection-state (*estabilished* | *invalid* | *new* | *related*) - interprets the connection tracking analysis data for a particular packet

- **estabilished** - a packet which belongs to an existing connection, exempli gratia a reply packet or a packet which belongs to already replied connection
- **invalid** - a packet which could not be identified for some reason. This includes out of memory condition and ICMP errors which do not correspond to any known connection. It is generally advised to drop these packets
- **new** - a packet which begins a new TCP connection
- **related** - a packet which is related to, but not part of an existing connection, such as ICMP errors or a packet which begins FTP data connection (the later requires enabled FTP connection tracking helper under /ip firewall service-port)

connection-type (*ftp* | *gre* | *h323* | *irc* | *mms* | *pptp* | *quake3* | *tftp*) - match packets from related connections based on information from their connection tracking helpers. A relevant connection helper must be enabled under /ip firewall service-port

content (*text*) - the text packets should contain in order to match the rule

dscp (*integer*: 0..63) - DSCP (ex-ToS) IP header field value

dst-address (*IP addressnetmaskIP addressIP address*) - specify the address range an IP packet is destined to. Note that console converts entered address/netmask value to a valid network address, i.e.:1.1.1.1/24 is converted to 1.1.1.0/24

dst-address-list (*name*) - match destination address of a packet against user-defined address list

dst-address-type (*unicast* | *local* | *broadcast* | *multicast*) - match destination address type of the IP packet, one of the:

- **unicast** - IP addresses used for one point to another point transmission. There is only one sender and one receiver in this case
- **local** - match addresses assigned to router's interfaces
- **broadcast** - the IP packet is sent from one point to all other points in the IP subnetwork
- **multicast** - this type of IP addressing is responsible for transmission from one or more points to a set of other points

dst-limit (*integertimeintegerdst-address* | *dst-port* | *src-addresstime*) - limit the packet per second (pps) rate on a per destination IP or per destination port base. As opposed to the limit match, every destination IP address / destination port has it's own limit. The options are as follows (in order of appearance):

- **count** - maximum average packet rate, measured in packets per second (pps), unless followed by time option
- **time** - specifies the time interval over which the packet rate is measured
- **burst** - number of packets to match in a burst
- **mode** - the classifier(-s) for packet rate limiting

- **expire** - specifies interval after which recorded IP addresses / ports will be deleted

dst-port (*integer*: 0..65535*integer*: 0..65535) - destination port number or range

fragment (yes | no) - whether the packet is a fragment of an IP packet. Starting packet (i.e., first fragment) does not count. Note that is the connection tracking is enabled, there will be no fragments as the system automatically assembles every packet

hotspot (*multiple choice: auth | from-client | http | local-dst | to-client*) - matches packets received from clients against various HotSpot conditions. All values can be negated

- **auth** - true, if a packet comes from an authenticted HotSpotclient
- **from-client** - true, if a packet comes from any HotSpot client
- **http** - true, if a HotSpot client sends a packet to the address and port previously detected as his proxy server (Universal Proxy technique) or if the destination port is 80 and transparent proxying is enabled for that particular client
- **local-dst** - true, if a packet has local destination IP address
- **to-client** - true, if a packet is sent to a client

icmp-options (*integerinteger*) - match ICMP Type:Code fields

in-bridge-port (*name*) - actual interface the packet has entered the router through (if bridged, this property matches the actual bridge port, while in-interface - the bridge itself)

in-interface (*name*) - interface the packet has entered the router through (if the interface is bridged, then the packet will appear to come from the bridge interface itself)

ingress-priority (*integer*: 0..63) - INGRESS (received) priority of the packet, if set (0 otherwise). The priority may be derived from either VLAN or WMM priority

ipv4-options (*any* | *loose-source-routing* | *no-record-route* | *no-router-alert* | *no-source-routing* | *no-timestamp* | *none* | *record-route* | *router-alert* | *strict-source-routing* | *timestamp*) - match ipv4 header options

- **any** - match packet with at least one of the ipv4 options
- **loose-source-routing** - match packets with loose source routing option. This option is used to route the internet datagram based on information supplied by the source
- **no-record-route** - match packets with no record route option. This option is used to route the internet datagram based on information supplied by the source
- **no-router-alert** - match packets with no router alter option
- **no-source-routing** - match packets with no source routing option
- **no-timestamp** - match packets with no timestamp option
- **record-route** - match packets with record route option
- **router-alert** - match packets with router alter option
- **strict-source-routing** - match packets with strict source routing option
- **timestamp** - match packets with timestamp

jump-target (*forward* | *input* | *output* | *postrouting* | *preroutingname*) - name of the target chain to jump to, if the action=jump is used

layer7-protocol (*name*) - Layer 7 filter name as set in the /ip firewall layer7-protocol menu. Caution: this matcher needs high computational power

limit (*integertimeinteger*) - restrict packet match rate to a given limit. Usefull to reduce the amount of log messages

- **count** - maximum average packet rate, measured in packets per second (pps), unless followed by time option
- **time** - specify the time interval over which the packet rate is measured
- **burst** - number of packets to match in a burst

log-prefix (*text*) - all messages written to logs will contain the prefix specified herein. Used in conjunction with action=log

new-connection-mark (*name*) - specify the new value of the connection mark to be used in conjunction with action=mark-connection

new-dscp (*integer*: 0..63) - specify the new value of the DSCP field to be used in conjunction with action=change-dscp

new-mss (*integer*) - specify MSS value to be used in conjunction with action=change-mss

new-packet-mark (*name*) - specify the new value of the packet mark to be used in conjunction with action=mark-packet

new-priority (*integer*) - specify the new value of packet priority for the priority-enabled interfaces, used in conjunction with action=set-priority

- **from-dscp** - set packet priority form its DSCP field value
- **from-ingress** - set packet priority from the INGRESS priority of the packet (in case packet has been received from an interface that supports priorities - VLAN or WMM-enabled wireless interface; 0 if not set)

new-routing-mark (*name*) - specify the new value of the routing mark used in conjunction with action=mark-routing

new-ttl (*decrement* | *increment* | *setinteger*) - specify the new TTL field value used in conjunction with action=change-ttl

- **decrement** - the value of the TTL field will be decremented for value
- **increment** - the value of the TTL field will be incremented for value
- **set:** - the value of the TTL field will be set to value

nth (*integerinteger*: 0..15*integer*) - match a particular Nth packet received by the rule. One of 16 available counters can be used to count packets

- **every** - match every every+1th packet. For example, if every=1 then the rule matches every 2nd packet
- **counter** - specifies which counter to use. A counter increments each time the rule containing nth match matches
- **packet** - match on the given packet number. The value by obvious reasons must be between 0 and every. If this option is used for a given counter, then there must be at least every+1 rules with this option, covering all values between 0 and every inclusively.

out-bridge-port (*name*) - actual interface the packet is leaving the router through (if bridged, this property matches the actual bridge port, while out-interface - the bridge itself)

out-interface (*name*) - interface the packet is leaving the router through (if the interface is bridged, then the packet will appear to leave through the bridge interface itself)

p2p (*all-p2p* | *bit-torrent* | *direct-connect* | *edonkey* | *fasttrack* | *gnutella* | *soulseek* | *warez* | *winmx*) - match packets belonging to connections of the above P2P protocols

packet-mark (*name*) - match the packets marked in mangle with specific packet mark

packet-size (*integer*: 0..65535*integer*: 0..65535) - matches packet of the specified size or size range in bytes

- **min** - specifies lower boundary of the size range or a standalone value
- **max** - specifies upper boundary of the size range

passthrough (yes | no; default: **yes**) - whether to let the packet to pass further (like action passthrough) after marking it with a given mark (property only valid if action is mark packet, connection or routing mark)

port (*port*) - matches if any (source or destination) port matches the specified list of ports or port ranges (note that the protocol must still be selected, just like for the regular src-port and dst-port matchers)

protocol (*ddp* | *egp* | *encap* | *ggp* | *gre* | *hmp* | *icmp* | *idrp-cmtp* | *igmp* | *ipencap* | *ipip* | *ipsec-ah* | *ipsec-esp* | *iso-tp4* | *ospf* | *pup* | *rdp* | *rspf* | *st* | *tcp* | *udp* | *vmtp* | *xns-idp* | *xtpinteger*) - matches particular IP protocol specified by protocol name or number. You should specify this setting if you want to specify ports

psd (*integertimeintegerinteger*) - attempts to detect TCP and UDP scans. It is advised to assign lower weight to ports with high numbers to reduce the frequency of false positives, such as from passive mode FTP transfers

- **WeightThreshold** - total weight of the latest TCP/UDP packets with different destination ports coming from the same host to be treated as port scan sequence
- **DelayThreshold** - delay for the packets with different destination ports coming from the same host to be treated as possible port scan subsequence
- **LowPortWeight** - weight of the packets with privileged (<=1024) destination port
- **HighPortWeight** - weight of the packet with non-priviliged destination port

random (*integer*: 1..99) - matches packets randomly with given propability

routing-mark (*name*) - matches packets marked with the specified routing mark

src-address (*IP addressnetmaskIP addressIP address*) - specifies the address range an IP packet is originated from. Note that console converts entered address/netmask value to a valid network address, i.e.:1.1.1.1/24 is converted to 1.1.1.0/24

src-address-list (*name*) - matches source address of a packet against user-defined address list

src-address-type (*unicast* | *local* | *broadcast* | *multicast*) - matches source address type of the IP packet, one of the:

- **unicast** - IP addresses used for one point to another point transmission. There is only one sender and one receiver in this case
- **local** - matches addresses assigned to router's interfaces
- **broadcast** - the IP packet is sent from one point to all other points in the IP subnetwork
- **multicast** - this type of IP addressing is responsible for transmission from one or more points to a set of other points

src-mac-address (*MAC address*) - source MAC address

src-port (*integer*: 0..65535*integer*: 0..65535) - source port number or range

tcp-flags (*multiple choice: ack* | *cwr* | *ece* | *fin* | *psh* | *rst* | *syn* | *urg*) - tcp flags to match

- **ack** - acknowledging data
- **cwr** - congestion window reduced

- **ece** - ECN-echo flag (explicit congestion notification)
- **fin** - close connection
- **psh** - push function
- **rst** - drop connection
- **syn** - new connection
- **urg** - urgent data

tcp-mss (*integer*: 0..65535) - matches TCP MSS value of an IP packet

time (*timetimesat* | *fri* | *thu* | *wed* | *tue* | *mon* | *sun*) - allows to create filter based on the packets' arrival time and date or, for locally generated packets, departure time and date

Notes

Instead of making two rules if you want to mark a packet, connection or routing-mark and finish mangle table processing on that event (in other words, mark and simultaneously accept the packet), you may disable the set by default **passthrough** property of the marking rule.

Usually routing-mark is not used for P2P, since P2P traffic always is routed over a default getaway.

Application Examples

Description

The following section discusses some examples of using the mangle facility.

Peer-to-Peer Traffic Marking

To ensure the quality of service for network connection, interactive traffic types such as VoIP and HTTP should be prioritized over non-interactive, such as peer-to-peer network traffic. RouterOS QOS implementation uses mangle to mark different types of traffic first, and then place them into queues with different limits.

The following example enforces the P2P traffic will get no more than 1Mbps of the total link capacity when the link is heavily used by other traffic otherwice expanding to the full link capacity:

```
[admin@MikroTik] > /ip firewall mangle add chain=forward \
\... p2p=all-p2p action=mark-connection new-connection-mark=p2p_conn
[admin@MikroTik] > /ip firewall mangle add chain=forward \
\... connection-mark=p2p_conn action=mark-packet new-packet-mark=p2p
[admin@MikroTik] > /ip firewall mangle add chain=forward \
\... connection-mark=!p2p_conn action=mark-packet new-packet-mark=other
[admin@MikroTik] > /ip firewall mangle print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=forward p2p=all-p2p action=mark-connection new-connection-mark=p2p_conn

 1   chain=forward connection-mark=p2p_conn action=mark-packet new-packet-mark=p2p

 2   chain=forward packet-mark=!p2p_conn action=mark-packet new-packet-mark=other
[admin@MikroTik] >
[admin@MikroTik] > /queue tree add parent=Public packet-mark=p2p limit-at=1000000 \
\... max-limit=100000000 priority=8
[admin@MikroTik] > /queue tree add parent=Local packet-mark=p2p limit-at=1000000 \
\... max-limit=100000000 priority=8
[admin@MikroTik] > /queue tree add parent=Public packet-mark=other  limit-at=1000000 \
\... max-limit=100000000 priority=1
[admin@MikroTik] > /queue tree add parent=Local packet-mark=other  limit-at=1000000 \
\... max-limit=100000000 priority=1
```

Mark by MAC address

To mark traffic from a known MAC address which goes to the router or through it, do the following:

```
[admin@MikroTik] > / ip firewall mangle add chain=prerouting \
\... src-mac-address=00:01:29:60:36:E7 action=mark-connection
new-connection-mark=known_mac_conn
[admin@MikroTik] > / ip firewall mangle add chain=prerouting \
\... connection-mark=known_mac_conn action=mark-packet new-packet-mark=known_mac
```

Change MSS

It is a well known fact that VPN links have smaller packet size due to incapsulation overhead. A large packet with MSS that exceeds the MSS of the VPN link should be fragmented prior to sending it via that kind of connection. However, if the packet has DF flag set, it cannot be fragmented and should be discarded. On links that have broken path MTU discovery (PMTUD) it may lead to a number of problems, including problems with FTP and HTTP data transfer and e-mail services.

In case of link with broken PMTUD, a decrease of the MSS of the packets coming through the VPN link solves the problem. The following example demonstrates how to decrease the MSS value via mangle:

```
[admin@MikroTik] > /ip firewall mangle add out-interface=pppoe-out \
\... protocol=tcp tcp-flags=syn action=change-mss new-mss=1300 chain=forward
[admin@MikroTik] > /ip firewall mangle print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=forward out-interface=pppoe-out protocol=tcp tcp-flags=syn
     action=change-mss new-mss=1300

[admin@MikroTik] >
```

NAT

Document revision 2.9 (February 11, 2008, 4:14 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Network Address Translation (NAT) is a router facility that replaces source and (or) destination IP addresses of the IP packet as it pass through thhe router. It is most commonly used to enable multiple host on a private network to access the Internet using a single public IP address.

Specifications

Packages required: ***system***
License required: ***level1 (number of rules limited to 1), level3***
Home menu level: ***/ip firewall nat***
Standards and Technologies: ***IP***, ***RFC1631***, ***RFC2663***
Hardware usage: *Increases with the count of rules*

NAT

Description

Network Address Translation is an Internet standard that allows hosts on local area networks to use one set of IP addresses for internal communications and another set of IP addresses for external communications. A LAN that uses NAT is referred as *natted* network. For NAT to function, there should be a NAT gateway in each natted network. The NAT gateway (NAT router) performs IP address rewriting on the way a packet travel from/to LAN.

There are two types of NAT:

- source NAT or srcnat. This type of NAT is performed on packets that are originated from a natted network. A NAT router replaces the private source address of an IP packet with a new public IP address as it travels through the router. A reverse operation is applied to the reply packets travelling in the other direction.
- destination NAT or dstnat. This type of NAT is performed on packets that are destined to the natted network. It is most comonly used to make hosts on a private network to be acceesible from the Internet. A NAT router performing dstnat replaces the destination IP address of an IP packet as it travel through the router towards a private network.

NAT Drawbacks

Hosts behind a NAT-enabled router do not have true end-to-end connectivity. Therefore some Internet protocols might not work in scenarios with NAT. Services that require the initiation of TCP connection from outside the private network or stateless protocols such as UDP, can be disrupted. Moreover, some protocols are inherently incompatible with NAT, a bold example is AH protocol from the IPsec suite.

RouterOS includes a number of so-called NAT helpers, that enable NAT traversal for various protocols.

Redirect and Masquerade

Redirect and masquerade are special forms of destination NAT and source NAT, respectively. Redirect is similar to the regular destination NAT in the same way as masquerade is similar to the source NAT - masquerade is a special form of source NAT without need to specify **to-addresses** - outgoing interface address is used automatically. The same is for redirect - it is a form of destination NAT where **to-addresses** is not used - incoming interface address is used instead. Note that **to-ports** is meaningful for redirect rules - this is the port of the service on the router that will handle these requests (e.g. web proxy).

When packet is dst-natted (no matter - **action=nat** or **action=redirect**), dst address is changed. Information about translation of addresses (including original dst address) is kept in router's internal tables. Transparent web proxy working on router (when web requests get redirected to proxy port on router) can access this information from internal tables and get address of web server from them. If you are dst-natting to some different proxy server, it has no way to find web server's address from IP header (because dst address of IP packet that previously was address of web server has changed to address of proxy server). Starting from HTTP/1.1 there is special header in HTTP request which tells web server address, so proxy server can use it, instead of dst address of IP packet. If there is no such header (older HTTP version on client), proxy server can not determine web server address and therefore can not work.

It means, that it is impossible to correctly transparently redirect HTTP traffic from router to some other transparent-proxy box. Only correct way is to add transparent proxy on the router itself, and configure it so that your "real" proxy is its parent-proxy. In this situation your "real" proxy does not have to be transparent any more, as proxy on router will be transparent and will forward proxy-style requests (according to standard; these requests include all necessary information about web server) to "real" proxy.

Property Description

action (*accept* | *add-dst-to-address-list* | *add-src-to-address-list* | *dst-nat* | *jump* | *log* | *masquerade* | *netmap* | *passthrough* | *redirect* | *return* | *same* | *src-nat*; default: **accept**) - action to undertake if the packet matches the rule

- **accept** - accepts the packet. No action is taken, i.e. the packet is passed through and no more rules are applied to it
- **add-dst-to-address-list** - adds destination address of an IP packet to the address list specified by address-list parameter
- **add-src-to-address-list** - adds source address of an IP packet to the address list specified by address-list parameter
- **dst-nat** - replaces destination address of an IP packet to values specified by to-addresses and to-ports parameters
- **jump** - jump to the chain specified by the value of the jump-target parameter
- **log** - each match with this action will add a message to the system log
- **masquerade** - replaces source address of an IP packet to an automatically determined by the routing facility IP address
- **netmap** - creates a static 1:1 mapping of one set of IP addresses to another one. Often used to distribute public IP addresses to hosts on private networks
- **passthrough** - ignores this rule goes on to the next one
- **redirect** - replaces destination address of an IP packet to one of the router's local addresses
- **return** - passes control back to the chain from where the jump took place
- **same** - gives a particular client the same source/destination IP address from supplied range for each connection. This is most frequently used for services that expect the same client address for multiple connections from the same client
- **src-nat** - replaces source address of an IP packet to values specified by to-addresses and to-ports parameters

address-list (*name*) - specifies the name of the address list to collect IP addresses from rules having action=add-dst-to-address-list or action=add-src-to-address-list actions. These address lists could be later used for packet matching

address-list-timeout (*time*; default: **00:00:00**) - time interval after which the address will be removed from the address list specified by address-list parameter. Used in conjunction with add-dst-to-address-list or add-src-to-address-list actions

- **00:00:00** - leave the address in the address list forever

chain (*dstnat* | *srcnatname*) - specifies the chain to put a particular rule into. As the different traffic is passed through different chains, always be careful in choosing the right chain for a new rule. If the input does not match the name of an already defined chain, a new chain will be created

- **dstnat** - a rule placed in this chain is applied before routing. The rules that replace destination addresses of IP packets should be placed there
- **srcnat** - a rule placed in this chain is applied after routing. The rules that replace the source addresses of IP packets should be placed there

comment (*text*) - a descriptive comment for the rule. A comment can be used to identify rules form scripts

connection-bytes (*integerinteger*) - matches packets only if a given amount of bytes has already been transfered through the particular connection

- **0** - means infinity, exempli gratia: connection-bytes=2000000-0 means that the rule matches if more than 2MB has been transfered through the relevant connection

connection-limit (*integernetmask*) - restrict connection number per address or address block

(matches if the specified number of connection has already been established)

connection-mark (*name*) - matches packets marked via mangle facility with particular connection mark

connection-type (*ftp* | *gre* | *h323* | *irc* | *mms* | *pptp* | *quake3* | *tftp*) - matches packets from related connections based on information from their connection tracking helpers. A relevant connection helper must be enabled under /ip firewall service-port

content (*text*) - the text packets should contain in order to match the rule

dscp (*integer*: 0..63) - DSCP (ex-ToS) IP header field value

dst-address (*IP addressnetmaskIP addressIP address*) - specifies the address range an IP packet is destined to. Note that console converts entered address/netmask value to a valid network address, i.e.:1.1.1.1/24 is converted to 1.1.1.0/24

dst-address-list (*name*) - matches destination address of a packet against user-defined address list

dst-address-type (*unicast* | *local* | *broadcast* | *multicast*) - matches destination address type of the IP packet, one of the:

- **unicast** - IP addresses used for one point to another point transmission. There is only one sender and one receiver in this case
- **local** - matches addresses assigned to router's interfaces
- **broadcast** - the IP packet is sent from one point to all other points in the IP subnetwork
- **multicast** - this type of IP addressing is responsible for transmission from one or more points to a set of other points

dst-limit (*integertimeintegerdst-address* | *dst-port* | *src-addresstime*) - limits the packet per second (pps) rate on a per destination IP or per destination port base. As opposed to the limit match, every destination IP address / destination port has it's own limit. The options are as follows (in order of appearance):

- **count** - maximum average packet rate, measured in packets per second (pps), unless followed by time option
- **time** - specifies the time interval over which the packet rate is measured
- **burst** - number of packets to match in a burst
- **mode** - the classifier(-s) for packet rate limiting
- **expire** - specifies interval after which recorded IP addresses / ports will be deleted

dst-port (*integer*: 0..65535*integer*: 0..65535) - destination port number or range

fragment (yes | no) - whether the packet is a fragment of an IP packet. Starting packet (i.e., first fragment) does not count. Note that is the connection tracking is enabled, there will be no fragments as the system automatically assembles every packet

hotspot (*multiple choice: auth* | *from-client* | *http* | *local-dst* | *to-client*) - matches packets received from clients against various HotSpot conditions. All values can be negated

- **auth** - true, if a packet comes from an authenticted HotSpotclient
- **from-client** - true, if a packet comes from any HotSpot client
- **http** - true, if a HotSpot client sends a packet to the address and port previously detected as his proxy server (Universal Proxy technique) or if the destination port is 80 and transparent proxying is enabled for that particular client
- **local-dst** - true, if a packet has local destination IP address

- **to-client** - true, if a packet is sent to a client

icmp-options (*integerinteger*) - matches ICMP Type:Code fields

in-bridge-port (*name*) - actual interface the packet has entered the router through (if bridged, this property matches the actual bridge port, while in-interface - the bridge itself)

in-interface (*name*) - interface the packet has entered the router through (if the interface is bridged, then the packet will appear to come from the bridge interface itself)

ingress-priority (*integer*: 0..63) - INGRESS (received) priority of the packet, if set (0 otherwise). The priority may be derived from either VLAN or WMM priority

ipv4-options (*any* | *loose-source-routing* | *no-record-route* | *no-router-alert* | *no-source-routing* | *no-timestamp* | *none* | *record-route* | *router-alert* | *strict-source-routing* | *timestamp*) - match ipv4 header options

- **any** - match packet with at least one of the ipv4 options
- **loose-source-routing** - match packets with loose source routing option. This option is used to route the internet datagram based on information supplied by the source
- **no-record-route** - match packets with no record route option. This option is used to route the internet datagram based on information supplied by the source
- **no-router-alert** - match packets with no router alter option
- **no-source-routing** - match packets with no source routing option
- **no-timestamp** - match packets with no timestamp option
- **record-route** - match packets with record route option
- **router-alert** - match packets with router alter option
- **strict-source-routing** - match packets with strict source routing option
- **timestamp** - match packets with timestamp

jump-target (*dstnat* | *srcnatname*) - name of the target chain to jump to, if the action=jump is used

layer7-protocol (*name*) - Layer 7 filter name as set in the /ip firewall layer7-protocol menu. Caution: this matcher needs high computational power

limit (*integertimeinteger*) - restricts packet match rate to a given limit. Usefull to reduce the amount of log messages

- **count** - maximum average packet rate, measured in packets per second (pps), unless followed by time option
- **time** - specifies the time interval over which the packet rate is measured
- **burst** - number of packets to match in a burst

log-prefix (*text*) - all messages written to logs will contain the prefix specified herein. Used in conjunction with action=log

nth (*integerinteger*: 0..15*integer*) - match a particular Nth packet received by the rule. One of 16 available counters can be used to count packets

- **every** - match every every+1th packet. For example, if every=1 then the rule matches every 2nd packet
- **counter** - specifies which counter to use. A counter increments each time the rule containing nth match matches
- **packet** - match on the given packet number. The value by obvious reasons must be between 0 and every. If this option is used for a given counter, then there must be at least every+1 rules

with this option, covering all values between 0 and every inclusively.

out-bridge-port (*name*) - actual interface the packet is leaving the router through (if bridged, this property matches the actual bridge port, while out-interface - the bridge itself)

out-interface (*name*) - interface the packet is leaving the router through (if the interface is bridged, then the packet will appear to leave through the bridge interface itself)

packet-mark (*text*) - matches packets marked via mangle facility with particular packet mark

packet-size (*integer*: 0..65535*integer*: 0..65535) - matches packet of the specified size or size range in bytes

- **min** - specifies lower boundary of the size range or a standalone value
- **max** - specifies upper boundary of the size range

port (*port*) - matches if any (source or destination) port matches the specified list of ports or port ranges (note that the protocol must still be selected, just like for the regular src-port and dst-port matchers)

protocol (*ddp* | *egp* | *encap* | *ggp* | *gre* | *hmp* | *icmp* | *idrp-cmtp* | *igmp* | *ipencap* | *ipip* | *ipsec-ah* | *ipsec-esp* | *iso-tp4* | *ospf* | *pup* | *rdp* | *rspf* | *st* | *tcp* | *udp* | *vmtp* | *xns-idp* | *xtpinteger*) - matches particular IP protocol specified by protocol name or number. You should specify this setting if you want to specify ports

psd (*integertimeintegerinteger*) - attempts to detect TCP and UDP scans. It is advised to assign lower weight to ports with high numbers to reduce the frequency of false positives, such as from passive mode FTP transfers

- **WeightThreshold** - total weight of the latest TCP/UDP packets with different destination ports coming from the same host to be treated as port scan sequence
- **DelayThreshold** - delay for the packets with different destination ports coming from the same host to be treated as possible port scan subsequence
- **LowPortWeight** - weight of the packets with privileged (<=1024) destination port
- **HighPortWeight** - weight of the packet with non-priviliged destination port

random (*integer*) - match packets randomly with given propability

routing-mark (*name*) - matches packets marked by mangle facility with particular routing mark

same-not-by-dst (*yes* | *no*) - specifies whether to account or not to account for destination IP address when selecting a new source IP address for packets matched by rules with action=same

src-address (*IP addressnetmaskIP addressIP address*) - specifies the address range an IP packet is originated from. Note that console converts entered address/netmask value to a valid network address, i.e.:1.1.1.1/24 is converted to 1.1.1.0/24

src-address-list (*name*) - matches source address of a packet against user-defined address list

src-address-type (*unicast* | *local* | *broadcast* | *multicast*) - matches source address type of the IP packet, one of the:

- **unicast** - IP addresses used for one point to another point transmission. There is only one sender and one receiver in this case
- **local** - matches addresses assigned to router's interfaces
- **broadcast** - the IP packet is sent from one point to all other points in the IP subnetwork
- **multicast** - this type of IP addressing is responsible for transmission from one or more points to a set of other points

src-mac-address (*MAC address*) - source MAC address

src-port (*integer*: 0..65535*integer*: 0..65535) - source port number or range

tcp-mss (*integer*: 0..65535) - matches TCP MSS value of an IP packet

time (*timetimesat* | *fri* | *thu* | *wed* | *tue* | *mon* | *sun*) - allows to create filter based on the packets' arrival time and date or, for locally generated packets, departure time and date

to-addresses (*IP addressIP address*; default: **0.0.0.0**) - address or address range to replace original address of an IP packet with

to-ports (*integer*: 0..65535*integer*: 0..65535) - port or port range to replace original port of an IP packet with

NAT Applications

Description

In this section some NAT applications and examples of them are discussed.

Basic NAT configuration

Assume we want to create router that:

- "hides" the private LAN "behind" one address
- provides Public IP to the Local server
- creates 1:1 mapping of network addresses

Example of Source NAT (Masquerading)

If you want to "hide" the private LAN 192.168.0.0/24 "behind" one address 10.5.8.109 given to you by the ISP, you should use the source network address translation (masquerading) feature of the MikroTik router. The masquerading will change the source IP address and port of the packets originated from the network 192.168.0.0/24 to the address 10.5.8.109 of the router when the packet is routed through it.

To use masquerading, a source NAT rule with action 'masquerade' should be added to the firewall configuration:

```
/ip firewall nat add chain=srcnat action=masquerade out-interface=Public
```

All outgoing connections from the network 192.168.0.0/24 will have source address 10.5.8.109 of the router and source port above 1024. No access from the Internet will be possible to the Local addresses. If you want to allow connections to the server on the local network, you should use destination Network Address Translation (NAT).

Example of Destination NAT

If you want to link Public IP 10.5.8.200 address to Local one 192.168.0.109, you should use destination address translation feature of the MikroTik router. Also if you want allow Local server to talk with outside with given Public IP you should use source address translation, too

Add Public IP to Public interface:

```
/ip address add address=10.5.8.200/32 interface=Public
```

Add rule allowing access to the internal server from external networks:

```
/ip firewall nat add chain=dstnat dst-address=10.5.8.200 action=dst-nat \
        to-addresses=192.168.0.109
```

Add rule allowing the internal server to talk to the outer networks having its source address translated to 10.5.8.200:

```
/ip firewall nat add chain=srcnat src-address=192.168.0.109 action=src-nat \
        to-addresses=10.5.8.200
```

Example of 1:1 mapping

If you want to link Public IP subnet 11.11.11.0/24 to local one 2.2.2.0/24, you should use destination address translation and source address translation features with **action=netmap**.

```
/ip firewall nat add chain=dstnat dst-address=11.11.11.1-11.11.11.254 \
        action=netmap to-addresses=2.2.2.1-2.2.2.254

/ip firewall nat add chain=srcnat src-address=2.2.2.1-2.2.2.254 \
        action=netmap to-addresses=11.11.11.1-11.11.11.254
```

Packet Flow

Document revision 2.8 (February 11, 2008, 4:14 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

This manual describes the order in which an IP packet traverses various internal facilities of the router and some general information regarding packet handling, common IP protocols and protocol options.

Specifications

Packages required: ***system***
License required: ***level3***
Home menu level: ***/ip firewall***
Standards and Technologies: ***IP***
Hardware usage: *Increases with NAT, mangle and filter rules count*

Packet Flow

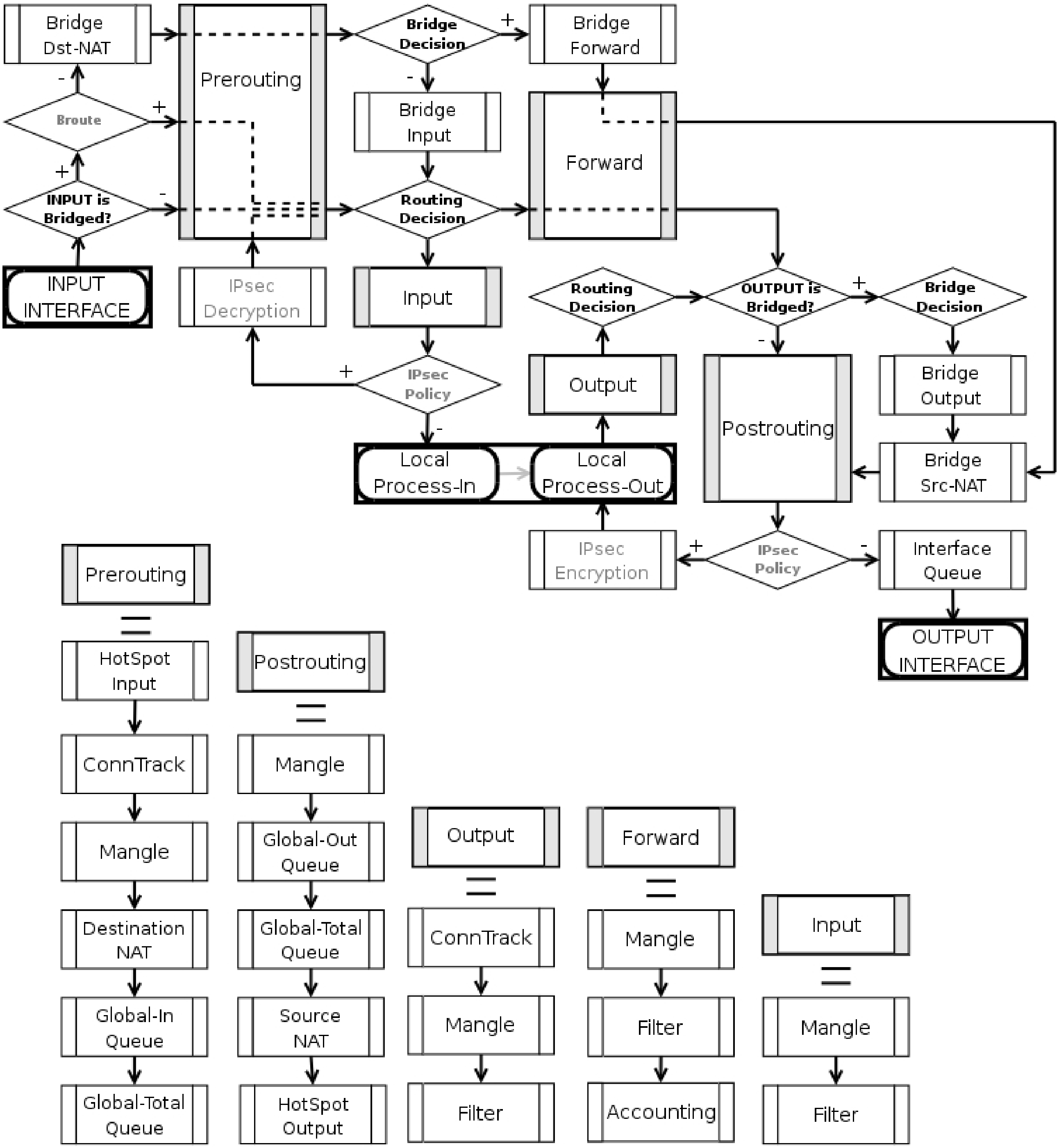
Description

MikroTik RouterOS is designed to be easy to operate in various aspects, including IP firewall. Therefore regular firewall policies can be created and deployed without the knowledge about how the packets are processed in the router. For example, if all that required is just natting internal clients to a public address, the following command can be issued (assuming the interface to the Internet is named **Public**):

```
/ip firewall nat add action=masquerade out-interface=Public chain=srcnat
```

Regular packet filtering, bandwith management or packet marking can be configured with ease in a similar manner. However, a more complicated configuration could be deployed only with a good understanding of the underlying processes in the router.

The packet flow through the router is depicted in the following diagram:



As can be seen on the diagram, there are five chains in the processing pipeline. These are **prerouting**,

input, **forward**, **output** and **postrouting**. The actions performed on a packet in each chain are discussed later in this chapter.

Additional arrows from IPsec boxes shows the processing of encrypted packets (they need to be encrypted / decrypted first and then processed as usual, *id est* from the point an ordinal packet enters the router).

A packet can enter processing conveyer of the router in two ways. First, a packet can come from one of the interfaces present in the roter (then the interface is referred as **input interface**). Second, it can be originated from a local process, like web proxy, VPN or others. Alike, there are two ways for a packet to leave the processing pipeline. A packet can leave through the one of the router's interfaces (in this case the interface is referred as **output interface**) or it can end up in the local process. In general, traffic can be destined to one of the router's IP addresses, it can originate from the router or simply should be passed through. To further complicate things the traffic can be bridged or routed one, which is determined during the **Bridge Decision** stage.

Routed traffic

The traffic received for the router's MAC address on the respective port, is passed to the routing procedures and can be of one of these four types:

- the traffic which is destined to the router itself. The IP packets has destination address equal to one of the router's own IP addresses. A packet enters the router through the **input interface**, sequentially traverses **prerouting** and **input** chains and ends up in the local process that listens for that particular kind of traffic (if no process is expecting the packet, it is discarded). Consequently, a packet can be filtered in the **input** chain filter and mangled in two places: the **input** and the **prerouting** chain filters.
- the traffic is originated from the router. In this case the IP packets have their source addresses identical to one of the router's IP addresses. If no address is assumed by the sender (either explicitly set by the sending process or, as a reply, is set to the address the request came to), the actual source address is set by the routing process to the preferred address of the respective route. Such packets travel through the **output** chain, then they are passed to the routing facility where an appropriate routing path for each packet is determined and leave through the **postrouting** chain.
- routable traffic, which is received at the router's MAC address, has an IP address different from any of the router's own addresses, and its destination can be found in the routing tables. These packets go through the **prerouting**, **forward** and **postrouting** chains.
- unroutable traffic, which is received at the router's MAC address, has an IP address different from any of the router's own addresses, but its destination can not be found in the routing tables. These packets go through the **prerouting** and stop in the **routing recision**.

The actions imposed by various router facilities are sequentially applied to a packet in each of the default chains. The exact order they are applied is pictured at the bottom of the flow diagram. *Exempli gratia*, for a packet passing **postrouting** chain the mangle rules are applied first, two types of queuing come in second place and finally source NAT is performed on packets that need to be natted.

Note, that any given packet can come through only one of the **input**, **forward** or **output** chains.

Bridged Traffic

In case the incoming traffic needs to be bridged (do not confuse it with the traffic coming to the bridge interface at the router's own MAC address and, thus, classified as routed traffic) it is first determined whether it is an IP traffic or not. After that, IP traffic goes through the **prerouting**, **forward** and **postrouting** chains, while non-IP traffic bypasses all IP firewall rules and goes directly to the interface queue. Both types of traffic, however, undergo the full set of bridge firewall chains anyway, regardless of the protocol.

Connection Tracking

Home menu level: ***/ip firewall connection***

Description

Connection tracking refers to the ability to maintain the state information about connections, such as source and destination IP address and ports pairs, connection states, protocol types and timeouts. Firewalls that do connection tracking are known as "stateful" and are inherently more secure that those who do only simple "stateless" packet processing.

The *state* of a particular connection could be **estabilished** meaning that the packet is part of already known connection, **new** meaning that the packet starts a new connection or belongs to a connection that has not seen packets in both directions yet, **related** meaning that the packet starts a new connection, but is associated with an existing connection, such as FTP data transfer or ICMP error message and, finally, **invalid** meaning that the packet does not belong to any known connection and, at the same time, does not open a valid new connection.

Connection tracking is done in the **prerouting** chain, or the **output** chain for locally generated packets.

Another function of connection tracking which cannot be overestimated is that it is needed for NAT. You should be aware that no NAT can be performed unless you have connection tracking enabled, the same applies for p2p protocols recognition. Connection tracking also assembles IP packets from fragments before further processing.

The maximum number of connections the **/ip firewall connection** state table can contain is determined by the amount of physical memory present in the router.

Please ensure that your router is equipped with sufficient amount of physical memory to properly handle all connections.

Property Description

assured (*read-only: true | false*) - shows whether replay was seen for the last packet matching this entry

connection-mark (*read-only: text*) - Connection mark set in mangle

dst-address (*read-only: IP addressport*) - the destination address and port the connection is established to

icmp-id (*read-only: integer*) - contains the ICMP ID. Each ICMP packet gets an ID set to it when it is sent, and when the receiver gets the ICMP message, it sets the same ID within the new ICMP message so that the sender will recognize the reply and will be able to connect it with the appropriate ICMP request

icmp-option (*read-only: integer*) - the ICMP type and code fields

p2p (*read-only: text*) - peer to peer protocol

protocol (*read-only: text*) - IP protocol name or number

reply-dst-address (*read-only: IP addressport*) - the destination address and port the reply connection is established to

reply-icmp-id (*read-only: integer*) - contains the ICMP ID of received packet

reply-icmp-option (*read-only: integer*) - the ICMP type and code fields of received packet

reply-src-address (*read-only: IP addressport*) - the source address and port the reply connection is established from

src-address (*read-only: IP addressport*) - the source address and port the connection is established from

tcp-state (*read-only: text*) - the state of TCP connection

timeout (*read-only: time*) - the amount of time until the connection will be timed out

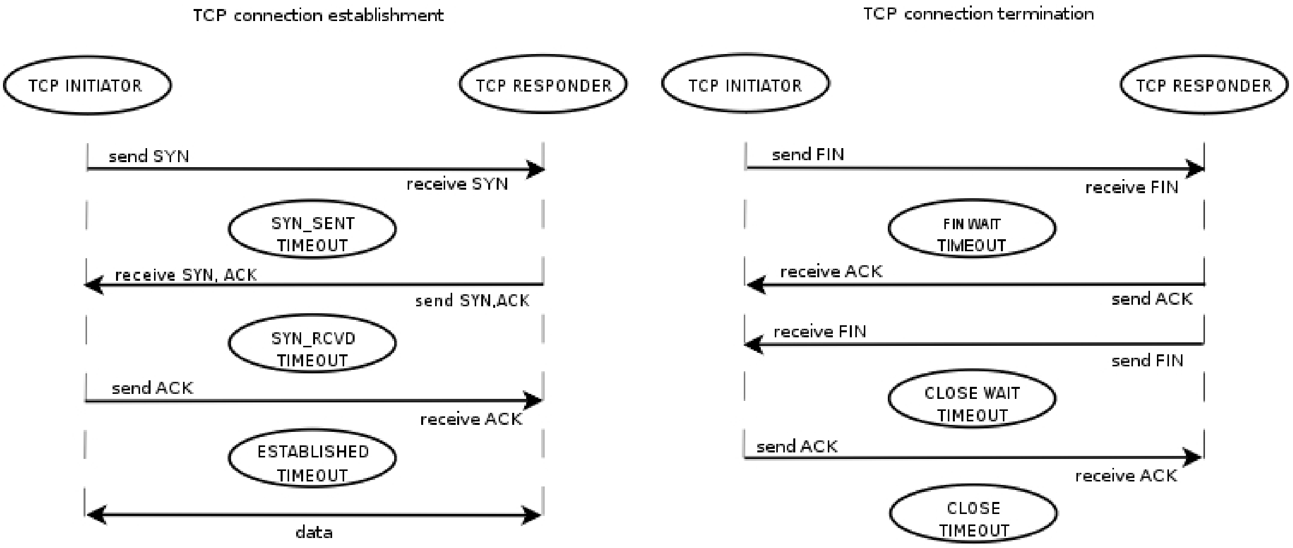
unreplied (*read-only: true | false*) - shows whether the request was unreplied

Connection Timeouts

Home menu level: ***/ip firewall connection tracking***

Description

Connection tracking provides several timeouts. When particular timeout expires the according entry is removed from the connection state table. The following diagram depicts typical TCP connection establishment and termination and tcp timeouts that take place during these processes:



Property Description

enable (*yes | no*; default: **yes**) - whether to allow or disallow connection tracking

generic-timeout (*time*; default: **10m**) - maximal amount of time connection state table entry that keeps tracking of packets that are neither TCP nor UDP (for instance GRE) will survive after having seen last packet matching this entry. Creating PPTP connection this value will be increased automaticly

icmp-timeout (*time*; default: **10s**) - maximal amount of time connection tracking entry will survive after having seen ICMP request

max-entries (*read-only: integer*) - the maximum number of connections the connection state table can contain, depends on an amount of total memory

tcp-close-timeout (*time*; default: **10s**) - maximal amount of time connection tracking entry will survive after having seen connection reset request (RST) or an acknowledgment (ACK) of the connection termination request from connection release initiator

tcp-close-wait-timeout (*time*; default: **10s**) - maximal amount of time connection tracking entry will survive after having seen an termination request (FIN) from responder

tcp-established-timeout (*time*; default: **1d**) - maximal amount of time connection tracking entry will survive after having seen an acknowledgment (ACK) from connection initiator

tcp-fin-wait-timeout (*time*; default: **10s**) - maximal amount of time connection tracking entry will survive after having seen connection termination request (FIN) from connection release initiator

tcp-syn-received-timeout (*time*; default: **1m**) - maximal amount of time connection tracking entry will survive after having seen a matching connection request (SYN)

tcp-syn-sent-timeout (*time*; default: **1m**) - maximal amount of time connection tracking entry will survive after having seen a connection request (SYN) from connection initiator

tcp-syncookie (yes | no; default: **no**) - enable TCP SYN cookies for connections destined to the router itself (this may be useful for HotSpot and tunnels)

tcp-time-wait-timeout (*time*; default: **10s**) - maximal amount of time connection tracking entry will survive after having seen connection termination request (FIN) just after connection request (SYN) or having seen another termination request (FIN) from connection release initiator

total-entries (*read-only: integer*) - number of connections currently recorded in the connection state table

udp-stream-timeout (*time*; default: **3m**) - maximal amount of time connection tracking entry will survive after replay is seen for the last packet matching this entry (connection tracking entry is assured). It is used to increase timeout for such connections as H323, VoIP, etc.

udp-timeout (*time*; default: **10s**) - maximal amount of time connection tracking entry will survive after having seen last packet matching this entry

Notes

The maximum timeout value depends on amount of entries in connection state table. If amount of entries in the table is more than:

- 1/16 of maximum number of entries the maximum timeout value will be 1 day
- 3/16 of maximum number of entries the maximum timeout value will be 1 hour
- 1/2 of maximum number of entries the maximum timeout value will be 10 minute
- 13/16 of maximum number of entries the maximum timeout value will be 1 minute

The shortest timeout will always be chosen between the configured timeout and the value listed above.

If connection tracking timeout value is less than the normal interval between the data packets rate (timeout expires before the next packet arives), NAT and statefull-firewalling stop working.

Service Ports

Home menu level: ***/ip firewall service-port***

Description

Some network protocols are not compatible with network address translation, for example due to some additional infomation about the actual addresses or ports is present in the packet payload, which is not known for the NAT procedures, as they only look at the IP, UDP and TCP headers, not inside the packets. For these protocols to work correctly, a connection tracking helper is needed to work around such design issues. You may enable and disable helpers here (you may want to disable some of them to increase performance or if you are experiencing problems with some protocols detected incorrectly). Note that you can not add or remove the helpers, just enable or disable the existing ones.

Property Description

name - protocol name

ports (*integer*) - port range that is used by the protocol (only some helpers need this)

General Firewall Information

Description

ICMP TYPE:CODE values

In order to protect your router and attached private networks, you need to configure firewall to drop or reject most of ICMP traffic. However, some ICMP packets are vital to maintain network reliability or provide troubleshooting services.

The following is a list of ICMP TYPE:CODE values found in good packets. It is generally suggested to allow these types of ICMP traffic.

- - **8:0** - echo request
  - **0:0** - echo reply

  Ping

- - **11:0** - TTL exceeded
  - **3:3** - Port unreachable

  Trace

- - **3:4** - Fragmentation-DF-Set

  Path MTU discovery

General suggestion to apply ICMP filtering

- Allow ping—ICMP Echo-Request outbound and Echo-Reply messages inbound
- Allow traceroute—TTL-Exceeded and Port-Unreachable messages inbound
- Allow path MTU—ICMP Fragmentation-DF-Set messages inbound
- Block everything else

Peer-to-Peer protocol filtering

Peer-to-peer protocols also known as *p2p* provide means for direct distributed data transfer between individual network hosts. While this technology powers many brilliant applications (like Skype), it is widely abused for unlicensed software and media destribution. Even when it is used for legal purposes, p2p may heavily disturb other network traffic, such as http and e-mail. RouterOS is able to recognize connections of the most popular P2P protocols and filter or enforce QOS on them.

The protocols which can be detected, are:

- **Fasttrack** (Kazaa, KazaaLite, Diet Kazaa, Grokster, iMesh, giFT, Poisoned, mlMac)
- **Gnutella** (Shareaza, XoLoX, , Gnucleus, BearShare, LimeWire (java), Morpheus, Phex, Swapper, Gtk-Gnutella (linux), Mutella (linux), Qtella (linux), MLDonkey, Acquisition (Mac OS), Poisoned, Swapper, Shareaza, XoloX, mlMac)
- **Gnutella2** (Shareaza, MLDonkey, Gnucleus, Morpheus, Adagio, mlMac)
- **DirectConnect** (DirectConnect (AKA DC++), MLDonkey, NeoModus Direct Connect, BCDC++, CZDC++)
- **eDonkey** (eDonkey2000, eMule, xMule (linux), Shareaza, MLDonkey, mlMac, Overnet)
- **Soulseek** (Soulseek, MLDonkey)
- **BitTorrent** (BitTorrent, BitTorrent++, uTorrent, Shareaza, MLDonkey, ABC, Azureus, BitAnarch, SimpleBT, BitTorrent.Net, mlMac)
- **Blubster** (Blubster, Piolet)
- **WPNP** (WinMX)
- **Warez** (Warez, Ares; starting from 2.8.18) - this protocol can only be dropped, speed limiting is impossible

Services, Protocols, and Ports

Document revision 1.1 (February 11, 2008, 4:14 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

This document lists protocols and ports used by various MikroTik RouterOS services. It helps you to determine why your MikroTik router listens to certain ports, and what you need to block/allow in case you want to prevent or grant access to the certain services. Please see the relevant sections of the Manual for more explanations.

Home menu level: ***/ip service***

Modifying Service Settings

Home menu level: ***/ip service***

Property Description

address (*IP addressnetmask*; default: **0.0.0.0/0**) - IP address(-es) from which the service is accessible

certificate (*namenone*; default: **none**) - the name of the certificate used by particular service (absent for the services that do not need certificates)

name - service name

port (*integer*: 1..65535) - the port particular service listens on

Example

To set **www** service to use **8081** port accesible from the **10.10.10.0/24** network:

```
[admin@MikroTik] ip service> print
Flags: X - disabled, I - invalid
 #   NAME                                  PORT  ADDRESS            CERTIFICATE
 0   telnet                                23    0.0.0.0/0
 1   ftp                                   21    0.0.0.0/0
 2   www                                   80    0.0.0.0/0
 3   ssh                                   22    0.0.0.0/0
 4   www-ssl                            443   0.0.0.0/0          none
```

```
[admin@MikroTik] ip service> set www port=8081 address=10.10.10.0/24
[admin@MikroTik] ip service> print
Flags: X - disabled, I - invalid
  #   NAME                                 PORT  ADDRESS            CERTIFICATE
  0   telnet                               23    0.0.0.0/0
  1   ftp                                  21    0.0.0.0/0
  2   www                                  8081  10.10.10.0/24
  3   ssh                                  22    0.0.0.0/0
  4   www-ssl                            443   0.0.0.0/0          none
[admin@MikroTik] ip service>
```

List of Services

Description

Below is the list of protocols and ports used by MikoTik RouterOS services. Some services require additional package to be installed, as well as to be enabled by administrator, *exempli gratia* bandwidth server.

Port/Protocol	Description
20/tcp	**File Transfer Protocol FTP [Data Connection]**
21/tcp	**File Transfer Protocol FTP [Control Connection]**
22/tcp	**Secure Shell SSH remote Login Protocol (Only with security package)**
23/tcp	**Telnet protocol**
53/tcp	**Domain Name Server DNS**
53/udp	**Domain Name Server DNS**
67/udp	**Bootstrap Protocol or DHCP Server (only with dhcp package)**
68/udp	**Bootstrap Protocol or DHCP Client (only with dhcp package)**
80/tcp	**World Wide Web HTTP**
123/udp	**Network Time Protocol NTP (Only with ntp package)**
161/udp	**Simple Network Menagment Protocol SNMP (Only with snmp package)**
443/tcp	**Secure Socket Layer SSL encrypted HTTP(Only with hotspot package)**
500/udp	**Internet Key Exchange IKE protocol (Only with ipsec package)**
520/udp	**Routing Information Protocol RIP (Only with routing package)**
521/udp	**Routing Information Protocol RIP (Only with routing package)**

179/tcp	**Border Gateway Protocol BGP (Only with routing package)**
1080/tcp	**SOCKS proxy protocol**
1701/udp	**Layer 2 Tunnel Protocol L2TP (Only with ppp package)**
1718/udp	**H.323 Gatekeeper Discovery (Only with telephony package)**
1719/tcp	**H.323 Gatekeeper RAS (Only with telephony package)**
1720/tcp	**H.323 Call Setup (Only with telephony package)**
1723/tcp	**Point-to-Point Tuneling Protocol PPTP (Only with ppp package)**
1731/tcp	**H.323 Audio Call Control (Only with telephony package)**
1900/udp	**Universal Plug and Play uPnP**
2828/tcp	**Universal Plug and Play uPnP**
2000/tcp	**Bandwidth-test server**
3986/tcp	**Proxy for winbox**
3987/tcp	**SSL proxy for secure winbox (Only with security package)**
5678/udp	**MikroTik Neighbor Discovery Protocol**
8080/tcp	**HTTP Web proxy (Only with web-proxy package)**
8291/tcp	**Winbox**
20561/udp	**MAC winbox**
5000+/udp	**H.323 RTP Audio Streem (Only with telephony package)**
/1	**ICMP - Internet Control Message Protocol**
/4	**IP - IP in IP (encapsulation)**
/47	**GRE - General Routing Encapsulation (Only for PPTP and EoIP)**
/50	**ESP - Encapsulating Security Payload for IPv4 (Only with security package)**
/51	**AH - Authentication Header for IPv4 (Only with security package)**
/89	**OSPFIGP - OSPF Interior Gateway Protocol**
/112	**VRRP - Virtual Router Redundancy Protocol**

DHCP Client and Server

Document revision 2.8 (December 12, 2007, 11:43 GMT)

This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The DHCP (Dynamic Host Configuration Protocol) is needed for easy distribution of IP addresses in a network. The MikroTik RouterOS implementation includes both server and client parts and is compliant with RFC2131.

General usage of DHCP:

- IP assignment in LAN, cable-modem, and wireless systems
- Obtaining IP settings on cable-modem systems

IP addresses can be bound to MAC addresses using static lease feature.

DHCP server can be used with MikroTik RouterOS HotSpot feature to authenticate and account DHCP clients. See the HotSpot Manual for more information.

Quick Setup Guide

This example will show you how to setup DHCP-Server and DHCP-Client on MikroTik RouterOS.

- Setup of a DHCP Server:
  1. Create an IP address pool

```
/ip pool add name=dhcp-pool ranges=172.16.0.10-172.16.0.20
```

  2. Add a DHCP network which will concern to the network **172.16.0.0/12** and will distribute a gateway with IP address **172.16.0.1** to DHCP clients:

```
/ip dhcp-server network add address=172.16.0.0/12 gateway=172.16.0.1
```

  3. Finally, add a DHCP server:

```
/ip dhcp-server add interface=wlan1 address-pool=dhcp-pool
```

- Setup of a DHCP Client (which will get a lease from the DHCP server, configured above).
  1. Add the DHCP client:

```
/ip dhcp-client add interface=wlan1 use-peer-dns=yes \
     add-default-route=yes disabled=no
```

  2. Check whether you have obtained a lease:

```
[admin@Server] ip dhcp-client> print detail
Flags: X - disabled, I - invalid
 0   interface=wlan1 add-default-route=yes use-peer-dns=yes status=bound
     address=172.16.0.20/12 gateway=172.16.0.1 dhcp-server=192.168.0.1
     primary-dns=159.148.147.194 expires-after=2d23:58:52
[admin@Server] ip dhcp-client>
```

Specifications

Packages required: ***dhcp***
License required: ***level1***
Home menu level: ***/ip dhcp-client, /ip dhcp-server, /ip dhcp-relay***
Standards and Technologies: ***DHCP***

Description

The DHCP protocol gives and allocates IP addresses to IP clients. DHCP is insecure by design and should only be used in trusted networks. DHCP server always listens on UDP 67 port, DHCP client - on UDP 68 port. The initial negotiation involves communication between broadcast addresses (on some phases sender will use source address of **0.0.0.0** and/or destination address of **255.255.255.255**). You should be aware of this when building firewall.

Additional Documents

- ISC Dynamic Host Configuration Protocol (DHCP)
- DHCP mini-HOWTO
- ISC DHCP FAQ

DHCP Client Setup

Home menu level: ***/ip dhcp-client***

Description

The MikroTik RouterOS DHCP client may be enabled on any Ethernet-like interface. There can only be one active DHCP client per interface. The client will accept an address, netmask, default gateway, two DNS server addresses and two NTP server addresses. All other information is ignored. The received IP address with the respective netmask will be added to the corresponding interface. The default gateway will be added to the routing table as a dynamic entry. Should the DHCP client be disabled or not renew an address, the dynamic default route will be removed. If there is already a default route installed prior the DHCP client obtains one, the route obtained by the DHCP client will be shown as invalid.

Property Description

add-default-route (yes | no; default: **yes**) - whether to add the default route to the gateway specified by the DHCP server

address (*read-only: IP addressnetmask*) - IP address and netmask, which is assigned to DHCP Client from the Server

client-id (*text*) - corresponds to the settings suggested by the network administrator or ISP. Commonly it is set to the client's MAC address, but it may as well be any text string

dhcp-server (*read-only: IP address*) - IP address of the DHCP server

expires-after (*read-only: time*) - time, when the lease expires (specified by the DHCP server)

gateway (*read-only: IP address*) - IP address of the gateway which is assigned by DHCP server

host-name (*text*) - the host name of the client as sent to a DHCP server

interface (*name*) - any Ethernet-like interface (this includes wireless and EoIP tunnels) on which the client searches for a DHCP server

primary-dns (*read-only: IP address*) - IP address of the primary DNS server, assigned by the DHCP server

primary-ntp (*read-only: IP address*) - IP address of the primary NTP server, assigned by the DHCP server

secondary-dns (*read-only: IP address*) - IP address of the secondary DNS server, assigned by the DHCP server

secondary-ntp (*read-only: IP address*) - IP address of the secondary NTP server, assigned by the DHCP server

status (*read-only: bound | error | rebinding... | renewing... | requesting... | searching... | stopped*) - shows the status of DHCP slient

use-peer-dns (yes | no; default: **yes**) - whether to accept the DNS settings advertized by DHCP server (they will override the settings put in the /ip dns submenu)

use-peer-ntp (yes | no; default: **yes**) - whether to accept the NTP settings advertized by DHCP server (they will override the settings put in the /system ntp client submenu)

Command Description

release - release current binding and restart DHCP client

renew - renew current leases. If the renew operation was not successful, client tries to reinitialize lease (i.e. it starts lease request procedure (rebind) as if it had not received an IP address yet)

Notes

If **host-name** property is not specified, client's system identity will be sent in the respective field of DHCP request.

If **client-id** property is not specified, client's MAC address will be sent in the respective field of DHCP request.

If **use-peer-dns** property is enabled, the DHCP client will unconditionally rewrite the settings in **/ip dns** submenu. In case two or more DNS servers were received, first two of them are set as primary and secondary servers respectively. In case one DNS server was received, it is put as primary server, and the secondary server is left intact.

Example

To add a DHCP client on **ether1** interface:

```
/ip dhcp-client add interface=ether1 disabled=no
[admin@MikroTik] ip dhcp-client> print detail
Flags: X - disabled, I - invalid
 0   interface=ether1 add-default-route=yes use-peer-dns=yes use-peer-ntp=yes
     status=bound address=192.168.0.65/24 gateway=192.168.0.1
     dhcp-server=192.168.0.1 primary-dns=192.168.0.1 primary-ntp=192.168.0.1
     expires-after=9m44s
[admin@MikroTik] ip dhcp-client>
```

DHCP Server Setup

Home menu level: ***/ip dhcp-server***

Description

The router supports an individual server for each Ethernet-like interface. The MikroTik RouterOS DHCP server supports the basic functions of giving each requesting client an IP address/netmask lease, default gateway, domain name, DNS-server(s) and WINS-server(s) (for Windows clients) information (set up in the DHCP networks submenu)

In order DHCP server to work, you must set up also IP pools (do not include the DHCP server's own IP address into the pool range) and DHCP networks.

It is also possible to hand out leases for DHCP clients using the RADIUS server, here are listed the parameters for used in RADIUS server.

Access-Request:

- **NAS-Identifier** - router identity
- **NAS-IP-Address** - IP address of the router itself
- **NAS-Port** - unique session ID
- **NAS-Port-Type** - Ethernet
- **Calling-Station-Id** - client identifier (active-client-id)
- **Framed-IP-Address** - IP address of the client (active-address)
- **Called-Station-Id** - name of DHCP server
- **User-Name** - MAC address of the client (active-mac-address)
- **Password** - ""

Access-Accept:

- **Framed-IP-Address** - IP address that will be assigned to client
- **Framed-Pool** - ip pool from which to assign ip address to client
- **Rate-Limit** - Datarate limitation for DHCP clients. Format is: rx-rate[/tx-rate] [rx-burst-rate[/tx-burst-rate] [rx-burst-threshold[/tx-burst-threshold] [rx-burst-time[/tx-burst-time][priority] [rx-rate-min[/tx-rate-min]]]]. All rates should be numbers with optional 'k' (1,000s) or 'M' (1,000,000s). If tx-rate is not specified, rx-rate is as tx-rate too. Same goes for tx-burst-rate and tx-burst-threshold and tx-burst-time. If both rx-burst-threshold and tx-burst-threshold are not specified (but burst-rate is specified), rx-rate and tx-rate are used as burst thresholds. If both rx-burst-time and tx-burst-time are not specified, 1s is used as default. Priority takes values 1..8, where 1 implies the highest priority,

but 8 - the lowest. If rx-rate-min and tx-rate-min are not specified rx-rate and tx-rate values are used. The rx-rate-min and tx-rate-min values can not exceed rx-rate and tx-rate values.

- **Ascend-Data-Rate** - tx/rx data rate limitation if multiple attributes are provided, first limits tx data rate, second - rx data rate. If used together with Ascend-Xmit-Rate, specifies rx rate. 0 if unlimited
- **Ascend-Xmit-Rate** - tx data rate limitation. It may be used to specify tx limit only instead of sending two sequental Ascend-Data-Rate attributes (in that case Ascend-Data-Rate will specify the receive rate). 0 if unlimited
- **Session-Timeout** - max lease time (lease-time)

Property Description

add-arp (yes | no; default: **no**) - whether to add dynamic ARP entry:

- **no** - either ARP mode should be enabled on that interface or static ARP entries should be administratively defined in /ip arp submenu

address-pool (*name* | *static-only*; default: **static-only**) - IP pool, from which to take IP addresses for clients

- **static-only** - allow only the clients that have a static lease (i.e. no dynamic addresses will be given to clients, only the ones added in lease submenu)

always-broadcast (yes | no; default: **no**) - always send replies as broadcasts

authoritative (*after-10sec-delay* | *after-2sec-delay* | *no* | *yes*; default: **after-2sec-delay**) - whether the DHCP server is the only one DHCP server for the network

- **after-10sec-delay** - to clients request for an address, dhcp server will wait 10 seconds and if there is another request from the client after this period of time, then dhcp server will offer the address to the client or will send DHCPNAK, if the requested address is not available from this server
- **after-2sec-delay** - to clients request for an address, dhcp server will wait 2 seconds and if there is another request from the client after this period of time, then dhcp server will offer the address to the client or will send DHCPNAK, if the requested address is not available from this server
- **no** - dhcp server ignores clients requests for addresses that are not available from this server
- **yes** - to clients request for an address that is not available from this server, dhcp server will send negative acknowledgment (DHCPNAK)

bootp-support (*none* | *static* | *dynamic*; default: **static**) - support for BOOTP clients

- **none** - do not respond to BOOTP requests
- **static** - offer only static leases to BOOTP clients
- **dynamic** - offer static and dynamic leases for BOOTP clients

delay-threshold (*time*; default: **none**) - if secs field in DHCP packet is smaller than delay-threshold, then this packet is ignored

- **none** - there is no threshold (all DHCP packets are processed)

interface (*name*) - Ethernet-like interface name

lease-time (*time*; default: **72h**) - the time that a client may use the assigned address. The client will try to renew this address after a half of this time and will request a new address after time limit expires

name (*name*) - reference name

relay (*IP address*; default: **0.0.0.0**) - the IP address of the relay this DHCP server should process requests from:

- **0.0.0.0** - the DHCP server will be used only for direct requests from clients (no DHCP really allowed)
- **255.255.255.255** - the DHCP server should be used for any incomming request from a DHCP relay except for those, which are processed by another DHCP server that exists in the /ip dhcp-server submenu

src-address (*IP address*; default: **0.0.0.0**) - the address which the DHCP client must send requests to in order to renew an IP address lease. If there is only one static address on the DHCP server interface and the source-address is left as 0.0.0.0, then the static address will be used. If there are multiple addresses on the interface, an address in the same subnet as the range of given addresses should be used

use-radius (yes | no; default: **no**) - whether to use RADIUS server for dynamic leases

Notes

Client will only receive a DHCP lease in case it is directly reachable by its MAC address through that interface (some wireless bridges may change client's MAC address).

If **authoritative** property is set to **yes**, the DHCP server is sending rejects for the leases it cannot bind or renew. It also may (although not always) help to prevent the network users to run their own DHCP servers illicitly, disturbing the proper way the network should be functioning.

If **relay** property of a DHCP server is not set to **0.0.0.0** the DHCP server will not respond to the direct requests from clients.

Example

To add a DHCP server to interface **ether1**, lending IP addresses from **dhcp-clients** IP pool for 2 hours:

```
/ip dhcp-server add name=dhcp-office disabled=no address-pool=dhcp-clients \
interface=ether1 lease-time=2h
[admin@MikroTik] ip dhcp-server> print
Flags: X - disabled, I - invalid
 #   NAME             INTERFACE RELAY           ADDRESS-POOL LEASE-TIME ADD-ARP
 0   dhcp-office      ether1                    dhcp-clients 02:00:00
[admin@MikroTik] ip dhcp-server>
```

Store Leases on Disk

Home menu level: ***/ip dhcp-server config***

Description

Leases are always stored on disk on graceful shutdown and reboot. If they would be saved on disk on every lease change, a lot of disk writes would happen. There are no problems if it happens on a hard drive, but is very bad for Compact Flash (especially, if lease times are very short). To minimize writes on disk, all changes are saved on disk every **store-leases-disk** seconds. If this time will be very short (immediately), then no changes will be lost even in case of hard reboots and power losts. But, on CF there may be too many writes in case of short lease times (as in case of hotspot). If this time will be very long (never), then there will be no writes on disk, but information about active leases may be lost in case of power loss. In these cases dhcp server may give out the same ip address to another client, if first one will not respond to ping requests.

Property Description

store-leases-disk (*time-interval* | *immediately* | *never*; default: **5min**) - how frequently lease changes should be stored on disk

DHCP Networks

Home menu level: ***/ip dhcp-server network***

Property Description

address (*IP addressnetmask*) - the network DHCP server(s) will lend addresses from

boot-file-name (*text*) - Boot file name

dhcp-option (*text*) - add additional DHCP options from /ip dhcp-server option list. You cannot redefine parameters which are already defined in this submenu:

- **Subnet-Mask (code 1)** - netmask
- **Router (code 3)** - gateway
- **Domain-Server (code 6)** - dns-server
- **Domain-Name (code 15)** - domain
- **NTP-Servers (code 42)** - ntp-server
- **NETBIOS-Name-Server (code 44)** - wins-server

dns-server (*text*) - the DHCP client will use these as the default DNS servers. Two comma-separated DNS servers can be specified to be used by DHCP client as primary and secondary DNS servers

domain (*text*) - the DHCP client will use this as the 'DNS domain' setting for the network adapter

gateway (*IP address*; default: **0.0.0.0**) - the default gateway to be used by DHCP clients

netmask (*integer*: 0..32; default: **0**) - the actual network mask to be used by DHCP client

- **0** - netmask from network address is to be used

next-server (*IP address*) - IP address of next server to use in bootstrap

ntp-server (*text*) - the DHCP client will use these as the default NTP servers. Two comma-separated NTP servers can be specified to be used by DHCP client as primary and secondary NTP servers

wins-server (*text*) - the Windows DHCP client will use these as the default WINS servers. Two comma-separated WINS servers can be specified to be used by DHCP client as primary and secondary WINS servers

Notes

The **address** field uses netmask to specify the range of addresses the given entry is valid for. The actual netmask clients will be using is specified in **netmask** property.

DHCP Server Leases

Home menu level: ***/ip dhcp-server lease***

Description

DHCP server lease submenu is used to monitor and manage server's leases. The issued leases are showed here as dynamic entries. You can also add static leases to issue a particular client (identified by MAC address) the desired IP address.

Generally, the DHCP lease it allocated as follows:

1. an unused lease is in **waiting** state
2. if a client asks for an IP address, the server chooses one
3. if the client will receive statically assigned address, the lease becomes **offered**, and then **bound** with the respective lease time
4. if the client will receive a dynamic address (taken from an IP address pool), the router sends a ping packet and waits for answer for 0.5 seconds. During this time, the lease is marked **testing**
5. in case, the address does not respond, the lease becomes **offered**, and then **bound** with the respective lease time
6. in other case, the lease becomes **busy** for the lease time (there is a command to retest all busy addresses), and the client's request remains unanswered (the client will try again shortly)

A client may free the leased address. The dynamic lease is removed, and the allocated address is returned to the address pool. But the static lease becomes **busy** until the client will reacquire the address.

Note that the IP addresses assigned statically are not probed.

Property Description

active-address (*read-only: IP address*) - actual IP address for this lease

active-client-id (*read-only: text*) - actual client-id of the client

active-mac-address (*read-only: MAC address*) - actual MAC address of the client

active-server (*read-only:*) - actual dhcp server, which serves this client

address (*IP address*) - specify ip address (or ip pool) for static lease
- **0.0.0.0** - use pool from server

agent-circuit-id (*read-only: text*) - circuit ID of DHCP relay agent

agent-remote-id (*read-only: text*) - Remote ID, set by DHCP relay agent

always-broadcast (yes | no) - send all repies as broadcasts

block-access (yes | no; default: **no**) - block access for this client (drop packets from this client)

blocked (*read-only: flag*) - whether the lease is blocked

client-id (*text*; default: **""**) - if specified, must match DHCP 'client identifier' option of the request

expires-after (*read-only: time*) - time until lease expires

host-name (*read-only: text*) - shows host name option from last received DHCP request

lease-time (*time*; default: **0s**) - time that the client may use the address

- **0s** - lease will never expire

mac-address (*MAC address*; default: **00:00:00:00:00:00**) - if specified, must match the MAC address of the client

radius (*read-only:* yes | no) - shows, whether this dynamic lease is authenticated by RADIUS or not

rate-limit (*read-only: text*; default: **""**) - sets rate limit for active lease. Format is: rx-rate[/tx-rate] [rx-burst-rate[/tx-burst-rate] [rx-burst-threshold[/tx-burst-threshold] [rx-burst-time[/tx-burst-time]]]]. All rates should be numbers with optional 'k' (1,000s) or 'M' (1,000,000s). If tx-rate is not specified, rx-rate is as tx-rate too. Same goes for tx-burst-rate and tx-burst-threshold and tx-burst-time. If both rx-burst-threshold and tx-burst-threshold are not specified (but burst-rate is specified), rx-rate and tx-rate is used as burst thresholds. If both rx-burst-time and tx-burst-time are not specified, 1s is used as default

server (*read-only: name*) - server name which serves this client

src-mac-address (*MAC address*) - source MAC address

status (*read-only: waiting* | *testing* | *authorizing* | *busy* | *offered* | *bound*) - lease status:

- **waiting** - not used static lease
- **testing** - testing whether this address is used or not (only for dynamic leases) by pinging it with timeout of 0.5s
- **authorizing** - waiting for response from radius server
- **busy** - this address is assigned statically to a client or already exists in the network, so it can not be leased
- **offered** - server has offered this lease to a client, but did not receive confirmation from the client
- **bound** - server has received client's confirmation that it accepts offered address, it is using it now and will free the address not later, than the lease time will be over

use-src-mac (*MAC address*) - use this source MAC address instead

Command Description

check-status - check status of a given busy dynamic lease, and free it in case of no response

make-static - convert a dynamic lease to a static one

Notes

If **rate-limit** is specified, a simple queue is added with corresponding parameters when lease enters bound state. Arp entry is added right after adding of queue is done (only if add-arp is enabled for dhcp server). To be sure, that client cannot use his ip address without getting dhcp lease and thus avoiding rate-limit, reply-only mode must be used on that ethernet interface.

Even though client address may be changed (with adding a new item) in **lease print** list, it will not change for the client. It is true for any changes in the DHCP server configuration because of the nature of the DHCP protocol. Client tries to renew assigned IP address only when half a lease time is past (it tries to renew several times). Only when full lease time is past and IP address was not renewed, new lease is asked (rebind operation).

the deault **mac-address** value will never work! You should specify a correct MAC address there.

Example

To assign 10.5.2.100 static IP address for the existing DHCP client (shown in the lease table as item #0):

```
[admin@MikroTik] ip dhcp-server lease> print
Flags: X - disabled, R - radius, D - dynamic, B - blocked
 #   ADDRESS         MAC-ADDRESS       HOST-NAME     SERVER RATE-LIMIT STATUS
 0 D 10.5.2.90       00:04:EA:C6:0E:40               switch            bound
 1 D 10.5.2.91       00:04:EA:99:63:C0               switch            bound
[admin@MikroTik] ip dhcp-server lease> add copy-from=0 address=10.5.2.100
[admin@MikroTik] ip dhcp-server lease> print
Flags: X - disabled, R - radius, D - dynamic, B - blocked
 #   ADDRESS         MAC-ADDRESS       HOST-NAME     SERVER RATE-LIMIT STATUS
 0 D 10.5.2.91       00:04:EA:99:63:C0               switch            bound
 1   10.5.2.100      00:04:EA:C6:0E:40               switch            bound
[admin@MikroTik] ip dhcp-server lease>
```

DHCP Alert

Home menu level: ***/ip dhcp-server alert***

Description

To find any rogue DHCP servers as soon as they appear in your network, DHCP Alert tool can be used. It will monitor ethernet for all DHCP replies and check, whether this reply comes from a valid DHCP server. If reply from unknown DHCP server is detected, alert gets triggered:

```
[admin@MikroTik] ip dhcp-server alert>/log print
00:34:23 dhcp,critical,error,warning,info,debug dhcp alert on Public:
    discovered unknown dhcp server, mac 00:02:29:60:36:E7, ip 10.5.8.236
[admin@MikroTik] ip dhcp-server alert>
```

When the system alerts about a rogue DHCP server, it can execute a custom script.

As DHCP replies can be unicast, rogue dhcp detector may not receive any offer to other dhcp clients at all. To deal with this, rogue dhcp detector acts as a dhcp client as well - it sends out dhcp discover requests once a minute

Property Description

alert-timeout (*nonetime*; default: **none**) - time, after which alert will be forgotten. If after that time the same server will be detected, new alert will be generated

- **none** - infinite time

interface (*name*) - interface, on which to run rogue DHCP server finder

on-alert (*text*) - script to run, when an unknown DHCP server is detected

unknown-server (*read-only: text*) - list of MAC addresses of detected unknown DHCP servers. Server is removed from this list after alert-timeout

valid-server (*text*) - list of MAC addresses of valid DHCP servers

Notes

All alerts on an interface can be cleared at any time using command: **/ip dhcp-server alert reset-alert <interface>**

Note, that e-mail can be sent, using */system logging action add target=email*

DHCP Option

Home menu level: ***/ip dhcp-server option***

Description

With help of DHCP Option list, it is possible to define additional custom options for DHCP Server to advertise.

Property Description

code (*integer*: 1..254) - dhcp option code. All codes are available at http://www.iana.org/assignments/bootp-dhcp-parameters

name (*name*) - descriptive name of the option

value (*text*) - parameter's value in form of a string. If the string begins with "0x", it is assumed as a hexadecimal value

Notes

The defined options you can use in */ip dhcp-server network* submenu

According to the DHCP protocol, a parameter is returned to the DHCP client only if it requests this parameter, specifying the respective code in DHCP request Parameter-List (code 55) attribute. If the code is not included in Parameter-List attribute, DHCP server will not send it to the DHCP client.

Example

This example shows how to set DHCP server to reply on DHCP client's Hostname request (code 12) with value **Host-A**.

Add an option named **Option-Hostname** with code **12** (Hostname) and value **Host-A**:

```
[admin@MikroTik] ip dhcp-server option> add name=Hostname code=12 \
value="Host-A"
[admin@MikroTik] ip dhcp-server option> print
 # NAME                                 CODE VALUE
 0 Option-Hostname                      12   Host-A
[admin@MikroTik] ip dhcp-server option>
```

Use this option in DHCP server network list:

```
[admin@MikroTik] ip dhcp-server network> add address=10.1.0.0/24 \
\... gateway=10.1.0.1 dhcp-option=Option-Hostname dns-server=159.148.60.20
[admin@MikroTik] ip dhcp-server network> print detail
 0 address=10.1.0.0/24 gateway=10.1.0.1 dns-server=159.148.60.20
   dhcp-option=Option-Hostname
```

```
[admin@MikroTik] ip dhcp-server network>
```

Now the DHCP server will reply with its Hostname **Host-A** to DHCP client (if requested)

DHCP Relay

Home menu level: ***/ip dhcp-relay***

Description

DHCP Relay is just a proxy that is able to receive a DHCP request and resend it to the real DHCP server

Property Description

delay-threshold (*time*; default: **none**) - if secs field in DHCP packet is smaller than delay-threshold, then this packet is ignored

dhcp-server (*text*) - list of DHCP servers' IP addresses which should the DHCP requests be forwarded to

interface (*name*) - interface name the DHCP relay will be working on

local-address (*IP address*; default: **0.0.0.0**) - the unique IP address of this DHCP relay needed for DHCP server to distinguish relays:

- **0.0.0.0** - the IP address will be chosen automatically

name (*name*) - descriptive name for relay

Notes

DHCP relay does not choose the particular DHCP server in the dhcp-server list, it just send the incoming request to all the listed servers.

Example

To add a DHCP relay named **relay** on **ether1** interface resending all received requests to the **10.0.0.1** DHCP server:

```
[admin@MikroTik] ip dhcp-relay> add name=relay interface=ether1 \
\... dhcp-server=10.0.0.1 disabled=no
[admin@MikroTik] ip dhcp-relay> print
Flags: X - disabled, I - invalid
  #   NAME                            INTERFACE DHCP-SERVER     LOCAL-ADDRESS
  0   relay                           ether1    10.0.0.1        0.0.0.0

[admin@MikroTik] ip dhcp-relay>
```

Question&Answer-Based Setup

Command name: ***/ip dhcp-server setup***

Questions

addresses to give out (*text*) - the pool of IP addresses DHCP server should lease to the clients

dhcp address space (*IP addressnetmask*; default: **192.168.0.0/24**) - network the DHCP server will lease to the clients

dhcp relay (*IP address*; default: **0.0.0.0**) - the IP address of the DHCP relay between the DHCP server and the DHCP clients

dhcp server interface (*name*) - interface to run DHCP server on

dns servers (*IP address*) - IP address of the appropriate DNS server to be propagated to the DHCP clients

gateway (*IP address*; default: **0.0.0.0**) - the default gateway of the leased network

lease time (*time*; default: **3d**) - the time the lease will be valid

Notes

Depending on current settings and answers to the previous questions, default values of following questions may be different. Some questions may disappear if they become redundant (for example, there is no use of asking for 'relay' when the server will lend the directly connected network)

Example

To configure DHCP server on **ether1** interface to lend addresses from 10.0.0.2 to 10.0.0.254 which belong to the **10.0.0.0/24** network with **10.0.0.1** gateway and **159.148.60.2** DNS server for the time of 3 days:

```
[admin@MikroTik] ip dhcp-server> setup
Select interface to run DHCP server on

dhcp server interface: ether1
Select network for DHCP addresses

dhcp address space: 10.0.0.0/24
Select gateway for given network

gateway for dhcp network: 10.0.0.1
Select pool of ip addresses given out by DHCP server

addresses to give out: 10.0.0.2-10.0.0.254
Select DNS servers

dns servers: 159.148.60.20
Select lease time

lease time: 3d
[admin@MikroTik] ip dhcp-server>
```

The wizard has made the following configuration based on the answers above:

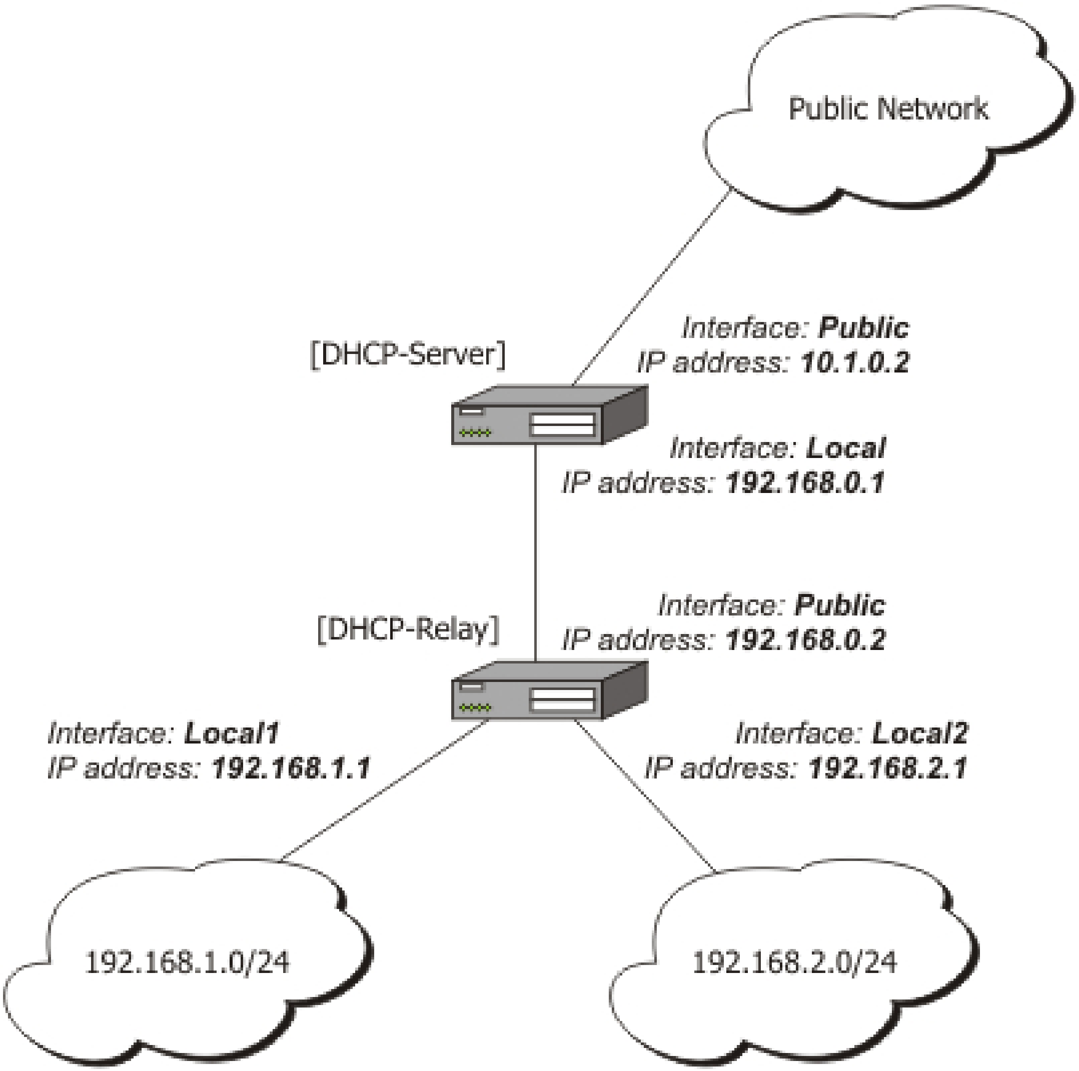
```
[admin@MikroTik] ip dhcp-server> print
Flags: X - disabled, I - invalid
  #   NAME            INTERFACE RELAY           ADDRESS-POOL LEASE-TIME ADD-ARP
  0   dhcp1           ether1    0.0.0.0         dhcp_pool1   3d         no

[admin@MikroTik] ip dhcp-server> network print
  # ADDRESS            GATEWAY         DNS-SERVER      WINS-SERVER     DOMAIN
  0 10.0.0.0/24        10.0.0.1        159.148.60.20

[admin@MikroTik] ip dhcp-server> /ip pool print
  # NAME                                        RANGES
  0 dhcp_pool1                                      10.0.0.2-10.0.0.254
[admin@MikroTik] ip dhcp-server>
```

Application Examples

Dynamic Addressing, using DHCP-Relay

Let us consider that you have several IP networks 'behind' other routers, but you want to keep all DHCP servers on a single router. To do this, you need a DHCP relay on your network which relies DHCP requests from clients to DHCP server.

This example will show you how to configure a DHCP server and a DHCP relay which serve 2 IP networks - **192.168.1.0/24** and **192.168.2.0/24** that are behind a router **DHCP-Relay**.



IP addresses of **DHCP-Server**:

```
[admin@DHCP-Server] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   192.168.0.1/24     192.168.0.0     192.168.0.255   To-DHCP-Relay
 1   10.1.0.2/24 10.1.0.0 10.1.0.255 Public
[admin@DHCP-Server] ip address>
```

IP addresses of **DHCP-Relay**:

```
[admin@DHCP-Relay] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         BROADCAST       INTERFACE
 0   192.168.0.1/24     192.168.0.0     192.168.0.255   To-DHCP-Server
 1   192.168.1.1/24     192.168.1.0     192.168.1.255   Local1
 2   192.168.2.1/24     192.168.2.0     192.168.2.255   Local2
[admin@DHCP-Relay] ip address>
```

To setup 2 DHCP Servers on **DHCP-Server** router add 2 pools. For networks **192.168.1.0/24** and **192.168.2.0**:

```
/ip pool add name=Local1-Pool ranges=192.168.1.11-192.168.1.100
/ip pool add name=Local1-Pool ranges=192.168.2.11-192.168.2.100
```

```
[admin@DHCP-Server] ip pool> print
 # NAME                                         RANGES
 0 Local1-Pool                                  192.168.1.11-192.168.1.100
 1 Local2-Pool                                  192.168.2.11-192.168.2.100
[admin@DHCP-Server] ip pool>
```

Create DHCP Servers:

```
/ip dhcp-server add interface=To-DHCP-Relay relay=192.168.1.1 \
   address-pool=Local1-Pool name=DHCP-1 disabled=no
/ip dhcp-server add interface=To-DHCP-Relay relay=192.168.2.1 \
   address-pool=Local2-Pool name=DHCP-2 disabled=no
```

```
[admin@DHCP-Server] ip dhcp-server> print
Flags: X - disabled, I - invalid
 #   NAME         INTERFACE     RELAY           ADDRESS-POOL LEASE-TIME ADD-ARP
 0   DHCP-1       To-DHCP-Relay 192.168.1.1     Local1-Pool  3d00:00:00
 1   DHCP-2       To-DHCP-Relay 192.168.2.1     Local2-Pool  3d00:00:00
[admin@DHCP-Server] ip dhcp-server>
```

Configure respective networks:

```
/ip dhcp-server network add address=192.168.1.0/24 gateway=192.168.1.1 \
   dns-server=159.148.60.20
/ip dhcp-server network add address=192.168.2.0/24 gateway=192.168.2.1 \
   dns-server 159.148.60.20
```

```
[admin@DHCP-Server] ip dhcp-server network> print
 # ADDRESS            GATEWAY         DNS-SERVER      WINS-SERVER     DOMAIN
 0 192.168.1.0/24     192.168.1.1     159.148.60.20
 1 192.168.2.0/24     192.168.2.1     159.148.60.20
[admin@DHCP-Server] ip dhcp-server network>
```

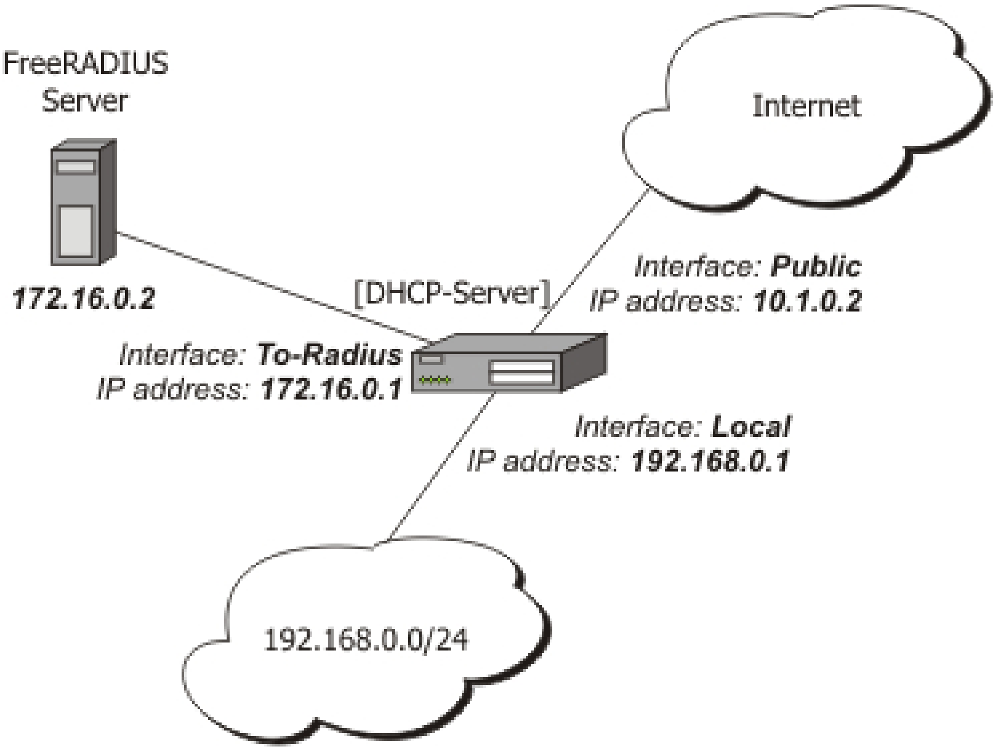
Configuration of **DHCP-Server** is done. Now let's configure **DHCP-Relay**:

```
/ip dhcp-relay add name=Local1-Relay interface=Local1 \
   dhcp-server=192.168.0.1 local-address=192.168.1.1 disabled=no
/ip dhcp-relay add name=Local2-Relay interface=Local2 \
   dhcp-server=192.168.0.1 local-address=192.168.2.1 disabled=no
```

```
[admin@DHCP-Relay] ip dhcp-relay> print
Flags: X - disabled, I - invalid
 #   NAME                        INTERFACE      DHCP-SERVER     LOCAL-ADDRESS
 0   Local1-Relay                Local1         192.168.0.1      192.168.1.1
 1   Local2-Relay                Local2         192.168.0.1      192.168.2.1
[admin@DHCP-Relay] ip dhcp-relay>
```

IP Address assignment, using FreeRADIUS Server

Let us consider that we want to assign IP addresses for clients, using the RADIUS server.



We assume that you already have installed FreeRADIUS. Just add these lines to specified files:

users file:

```
00:0B:6B:31:02:4B       Auth-Type := Local, Password == ""
        Framed-IP-Address = 192.168.0.55
```

clients.conf file

```
client 172.16.0.1 {
    secret = MySecret
    shortname = Server
}
```

Configure Radius Client on RouterOS:

```
/radius add service=dhcp address=172.16.0.2 secret=MySecret
```

```
[admin@DHCP-Server] radius> print detail
Flags: X - disabled
 0   service=dhcp called-id="" domain="" address=172.16.0.2 secret="MySecret"
     authentication-port=1812 accounting-port=1813 timeout=00:00:00.300
     accounting-backup=no realm=""
[admin@DHCP-Server] radius>
```

Setup DHCP Server:

1. Create an address pool:

```
/ip pool add name=Radius-Clients ranges=192.168.0.11-192.168.0.100
```

2. Add a DHCP server:

```
/ip dhcp-server add address-pool=Radius-Clients use-radius=yes interface=Local \
   disabled=no
```

3. Configure DHCP networks:

```
/ip dhcp-server network add address=192.168.0.0/24 gateway=192.168.0.1 \
dns-server=159.148.147.194,159.148.60.20
```

Now the client with MAC address **00:0B:6B:31:02:4B** will always receive IP address **192.168.0.55**.

DNS Client and Cache

Document revision 1.3 (November 28, 2007, 10:44 GMT)

This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

DNS cache is used to minimize DNS requests to an external DNS server as well as to minimize DNS resolution time. This is a simple recursive DNS server with local items.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip dns***
Standards and Technologies: ***DNS***
Hardware usage: ***Not significant***

Description

A MikroTik router with DNS feature enabled can be set as a DNS server for any DNS-compliant client. Moreover, MikroTik router can be specified as a primary DNS server under its dhcp-server settings. When the remote requests are enabled, the MikroTik router responds to TCP and UDP DNS requests on port 53.

Additional Documents

- http://www.freesoft.org/CIE/Course/Section2/3.htm
- http://www.networksorcery.com/enp/protocol/dns.htm
- RFC1035

DNS Cache Setup

Home menu level: ***/ip dns***

Description

DNS facility is used to provide domain name resolution for router itself as well as for the clients connected to it.

Property Description

allow-remote-requests (yes | no; default: **no**) - specifies whether to allow network requests

cache-max-ttl (*time*; default: **1w**) - specifies maximum time-to-live for cache records. In other words, cache records will expire unconditionally after cache-max-ttl time. Shorter TTL received from DNS servers are respected

cache-size (*integer*: 512..10240; default: **2048KiB**) - specifies the size of DNS cache in KiB

cache-used (*read-only: integer*) - displays the current cache size in KiB

primary-dns (*IP address*; default: **0.0.0.0**) - primary DNS server

secondary-dns (*IP address*; default: **0.0.0.0**) - secondary DNS server

Notes

If the property **use-peer-dns** under **/ip dhcp-client** is set to **yes** then **primary-dns** under **/ip dns** will change to a DNS address given by DHCP Server.

Example

To set 159.148.60.2 as the primary DNS server and allow the router to be used as a DNS server, do the following:

```
[admin@MikroTik] ip dns> set primary-dns=159.148.60.2 \
\... allow-remote-requests=yes
[admin@MikroTik] ip dns> print
            primary-dns: 159.148.60.2
          secondary-dns: 0.0.0.0
  allow-remote-requests: yes
             cache-size: 2048KiB
          cache-max-ttl: 1w
```

```
             cache-used: 7KiB
[admin@MikroTik] ip dns>
```

Cache Monitoring

Home menu level: ***/ip dns cache***

Description

This menu provides a list with all address (DNS type "A") records stored on the server

Property Description

address (*read-only: IP address*) - IP address of the host

name (*read-only: name*) - DNS name of the host

ttl (*read-only: time*) - remaining time-to-live for the record

All DNS Entries

Home menu level: ***/ip dns cache all***

Description

This menu provides a complete list with all DNS records stored on the server

Property Description

data (*read-only: text*) - DNS data field. IP address for type "A" records. Other record types may have different contents of the data field (like hostname or arbitrary text)

name (*read-only: name*) - DNS name of the host

ttl (*read-only: time*) - remaining time-to-live for the record

type (*read-only: text*) - DNS record type

Static DNS Entries

Home menu level: ***/ip dns static***

Description

The MikroTik RouterOS has an embedded DNS server feature in DNS cache. It allows you to link the particular domain names with the respective IP addresses and advertize these links to the DNS clients using the router as their DNS server. This feature can also be used to provide fake DNS information to your network clients. For example, resolving any DNS request for a certain set of domains (or for the whole Internet) to your own page.

The server is capable of resolving DNS requests based on POSIX basic regular expressions, so that multiple requets can be matched with the same entry. In case an entry does not conform with DNS naming standards, it is considered a regular expression and marked with ‘R’ flag. The list is ordered and is checked from top to bottom. Regular expressions are checked first, then the plain records.

Property Description

address (*IP address*) - IP address to resolve domain name with

name (*text*) - DNS name to be resolved to a given IP address. May be a regular expression

ttl (*time*) - time-to-live of the DNS record

Notes

Reverse DNS lookup (Address to Name) of the regular expression entries is not possible. You can, however, add an additional plain record with the same IP address and specify some name for it.

Remember that the meaning of a dot (.) in regular expressions is any character, so the expression should be escaped properly. For example, if you need to match anything within **example.com** domain but not all the domains that just end with *example.com*, like *www.another-example.com*, use `name=".*\\.example\\.com"`

Regular expression matching is significantly slower than of the plain entries, so it is advised to minimize the number of regular expression rules and optimize the expressions themselves.

Example

To add a static DNS entry for **www.example.com** to be resolved to **10.0.0.1** IP address:

```
[admin@MikroTik] ip dns static> add name www.example.com address=10.0.0.1
[admin@MikroTik] ip dns static> print
Flags: D - dynamic, X - disabled, R - regexp
 #     NAME                ADDRESS                                 TTL
 0     www.example.com     10.0.0.1                                1d
[admin@MikroTik] ip dns static>
```

Flushing DNS cache

Command name: ***/ip dns cache flush***

Command Description

flush - clears internal DNS cache

Example

```
[admin@MikroTik] ip dns> cache flush
[admin@MikroTik] ip dns> print
              primary-dns: 159.148.60.2
            secondary-dns: 0.0.0.0
    allow-remote-requests: yes
               cache-size: 2048 KiB
            cache-max-ttl: 1w
               cache-used: 10 KiB
[admin@MikroTik] ip dns>
```

HotSpot Gateway

Document revision 4.3 (January 14, 2008, 8:59 GMT)

This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik HotSpot Gateway enables providing of public network access for clients using wireless or wired network connections.

HotSpot Gateway features:

- authentication of clients using local client database, or RADIUS server
- accounting using local database, or RADIUS server
- Walled-garden system (accessing some web pages without authorization)

Quick Setup Guide

Given a router with two interfaces: Local (where HotSpot clients are connected to) and Public, which is connected to the Internet. To set up HotSpot on the Local interface:

1. first, a valid IP configuration is required on both interfaces. This can be done with **/setup** command or by setting up the router manually. In this example we will assume the configuration with DHCP server already enabled on the Local interface
2. valid DNS configuration must be set up in the **/ip dns** submenu
3. To put HotSpot on the Local interface, using the same IP address pool as DHCP server uses for that interface: `/ip hotspot add interface=local address-pool=dhcp-pool-1`

4. and finally, add at least one HotSpot user: `/ip hotspot user add name=admin`

These simple steps should be sufficient to enable HotSpot system

Please find HotSpot How-to's, which will answer most of your questions about configuring a HotSpot gateway, at the end of this manual. It is still recommended that you read and understand all the **Description** section below before deploying a HotSpot system.

If this does not work:

- check that **/ip dns** contains valid DNS servers, try to **/ping www.example.com** to see, that DNS resolving works
- make sure that connection tracking is enabled: `/ip firewall connection tracking set enabled=yes`

Specifications

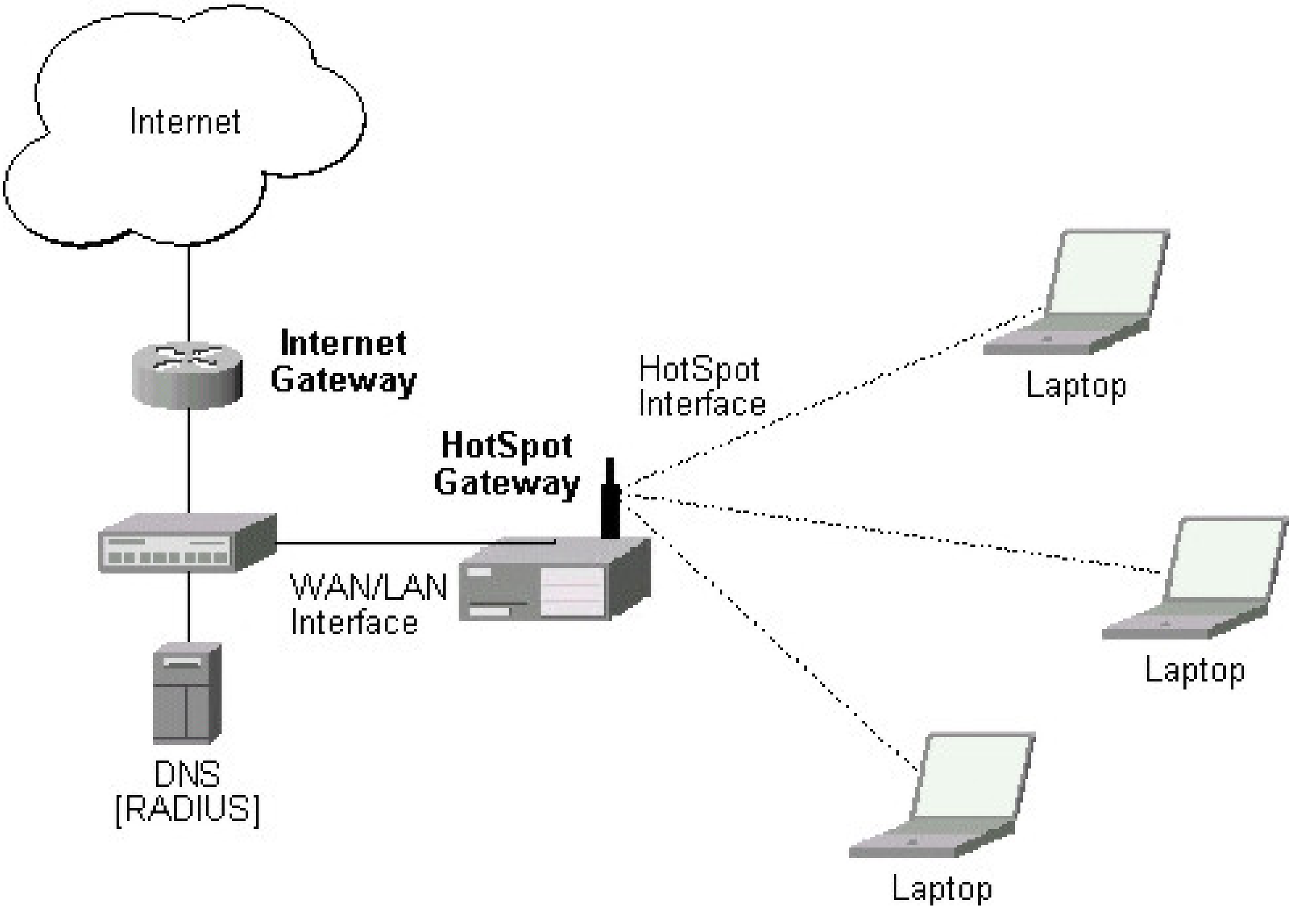
Packages required: ***hotspot, dhcp (optional)***
License required: ***level1 (Limited to 1 active user), level3 (Limited to 1 active user), level4 (Limited to 200 active users), level5 (Limited to 500 active users), level6***
Home menu level: ***/ip hotspot***
Standards and Technologies: ***ICMP***, ***DHCP***
Hardware usage: ***Not significant***

Description

MikroTik HotSpot Gateway should have at least two network interfaces:

1. HotSpot interface, which is used to connect HotSpot clients
2. LAN/WAN interface, which is used to access network resources. For example, DNS and RADIUS server(s) should be accessible

The diagram below shows a sample HotSpot setup.



The HotSpot interface should have an IP address assigned to it. Physical network connection has to be established between the HotSpot user's computer and the gateway. It can be wireless (the wireless card should be registered to AP), or wired (the NIC card should be connected to a hub or a switch).

Introduction to HotSpot

HotSpot is a way to authorize users to access some network resources. It does not provide traffic encryption. To log in, users may use almost any web browser (either HTTP or HTTPS protocol), so they are not required to install additional software. The gateway is accounting the uptime and amount of traffic each of its clients have used, and also can send this information to a RADIUS server. The HotSpot system may limit each particular user's bitrate, total amount of traffic, uptime and some other parameters to be mentioned further in this document.

The HotSpot system is targeted to provide authentication within a local network (for the local network users to access the Internet), but may as well be used to authorize access from outer networks to access local resources (like an authentication gateway for the outside world to access your network). Configuring Walled Garden feature, it is possible to allow users to access some web pages without the need of prior authentication.

Getting Address

First of all, a client must get an IP address. It may be set on the client statically, or leased from a DHCP server. The DHCP server may provide ways of binding lent IP addresses to clients MAC addresses, if required. The HotSpot system does not care how did a client get an address before he/she gets to the HotSpot login page.

Moreover, HotSpot server may automatically and transparently change any IP address (yes, meaning really **any** IP address) of a client to a valid unused address from the selected IP pool. If a user is able to get his/her Internet connection working at their place, he/she will be able to get his/her connection working in the HotSpot network. This feature gives a possibility to provide a network access (for example, Internet access) to mobile clients that are not willing (or are disallowed, not qualified enough or otherwise unable) to change their networking settings. The users will not notice the translation (i.e., there will not be any changes in the users' config), but the router itself will see completely different (from what is actually set on each client) source IP addresses on packets sent from the clients (even the firewall mangle table will 'see' the translated addresses). This technique is called one-to-one NAT, but is also known as "Universal Client" as that is how it was called in the RouterOS version 2.8.

One-to-one NAT accepts any incoming address from a connected network interface and performs a network address translation so that data may be routed through standard IP networks. Clients may use any preconfigured addresses. If the one-to-one NAT feature is set to translate a client's address to a public IP address, then the client may even run a server or any other service that requires a public IP address. This NAT is changing source address of each packet just after it is received by the router (it is like source NAT that is performed early in the packet path, so that even firewall mangle table, which normally 'sees' received packets unaltered, can only 'see' the translated address).

Note also that **arp** mode must be **enabled** on the interface you use one-to-one NAT on.

Before the authentication

When enabling HotSpot on an interface, the system automatically sets up everything needed to show login page for all clients that are not logged in. This is done by adding dynamic destination NAT rules, which you can observe on a working HotSpot system. These rules are needed to redirect all HTTP and HTTPS requests from unauthorized users to the HotSpot authentication proxy. Other rules that are also inserted, will be described later in a special section of this manual.

In most common setup, opening any HTTP page will bring up the HotSpot servlet login page (which can be customized extensively, as described later on). As normal user behavior is to open web pages by their DNS names, a valid DNS configuration should be set up on the HotSpot gateway itself (it is possible to reconfigure the gateway so that it will not require local DNS configuration, but such a configuration is impractical and thus not recommended).

Walled Garden

You may wish not to require authorization for some services (for example to let clients access the web server of your company without registration), or even to require authorization only to a number of services (for example, for users to be allowed to access an internal file server or another restricted area). This can be done by setting up Walled Garden system.

When a not logged-in user requests a service allowed in the Walled Garden configuration, the HotSpot gateway does not intercept it, or in case of HTTP, simply redirects the request to the original destination. Other requests are redirected to the HotSpot servlet (login page infrastructure). When a user is logged in, there is no effect of this table on him/her.

Walled Garden for HTTP requests is using the embedded proxy server (**/ip proxy**). This means that all the configured parameters of that proy server will also be effective for the WalledGarden clients (as well as for all clients that have transparent proxy enabled)

Authentication

- **HTTP PAP** - simplest method, which shows the HotSpot login page and expect to get the authentication info (i.e. username and password) in plain text. Note that passwords are not being encrypted when transferred over the network. Another use of this method is the possibility of hard-coded authentication information in the servlet's login page simply creating the appropriate link.
- **HTTP CHAP** - standard method, which includes CHAP challenge in the login page. The CHAP MD5 hash challenge is to be used together with the user's password for computing the string which will be sent to the HotSpot gateway. The hash result (as a password) together with username is sent over network to HotSpot service (so, password is never sent in plain text over IP network). On the client side, MD5 algorithm is implemented in JavaScript applet, so if a browser does not support JavaScript (like, for example, Internet Explorer 2.0 or some PDA browsers) or it has JavaScipt disabled, it will not be able to authenticate users. It is possible to allow unencrypted passwords to be accepted by turning on HTTP PAP authentication method, but it is not recommended (due to security considerations) to use that feature.
- **HTTPS** - the same as HTTP PAP, but using SSL protocol for encrypting transmissions. HotSpot user just send his/her password without additional hashing (note that there is no need to worry about plain-text password exposure over the network, as the transmission itself is encrypted). In either case, HTTP POST method (if not possible, then - HTTP GET method) is used to send data to the HotSpot gateway.
- **HTTP cookie** - after each successful login, a cookie is sent to the web browser and the same cookie is added to active HTTP cookie list. Next time the same user will try to log in, web browser will send the saved HTTP cookie. This cookie will be compared with the one stored on the HotSpot gateway and only if source MAC address and randomly generated ID match the ones stored on the gateway, user will be automatically logged in using the login information (username and password pair) was used when the cookie was first generated. Otherwise, the user will be prompted to log in, and in the case authentication is successful, old cookie will be removed from the local HotSpot active cookie list and the new one with different random ID and expiration time will be added to the list and sent to the web browser. It is also possible to erase cookie on user manual logoff (not in the default server pages, but you can modify them to perform this). This method may only be used together with HTTP PAP, HTTP CHAP or HTTPS methods as there would be nothing to generate cookies in the first place otherwise.
- **MAC address** - try to authenticate clients as soon as they appear in the hosts list (i.e., as soon as they have sent any packet to the HotSpot server), using client's MAC address as username.
- **Trial** - users may be allowed to use the service free of charge for some period of time for evaluation, and be required to authenticate only after this period is over. HotSpot can be configured to allow some amount of time per MAC address to be freely used with some limitations imposed by the provided user profile. In case the MAC address still has some trial time unused, the login page will contain the link for trial login. The time is automatically reset after the configured amount of time (so that, for example, any MAC address may use 30 minutes a day without ever registering). The username of such a user (as seen in the active user table and in the login link) is "T-XX:XX:XX:XX:XX:XX" (where XX:XX:XX:XX:XX:XX is his/her MAC address). The authentication procedure will not ask RADIUS server permission to

authorise such a user.

There are currently 6 different authentication methods. You can use one or more of them simultaneously:

HotSpot can authenticate users consulting the local user database or a RADIUS server (local database is consulted first, then - a RADIUS server). In case of HTTP cookie authentication via RADIUS server, the router will send the same information to the server as was used when the cookie was first generated. If authentication is done locally, profile corresponding to that user is used, otherwise (in case RADIUS reply did not contain the group for that user) the default profile is used to set default values for parameters, which are not set in RADIUS access-accept message. For more information on how the interaction with a RADIUS server works, see the respective manual section.

The HTTP PAP method also makes it possible to authenticate by requesting the page `/login?username=username&password=password` . In case you want to log in using telnet connection, the exact HTTP request would look like that: **GET /login?username=username&password=password HTTP/1.0** (note that the request is case-sensitive)

Authorization

After authentication, user gets access to the Internet, and receives some limitations (which are user profile specific). HotSpot may also perform a one-to-one NAT for the client, so that a particular user would always receive the same IP address regardless of what PC is he/she working at.

The system will automatically detect and redirect requests to a proxy server a client is using (if any; it may be set in his/her settings to use an unknown to us proxy server) to the proxy server embedded in the router.

Authorization may be delegated to a RADIUS server, which delivers similar configuration options as the local database. For any user requiring authorization, a RADIUS server gets queried first, and if no reply received, the local database is examined. RADIUS server may send a Change of Authorization request according to standards to alter the previously accepted parameters.

Advertisement

The same proxy used for unauthorized clients to provide Walled-Garden facility, may also be used for authorized users to show them advertisement popups. Transparent proxy for authorized users allows to monitor http requests of the clients and to take some action if required. It enables the possibility to open status page even if client is logged in by mac address, as well as to show advertisements time after time

When the time has come to show an advertisement, the server redirects client's web browser to the status page. Only requests, which provide html content, are redirected (images and other content will not be affected). The status page displays the advertisement and next advertise-interval is used to schedule next advertisement. If status page is unable to display an advertisement for configured timeout starting from moment, when it is scheduled to be shown, client access is blocked within walled-garden (just as unauthorized clients are). Client is unblocked when the scheduled page is finally shown. Note that if popup windows are blocked in the browser, the link on the status page may be used to open the advertisement manually.

While client is blocked, FTP and other services will not be allowed. Thus requiring client to open an advertisement for any Internet activity not especially allowed by the Walled-Garden.

Accounting

The HotSpot system implement accounting internally, you are not required to do anything special for it to work. The accounting information for each user may be sent to a RADIUS server.

Configuration menus

- **/ip hotspot** - HotSpot servers on particular interfaces (one server per interface). HotSpot server must be added in this menu in order for HotSpot system to work on an interface
- **/ip hotspot profile** - HotSpot server profiles. Settings, which affect login procedure for HotSpot clients are configured here. More than one HotSpot servers may use the same profile
- **/ip hotspot host** - dynamic list of active network hosts on all HotSpot interfaces. Here you can also find IP address bindings of the one-to-one NAT
- **/ip hotspot ip-binding** - rules for binding IP addresses to hosts on hotspot interfaces
- **/ip hotspot service-port** - address translation helpers for the one-to-one NAT
- **/ip hotspot walled-garden** - Walled Garden rules at HTTP level (DNS names, HTTP request substrings)
- **/ip hotspot walled-garden ip** - Walled Garden rules at IP level (IP addresses, IP protocols)
- **/ip hotspot user** - local HotSpot system users
- **/ip hotspot user profile** - local HotSpot system users profiles (user groups)
- **/ip hotspot active** - dynamic list of all authenticated HotSpot users
- **/ip hotspot cookie** - dynamic list of all valid HTTP cookies

Question&Answer-Based Setup

Command name: ***/ip hotspot setup***

Questions

address pool of network (*name*) - IP address pool for the HotSpot network

dns name (*text*) - DNS domain name of the HotSpot gateway (will be statically configured on the local DNS proxy

dns servers (*IP addressIP address*) - DNS servers for HotSpot clients

hotspot interface (*name*) - interface to run HotSpot on

ip address of smtp server (*IP address*; default: **0.0.0.0**) - IP address of the SMTP server to redirect SMTP requests (TCP port 25) to

- **0.0.0.0** - no redirect

local address of network (*IP address*; default: **10.5.50.1/24**) - HotSpot gateway address for the interface

masquerade network (yes | no; default: **yes**) - whether to masquerade the HotSpot network

name of local hotspot user (*text*; default: **admin**) - username of one automatically created user

passphrase (*text*) - the passphrase of the certificate you are importing

password for the user (*text*) - password for the automatically created user

select certificate (*namenone | import-other-certificate*) - choose SSL certificate from the list of the imported certificates

- **none** - do not use SSL
- **import-other-certificate** - setup the certificates not imported yet, and ask this question again

Notes

Depending on current settings and answers to the previous questions, default values of following questions may be different. Some questions may disappear if they become redundant

Example

To configure HotSpot on ether1 interface (which is already configured with address of 192.0.2.1/25), and adding user admin with password rubbish:

```
[admin@MikroTik] > ip hotspot setup
hotspot interface: ether1
local address of network: 192.0.2.1/24
masquerade network: yes
address pool of network: 192.0.2.2-192.0.2.126
select certificate: none
ip address of smtp server: 0.0.0.0
dns servers: 192.0.2.254
dns name: hs.example.net
name of local hotspot user: admin
password for the user: rubbish
[admin@MikroTik] >
```

HotSpot Interface Setup

Home menu level: ***/ip hotspot***

Description

HotSpot system is put on individual interfaces. You can run completely different HotSpot configurations on different interfaces

Property Description

HTTPS (*read-only: flag*) - whether the HTTPS service is actually running on the interface (i.e., it is set up in the server profile, and a valid certificate is imported in the router)

address-pool (*namenone*; default: **none**) - IP address pool name for performing one-to-one NAT. You can choose not to use the one-to-one NAT

- **none** - do not perform one-to-one NAT for the clients of this HotSpot interface

addresses-per-mac (*integerunlimited*; default: **2**) - number of IP addresses allowed to be bind with any particular MAC address (it is a small chance to reduce denial of service attack based on taking over all free IP addresses in the address pool). Not available if address-pool is set to none

- **unlimited** - number of IP addresses per one MAC address is not limited

idle-timeout (*timenone*; default: **00:05:00**) - idle timeout (maximal period of inactivity) for unauthorized clients. It is used to detect, that client is not using outer networks (e.g. Internet), i.e., there is NO TRAFFIC coming from that client and going through the router. Reaching the timeout, user will be dropped of the host list, and the address used buy the user will be freed

- **none** - do not timeout idle users

interface (*name*) - interface to run HotSpot on

ip-of-dns-name (*read-only: IP address*) - IP address of the HotSpot gateway's DNS name set in the HotSpot interface profile

keepalive-timeout (*timenone*; default: **none**) - keepalive timeout for unauthorized clients. Used to detect, that the computer of the client is alive and reachable. If check will fail during this period, user will be dropped of the host list, and the address used buy the user will be freed

- **none** - do not timeout unreachable users

profile (*name*; default: **default**) - default HotSpot profile for the interface

Command Description

reset-html (*name*) - overwrite the existing HotSpot servlet with the original HTML files. It is used if you have changed the servlet and it is not working after that

Notes

addresses-per-mac property works only if address pool is defined. Also note that in case you are authenticating users connected through a router, than all the IP addresses will seem to have come from one MAC address.

Example

To add HotSpot system to the **local** interface, allowing the system to do one-to-one NAT for each client (addresses from the **HS-real** address pool will be used for the NAT):

```
[admin@MikroTik] ip hotspot> add interface=local address-pool=HS-real
[admin@MikroTik] ip hotspot> print
Flags: X - disabled, I - invalid, S - HTTPS
 #   NAME                            INTERFACE    ADDRESS-POOL PROFILE IDLE-TIMEOUT
 0   hs-local                        local        HS-real      default 00:05:00
[admin@MikroTik] ip hotspot>
```

HotSpot Server Profiles

Home menu level: ***/ip hotspot profile***

Description

There may be various different HotSpot systems, defined as HotSpot Server Profiles, on the same gateway machine. One or more interfaces can be grouped into one server profile. There are very few settings for the servers on particular interfaces - most of the configuration is set in the server profiles. For example, it is possible to make completely different set of servlet pages for each server profile, and define different RADIUS servers for authentication.

Property Description

dns-name (*text*) - DNS name of the HotSpot server. This is the DNS name used as the name of the HotSpot server (i.e., it appears as the location of the login page). This name will automatically be added as a static DNS entry in the DNS cache

hotspot-address (*IP address*; default: **0.0.0.0**) - IP address for HotSpot service

html-directory (*text*; default: **hotspot**) - name of the directory (accessible with FTP), which stores the HTML servlet pages (when changed, the default pages are automatically copied into specified directory if it does not exist already)

http-cookie-lifetime (*time*; default: **3d**) - validity time of HTTP cookies

http-proxy (*IP address*; default: **0.0.0.0**) - address of the proxy server the HotSpot service will use as a [parent] proxy server for all those requests intercepted by Universal Proxy system and not defined in the /ip proxy direct list. If not specified, the address defined in parent-proxy parameter of /ip proxy. If that is absent as well, the request will be resolved by the local proxy

login-by (*multiple choice: cookie | http-chap | http-pap | https | mac | trial*; default: **cookie,http-chap**) - which authentication methods to use

- **cookie** - use HTTP cookies to authenticate, without asking user credentials. Other method will be used in case the client does not have cookie, or the stored username and password pair are not valid anymore since the last authentication. May only be used together with other HTTP authentication methods (HTTP-PAP, HTTP-CHAP or HTTPS), as in the other case there would be no way for the cookies to be generated in the first place
- **http-chap** - use CHAP challenge-response method with MD5 hashing algorithm for hashing passwords. This way it is possible to avoid sending clear-text passwords over an insecure network. This is the default authentication method
- **http-pap** - use plain-text authentication over the network. Please note that in case this method will be used, your user passwords will be exposed on the local networks, so it will be possible to intercept them
- **https** - use encrypted SSL tunnel to transfer user communications with the HotSpot server. Note that in order this to work, a valid certificate must be imported into the router (see a separate manual on certificate management)
- **mac** - try to use client's MAC address first as its username. If the matching MAC address exists in the local user database or on the RADIUS server, the client will be authenticated without asking to fill the login form
- **trial** - does not require authentication for a certain amount of time

mac-auth-password (*text*) - if MAC authentication is used, this field can be used to specify password for the users to be authenticated by their MAC addresses

nas-port-type (*text*; default: **wireless-802.11**) - NAS-Port-Type attribute value to be sent to the RADIUS server

radius-accounting (yes | no; default: **yes**) - whether to send RADIUS server accounting information on each user once in a while (the "while" is defined in the radius-interim-update property)

radius-default-domain (*text*; default: **""**) - default domain to use for RADIUS requests. It allows to select different RADIUS servers depending on HotSpot server profile, but may be handful for single RADIUS server as well.

radius-interim-update (*timereceived*; default: **received**) - how often to sent cumulative accounting reports.

- **0s** - same as received
- **received** - use whatever value received from the RADIUS server

radius-location-id (*text*) - Raduis-Location-Id attribute value to be sent to the RADIUS server

radius-location-name (*text*) - Raduis-Location-Name attribute value to be sent to the RADIUS server

rate-limit (*text*; default: **""**) - Rate limitation in form of rx-rate[/tx-rate] [rx-burst-rate[/tx-burst-rate] [rx-burst-threshold[/tx-burst-threshold] [rx-burst-time[/tx-burst-time]]]] [priority] [rx-rate-min[/tx-rate-min]] from the point of view of the router (so "rx" is client upload, and "tx" is client download). All rates should be numbers with optional 'k' (1,000s) or 'M' (1,000,000s). If tx-rate is not specified, rx-rate is as tx-rate too. Same goes for tx-burst-rate and tx-burst-threshold and tx-burst-time. If both rx-burst-threshold and tx-burst-threshold are not specified (but burst-rate is specified), rx-rate and tx-rate is used as burst thresholds. If both rx-burst-time and tx-burst-time are not specified, 1s is used as default. rx-rate-min and tx-rate min are the values of limit-at properties

smtp-server (*IP address*; default: **0.0.0.0**) - default SMTP server to be used to redirect unconditionally all user SMTP requests to

split-user-domain (yes | no; default: **no**) - whether to split username from domain name when the username is given in "user@domain" or in "domain\user" format

ssl-certificate (*namenone*; default: **none**) - name of the SSL certificate to use for HTTPS authentication. Not used for other authentication methods

trial-uptime (*timetime*; default: **30m/1d**) - is used only when authentication method is trial. Specifies the amount of time the user identified by MAC address can use HotSpot services without authentication and the time, that has to pass that the user is allowed to use HotSpot services again

trial-user-profile (*name*; default: **default**) - is used only only when authentication method is trial. Specifies user profile, that trial users will use

use-radius (yes | no; default: **no**) - whether to use RADIUS to authenticate HotSpot users

Notes

If **dns-name** property is not specified, **hotspot-address** is used instead. If **hotspot-address** is also absent, then both are to be detected automatically.

In order to use RADIUS authentication, the **/radius** menu must be set up properly.

Trial authentication method should always be used together with one of the other authentication methods.

Example

HotSpot User Profiles

Home menu level: ***/ip hotspot user profile***

Description

Article moved to: HotSpot AAA section

HotSpot Users

Home menu level: ***/ip hotspot user***

Description

Article moved to: HotSpot AAA section

HotSpot Active Users

Home menu level: ***/ip hotspot active***

Description

Article moved to: HotSpot AAA section

HotSpot Cookies

Home menu level: ***/ip hotspot cookie***

Description

Cookies can be used for authentication in the Hotspot service

Property Description

domain (*read-only: text*) - domain name (if split from username)

expires-in (*read-only: time*) - how long is the cookie valid

mac-address (*read-only: MAC address*) - user's MAC address

user (*read-only: name*) - username

Notes

There can be multiple cookies with the same MAC address. For example, there will be a separate cookie for each web browser on the same computer.

Cookies can expire - that's the way how it is supposed to be. Default validity time for cookies is **3** days (72 hours), but it can be changed for each individual HotSpot server profile, for example :

```
/ip hotspot profile set default http-cookie-lifetime=1d
```

Example

To get the list of valid cookies:

```
[admin@MikroTik] ip hotspot cookie> print
```

```
  # USER               DOMAIN            MAC-ADDRESS       EXPIRES-IN
  0 ex                                   01:23:45:67:89:AB 23h54m16s
[admin@MikroTik] ip hotspot cookie>
```

HTTP-level Walled Garden

Home menu level: ***/ip hotspot walled-garden***

Description

Walled garden is a system which allows unauthorized use of some resources, but requires authorization to access other resources. This is useful, for example, to give access to some general information about HotSpot service provider or billing options.

This menu only manages Walled Garden for HTTP and HTTPS protocols. Other protocols can also be included in Walled Garden, but that is configured elsewhere (in **/ip hotspot walled-garden ip**; see the next section of this manual for details)

Property Description

action (*allow* | *deny*; default: **allow**) - action to undertake if a request matches the rule:

- **allow** - allow the access to the page without prior authorization
- **deny** - authorization is required to access this page

dst-address (*read-only: IP address*) - IP address of the destination web server (installed by IP-level walled garden)

dst-host (*wildcard*; default: "") - domain name of the destination web server

dst-port (*integer*; default: "") - the TCP port a client has send the request to

hits (*read-only: integer*) - how many times has this rule been used

method (*text*) - HTTP method of the request

path (*wildcard*; default: "") - the path of the request

server (*name*) - name of the HotSpot server this rule applies to

src-address (*IP address*) - IP address of the user sending the request

Notes

Wildcard properties (**dst-host** and **dst-path**) match a complete string (i.e., they will not match "example.com" if they are set to "example"). Available wildcards are '*' (match any number of any characters) and '?' (match any one character). Regular expressions are also accepted here, but for property to be treated as a regular expression, it should start with a colon (':').

Small hits in using regular expressions:

- **** symbol sequence is used to enter **** character in console
- **\.** pattern means **.** only (in regular expressions single dot in pattern means any symbol)
- to show that no symbols are allowed before the given pattern, we use **^** symbol at the beginning of the pattern

- to specify that no symbols are allowed after the given pattern, we use **$** symbol at the end of the pattern

You can not use **path** property for HTTPS requests as router can not (and should not - that is what the HTTPS protocol was made for!) decrypt the request.

IP-level walled garden, described in the following section, makes dynamic entries here. In this case, the **dst-address** property is filled (it is empty otherwise).

Example

To allow unauthorized requests to the **www.example.com** domain's **/paynow.html** page:

```
[admin@MikroTik] ip hotspot walled-garden> add path="/paynow.html" \
\... dst-host="www.example.com"
[admin@MikroTik] ip hotspot walled-garden> print detail
Flags: X - disabled, D - dynamic
 0   dst-host="www.example.com" path="/paynow.html" action=allow
[admin@MikroTik] ip hotspot walled-garden>
```

IP-level Walled Garden

Home menu level: ***/ip hotspot walled-garden ip***

Description

This menu is manages Walled Garden for generic IP requests. See the previous section for managing HTTP and HTTPS protocol specific properties (like the actual DNS name, HTTP method and path used in requests).

Property Description

action (*accept* | *drop* | *reject*; default: **accept**) - action to undertake if a packet matches the rule:

- **accept** - allow the access to the page without prior authorization
- **drop** - the authorization is required to access this page
- **reject** - the authorization is required to access this page, in case the page will be accsessed withot authorization ICMP reject message host-unreachable will be generated

dst-address (*IP address*) - IP address of the destination web server

dst-host (*text*; default: **""**) - domain name of the destination web server (this is not a regular expression or a wildcard of any kind). The DNS name specified is resolved to a list of IP addresses when the rule is added, and all those IP addresses are used

dst-port (*integer*; default: **""**) - the TCP or UDP port (protocol MUST be specified explicitly in the protocol property) a client has send the request to

protocol (*integerddp* | *egp* | *encap* | *ggp* | *gre* | *hmp* | *icmp* | *idpr-cmtp* | *igmp* | *ipencap* | *ipip* | *ipsec-ah* | *ipsec-esp* | *iso-tp4* | *ospf* | *pup* | *rdp* | *rspf* | *st* | *tcp* | *udp* | *vmtp* | *xns-idp* | *xtp*) - IP protocol name

server (*name*) - name of the HotSpot server this rule applied to

src-address (*IP address*) - IP address of the user sending the request

Example

One-to-one NAT static address bindings

Home menu level: ***/ip hotspot ip-binding***

Description

You can setup NAT translations statically based on either the original IP address (or IP network), or the original MAC address. You can also allow some addresses to bypass HotSpot authentication (i.e., they will be able work without having to log in to the network first) and completely block some addresses.

Property Description

address (*IP addressnetmask*; default: "") - the original IP address or network of the client

mac-address (*MAC address*; default: "") - the source MAC address of the client

server (*nameall*; default: **all**) - the name of the server the client is connecting to

to-address (*IP address*; default: "") - IP address to translate the original client address to. If address property is given as network, this is the starting address for the translation (i.e., the first address is translated to to-address, address + 1 to to-address + 1, and so on)

type (*regular* | *bypassed* | *blocked*) - type of the static binding entry

- **regular** - perform a one-to-one NAT translation according to the values set in this entry
- **bypassed** - perform the translation, but exclude the client from having to log in to the HotSpot system
- **blocked** - the translation will not be preformed, and all packets from the host will be dropped

Notes

This is an ordered list, so you can put more specific entries on the top of the list for them to override more common rules that appear lower. You can even put an entry with **0.0.0.0/0** address at the end of the list to make the desired action default for those addresses that will not match any other entry.

Active Host List

Home menu level: ***/ip hotspot host***

Description

This menu shows all active network hosts that are connected to the HotSpot gateway. This list includes all one-to-one NAT translations

Property Description

address (*read-only: IP address*) - the original IP address of the client

authorized (*read-only: flag*) - whether the client is successfully authenticated by the HotSpot system

bridge-port (*read-only: name*) - the actual physical interface, which the host is connected to. This is used when HotSpot service is put on a bridge interface to determine the host's actual port within the bridge.

bypassed (*read-only: flag*) - whether the client does not need to be authorized by the HotSpot system

bytes-in (*read-only: integer*) - how many bytes did the router receive from the client

bytes-out (*read-only: integer*) - how many bytes did the router send to the client

found-by (*read-only: text*) - how was this host discovered (first packet type, sender, recipient)

host-dead-time (*read-only: time*) - how long has the router not received any packets (including ARP replies, keepalive replies and user traffic) from this host

idle-time (*read-only: time*) - the amount of time has the user been idle

idle-timeout (*read-only: time*) - the exact value of idle-timeout that applies to this user. This property shows how long should the user stay idle for it to be logged off automatically

keepalive-timeout (*read-only: time*) - the exact value of keepalive-timeout that applies to this user. This property shows how long should the user's computer stay out of reach for it to be logged off automatically

mac-address (*read-only: MAC address*) - the actual MAC address of the user

packets-in (*read-only: integer*) - how many packets did the router receive from the client

packets-out (*read-only: integer*) - how many packets did the router send to the client

server (*read-only: name*) - name of the server, which the host is connected to

static (*read-only: flag*) - whether this translation has been taken from the static IP binding list

to-address (*read-only: IP address*) - what address is the original IP address of the host translated to

uptime (*read-only: time*) - current session time of the user (i.e., how long has the user been in the active host list)

Command Description

make-binding - copy a dynamic entry from this list to the static IP bindings list (*name*) - item number (*text*) - custom comment to the static entry to be created (*regular* | *bypassed* | *blocked*) - the type of the static entry

Service Port

Home menu level: ***/ip hotspot service-port***

Description

Just like for classic NAT, the HotSpot embedded one-to-one NAT 'breaks' some protocols that are incompatible with address translation. To leave these protocols consistent, helper modules must be used. For the one-to-one NAT the only such a module is for FTP protocol.

Property Description

name (*read-only: name*) - protocol name

ports (*read-only: integer*) - list of the ports on which the protocol is working

Example

To set the FTP protocol uses both 20 and 21 TCP port:

```
[admin@MikroTik] ip hotspot service-port> print
Flags: X - disabled
  #   NAME                                                                PORTS
  0   ftp                                                                 21
[admin@MikroTik] ip hotspot service-port> set ftp ports=20,21
[admin@MikroTik] ip hotspot service-port> print
Flags: X - disabled
  #   NAME                                                                PORTS
  0   ftp                                                                 20
                                                                          21
[admin@MikroTik] ip hotspot service-port>
```

Customizing HotSpot: Firewall Section

Description

Apart from the obvious dynamic entries in the **/ip hotspot** submenu itself (like hosts and active users), some additional rules are added in the firewall tables when activating a HotSpot service. Unlike RouterOS version 2.8, there are relatively few firewall rules added in the firewall as the main job is made by the one-to-one NAT algorithm.

NAT rules

From **/ip firewall nat print dynamic** command, you can get something like this (comments follow after each of the rules):

```
 0 D chain=dstnat action=jump jump-target=hotspot hotspot=from-client
```

Putting all HotSpot-related tasks for packets from all HotSpot clients into a separate chain.

```
 1 I chain=hotspot action=jump jump-target=pre-hotspot
```

Any actions that should be done before HotSpot rules apply, should be put in the **pre-hotspot** chain. This chain is under full administrator control and does not contain any rules set by the system, hence the invalid jump rule (as the chain does not have any rules by default).

```
 2 D chain=hotspot action=redirect to-ports=64872 dst-port=53 protocol=udp
 3 D chain=hotspot action=redirect to-ports=64872 dst-port=53 protocol=tcp
```

Redirect all DNS requests to the HotSpot service. The 64872 port provides DNS service for all HotSpot users. If you want HotSpot server to listen also to another port, add rules here the same way, changing **dst-port** property.

```
 4 D chain=hotspot action=redirect to-ports=64873 hotspot=local-dst dst-port=80
     protocol=tcp
```

Redirect all HTTP login requests to the HTTP login servlet. The 64873 is HotSpot HTTP servlet port.

```
 5 D chain=hotspot action=redirect to-ports=64875 hotspot=local-dst dst-port=443
     protocol=tcp
```

Redirect all HTTPS login requests to the HTTPS login servlet. The 64875 is HotSpot HTTPS servlet port.

```
 6 D chain=hotspot action=jump jump-target=hs-unauth hotspot=!auth protocol=tcp
```

All other packets except DNS and login requests from unauthorized clients should pass through the **hs-unauth** chain.

```
 7 D chain=hotspot action=jump jump-target=hs-auth hotspot=auth protocol=tcp
```

And packets from the authorized clients - through the **hs-auth** chain.

```
 8 D ;;; www.mikrotik.com
     chain=hs-unauth action=return dst-address=66.228.113.26 dst-port=80 protocol=tcp
```

First in the **hs-unauth** chain is put everything that affects TCP protocol in the **/ip hotspot walled-garden ip** submenu (i.e., everything where either protocol is not set, or set to TCP). Here we are excluding www.mikrotik.com from being redirected to the login page.

```
 9 D chain=hs-unauth action=redirect to-ports=64874 dst-port=80 protocol=tcp
```

All other HTTP requests are redirected to the Walled Garden proxy server which listens the 64874 port. If there is an **allow** entry in the **/ip hotspot walled-garden** menu for an HTTP request, it is being forwarded to the destination. Otherwise, the request will be automatically redirected to the HotSpot login servlet (port 64873).

```
10 D chain=hs-unauth action=redirect to-ports=64874 dst-port=3128 protocol=tcp
11 D chain=hs-unauth action=redirect to-ports=64874 dst-port=8080 protocol=tcp
```

HotSpot by default assumes that only these ports may be used for HTTP proxy requests. These two entries are used to "catch" client requests to unknown proxies (you can add more rules here for other ports). I.e., to make it possible for the clients with unknown proxy settings to work with the HotSpot system. This feature is called "Universal Proxy". If it is detected that a client is using some proxy server, the system will automatically mark that packets with the **http** hotspot mark to work around the unknown proxy problem, as we will see later on. Note that the port used (64874) is the same as for HTTP requests in the rule #9 (so both HTTP and HTTP proxy requests are processed by the same code).

```
12 D chain=hs-unauth action=redirect to-ports=64875 dst-port=443 protocol=tcp
```

HTTPS proxy is listening on the 64875 port.

```
13 I chain=hs-unauth action=jump jump-target=hs-smtp dst-port=25 protocol=tcp
```

Redirect for SMTP protocol may also be defined in the HotSpot configuration. In case it is, a redirect rule will be put in the **hs-smtp** chain. This is done so that users with unknown SMTP configuration would be able to send their mail through the service provider's (your) SMTP server instead of going to the [possibly unavailable outside their network of origin] SMTP server users have configured on their computers. The chain is empty by default, hence the invalid jump rule.

```
14 D chain=hs-auth action=redirect to-ports=64874 hotspot=http protocol=tcp
```

Providing HTTP proxy service for authorized users. Authenticated user requests may need to be subject to transparent proxying (the "Universal Proxy" technique and advertisement feature). This **http** mark is put automatically on the HTTP proxy requests to the servers detected by the HotSpot HTTP proxy (the one that is listening on the 64874 port) as HTTP proxy requests for unknown proxy servers. This is done so that users that have some proxy settings would use the HotSpot gateway instead of the [possibly unavailable outside their network of origin] proxy server users have configured in their computers. This mark is also applied when advertisement is due to be shown to the user, as well as on any HTTP requests done form the users whose profile is configured to transparently proxy their requests.

```
15 I chain=hs-auth action=jump jump-target=hs-smtp dst-port=25 protocol=tcp
```

Providing SMTP proxy for authorized users (the same as in rule #13).

Packet filter rules

From **/ip firewall filter print dynamic** command, you can get something like this (comments follow after each of the rules):

```
 0 D chain=forward action=jump jump-target=hs-unauth hotspot=from-client,!auth
```

Any packet that traverse the router from an unauthorized client will be sent to the **hs-unauth** chain. The **hs-unauth** implements the IP-based Walled Garden filter.

```
 1 D chain=forward action=jump jump-target=hs-unauth-to hotspot=to-client,!auth
```

Everything that comes to clients through the router, gets redirected to another chain, called **hs-unauth-to**. This chain should reject unauthorized requests to the clients.

```
 2 D chain=input action=jump jump-target=hs-input hotspot=from-client
```

Everything that comes from clients to the router itself, gets to yet another chain, called **hs-input**.

```
 3 I chain=hs-input action=jump jump-target=pre-hs-input
```

Before proceeding with [predefined] dynamic rules, the packet gets to the administratively controlled **pre-hs-input** chain, which is empty by default, hence the invalid state of the jump rule.

```
  4 D chain=hs-input action=accept dst-port=64872 protocol=udp
  5 D chain=hs-input action=accept dst-port=64872-64875 protocol=tcp
```

Allow client access to the local authentication and proxy services (as described earlier).

```
  6 D chain=hs-input action=jump jump-target=hs-unauth hotspot=!auth
```

All other traffic from unauthorized clients to the router itself will be treated the same way as the traffic traversing the routers.

```
  7 D chain=hs-unauth action=return protocol=icmp
  8 D ;;; www.mikrotik.com
      chain=hs-unauth action=return dst-address=66.228.113.26 dst-port=80 protocol=tcp
```

Unlike NAT table where only TCP-protocol related Walled Garden entries were added, in the packet filter **hs-unauth** chain is added everything you have set in the **/ip hotspot walled-garden ip** menu. That is why although you have seen only one entry in the NAT table, there are two rules here.

```
  9 D chain=hs-unauth action=reject reject-with=tcp-reset protocol=tcp
 10 D chain=hs-unauth action=reject reject-with=icmp-net-prohibited
```

Everything else that has not been while-listed by the Walled Garden will be rejected. Note usage of TCP Reset for rejecting TCP connections.

```
 11 D chain=hs-unauth-to action=return protocol=icmp
 12 D ;;; www.mikrotik.com
      chain=hs-unauth-to action=return src-address=66.228.113.26 src-port=80
 protocol=tcp
```

Same action as in rules #7 and #8 is performed for the packets destined to the clients (chain **hs-unauth-to**) as well.

```
 13 D chain=hs-unauth-to action=reject reject-with=icmp-host-prohibited
```

Reject all packets to the clients with ICMP reject message.

Customizing HotSpot: HTTP Servlet Pages

Description

You can create a completely different set of servlet pages for each HotSpot server you have, specifying the directory it will be stored in **html-directory** property of a HotSpot server profile (**/ip hotspot profile**). The default servlet pages are copied in the directory of your choice right after you create the profile. This directory can be accessed by connecting to the router with an FTP client. You can modify the pages as you like using the information from this section of the manual. Note that it is suggested to edit the files manually, as automated HTML editing tools may corrupt the pages by removing variables or other vital parts.

Available Servlet Pages

Main HTML servlet pages, which are shown to user:

- **redirect.html** - redirects user to another url (for example, to login page)
- **login.html** - login page shown to a user to ask for username and password. This page may take the following parameters:
  - **username** - username
  - **password** - either plain-text password (in case of PAP authentication) or MD5 hash of chap-id variable, password and CHAP challenge (in case of CHAP authentication). This value is used as e-mail address for trial users
  - **dst** - original URL requested before the redirect. This will be opened on successfull login
  - **popup** - whether to pop-up a status window on successfull login
  - **radius<id>** - send the attribute identified with <id> in text string form to the RADIUS server (in case RADIUS authentication is used; lost otherwise)
  - **radius<id>u** - send the attribute identified with <id> in unsigned integer form to the RADIUS server (in case RADIUS authentication is used; lost otherwise)
  - **radius<id>-<vnd-id>** - send the attribute identified with <id> and vendor ID <vnd-id> in text string form to the RADIUS server (in case RADIUS authentication is used; lost otherwise)
  - **radius<id>-<vnd-id>u** - send the attribute identified with <id> and vendor ID <vnd-id> in unsigned integer form to the RADIUS server (in case RADIUS authentication is used; lost otherwise)
- **md5.js** - JavaScript for MD5 password hashing. Used together with **http-chap** login method
- alogin.html - page shown after client has logged in. It pops-up status page and redirects browser to originally requested page (before he/she was redirected to the HotSpot login page)
- **status.html** - status page, shows statistics for the client. It is also able to display advertisements automatically
- **logout.html** - logout page, shown after user is logged out. Shows final statistics about the finished session. This page may take the following additional parameters:
  - **erase-cookie** - whether to erase cookies from the HotSpot server on logout (makes impossible to log in with cookie next time from the same browser, might be useful in multiuser environments)
- **error.html** - error page, shown on fatal errors only
  - **rlogin.html** - page, which redirects client from some other URL to the login page, if authorization of the client is required to access that URL
  - **rstatus.html** - similarly to rlogin.html, only in case if the client is already logged in and the original URL is not known
  - **radvert.html** - redirects client to the scheduled advertisement link

- **flogin.html** - shown instead of login.html, if some error has happened (invalid username or password, for example)
- **fstatus.html** - shown instead of redirect, if status page is requested, but client is not logged in
- **flogout.html** - shown instead of redirect, if logout page is requested, but client is not logged in

Some other pages are available as well, if more control is needed:

Serving Servlet Pages

The HotSpot servlet recognizes 5 different request types:

1. request for a remote host
   - if user is logged in and advertisement is due to be displayed, **radvert.html** is displayed. This page makes redirect to the scheduled advertisment page
   - if user is logged in and advertisement is not scheduled for this user, the requested page is served
   - if user is not logged in, but the destination host is allowed by walled garden, then the request is also served
   - if user is not logged in, and the destination host is disallowed by walled garden, **rlogin.html** is displayed; if **rlogin.html** is not found, **redirect.html** is used to redirect to the login page
2. request for "/" on the HotSpot host
   - if user is logged in, **rstatus.html** is displayed; if **rstatus.html** is not found, **redirect.html** is used to redirect to the status page
   - if user is not logged in, **rlogin.html** is displayed; if **rlogin.html** is not found, **redirect.html** is used to redirect to the login page
3. request for "/login" page
   - if user has successfully logged in (or is already logged in), **alogin.html** is displayed; if **alogin.html** is not found, **redirect.html** is used to redirect to the originally requested page or the status page (in case, original destination page was not given)
   - if user is not logged in (username was not supplied, no error message appeared), **login.html** is showed
   - if login procedure has failed (error message is supplied), **flogin.html** is displayed; if **flogin.html** is not found, **login.html** is used
   - in case of fatal errors, **error.html** is showed
4. request for "/status" page
   - if user is logged in, **status.html** is displayed
   - if user is not logged in, **fstatus.html** is displayed; if **fstatus.html** is not found, **redirect.html** is used to redirect to the login page
5. request for '/logout' page

- if user is logged in, **logout.html** is displayed
- if user is not logged in, **flogout.html** is displayed; if **flogout.html** is not found, **redirect.html** is used to redirect to the login page

Note that if it is not possible to meet a request using the pages stored on the router's FTP server, Error 404 is displayed

There are many possibilities to customize what the HotSpot authentication pages look like:

- The pages are easily modifiable. They are stored on the router's FTP server in the directory you choose for the respective HotSpot server profile.
- By changing the variables, which client sends to the HotSpot servlet, it is possible to reduce keyword count to one (username or password; for example, the client's MAC address may be used as the other value) or even to zero (License Agreement; some predefined values general for all users or client's MAC address may be used as username and password)
- Registration may occur on a different server (for example, on a server that is able to charge Credit Cards). Client's MAC address may be passed to it, so that this information need not be written in manually. After the registration, the server should change RADIUS database enabling client to log in for some amount of time.

To insert variable in some place in HTML file, the $(var_name) syntax is used, where the "var_name" is the name of the variable (without quotes). This construction may be used in any HotSpot HTML file accessed as '/', '/login', '/status' or '/logout', as well as any text or HTML (**.txt**, **.htm** or **.html**) file stored on the HotSpot server (with the exception of traffic counters, which are available in status page only, and **error**, **error-orig**, **chap-id**, **chap-challenge** and **popup** variables, which are available in login page only). For example, to show a link to the login page, following construction can be used:

```
<a href="$(link-login)">login</a>
```

Variables

All of the Servlet HTML pages use variables to show user specific values. Variable names appear only in the HTML source of the servlet pages - they are automatically replaced with the respective values by the HotSpot Servlet. For most variables there is an example of their possible value included in brackets. All the described variables are valid in all servlet pages, but some of them just might be empty at the time they are accesses (for example, there is no uptime before a user has logged in).

- Common server variables:
  - **hostname** - DNS name or IP address (if DNS name is not given) of the HotSpot Servlet ("hotspot.example.net")
  - **identity** - RouterOS identity name ("MikroTik")
  - **login-by** - authentication method used by user
  - **plain-passwd** - a "yes/no" representation of whether HTTP-PAP login method is allowed ("no")
  - **server-address** - HotSpot server address ("10.5.50.1:80")
  - **ssl-login** - a "yes/no" representation of whether HTTPS method was used to access that servlet page ("no")

- **server-name** - HotSpot server name (set in the /ip hotspot menu, as the name property)

- Links:
  - **link-login** - link to login page including original URL requested ("http://10.5.50.1/login?dst=http://www.example.com/")
  - **link-login-only** - link to login page, not including original URL requested ("http://10.5.50.1/login")
  - **link-logout** - link to logout page ("http://10.5.50.1/logout")
  - **link-status** - link to status page ("http://10.5.50.1/status")
  - **link-orig** - original URL requested ("http://www.example.com/")

- General client information
  - **domain** - domain name of the user ("example.com")
  - **interface-name** - physical HotSpot interface name (in case of bridged interfaces, this will return the actual bridge port name)
  - **ip** - IP address of the client ("10.5.50.2")
  - **logged-in** - "yes" if the user is logged in, otherwise - "no" ("yes")
  - **mac** - MAC address of the user ("01:23:45:67:89:AB")
  - **trial** - a "yes/no" representation of whether the user has access to trial time. If users trial time has expired, the value is "no"
  - **username** - the name of the user ("John")

- User status information:
  - **idle-timeout** - idle timeout ("20m" or "" if none)
  - **idle-timeout-secs** - idle timeout in seconds ("88" or "0" if there is such timeout)
  - **limit-bytes-in** - byte limit for send ("1000000" or "---" if there is no limit)
  - **limit-bytes-out** - byte limit for receive ("1000000" or "---" if there is no limit)
  - **refresh-timeout** - status page refresh timeout ("1m30s" or "" if none)
  - **refresh-timeout-secs** - status page refresh timeout in seconds ("90s" or "0" if none)
  - **session-timeout** - session time left for the user ("5h" or "" if none)
  - **session-timeout-secs** - session time left for the user, in seconds ("3475" or "0" if there is such timeout)
  - **session-time-left** - session time left for the user ("5h" or "" if none)
  - **session-time-left-secs** - session time left for the user, in seconds ("3475" or "0" if there is such timeout)
  - **uptime** - current session uptime ("10h2m33s")
  - **uptime-secs** - current session uptime in seconds ("125")

- Traffic counters, which are available only in the status page:
  - **bytes-in** - number of bytes received from the user ("15423")
  - **bytes-in-nice** - user-friendly form of number of bytes received from the user ("15423")
  - **bytes-out** - number of bytes sent to the user ("11352")

  - **bytes-out-nice** - user-friendly form of number of bytes sent to the user ("11352")
  - **packets-in** - number of packets received from the user ("251")
  - **packets-out** - number of packets sent to the user ("211")
  - **remain-bytes-in** - remaining bytes until limit-bytes-in will be reached ("337465" or "---" if there is no limit)
  - **remain-bytes-out** - remaining bytes until limit-bytes-out will be reached ("124455" or "---" if there is no limit)

- Miscellaneous variables
  - **session-id** - value of 'session-id' parameter in the last request
  - **var** - value of 'var' parameter in the last request
  - **error** - error message, if something failed ("invalid username or password")
  - **error-orig** - original error message (without translations retrieved from errors.txt), if something failed ("invalid username or password")
  - **chap-id** - value of chap ID ("\371")
  - **chap-challenge** - value of chap challenge ("\357\015\330\013\021\234\145\245\303\253\142\246\133\175\375\316")
  - **popup** - whether to pop-up checkbox ("true" or "false")
  - **advert-pending** - whether an advertisement is pending to be displayed ("yes" or "no")

- RADIUS-related variables
  - **radius<id>** - show the attribute identified with <id> in text string form (in case RADIUS authentication was used; "" otherwise)
  - **radius<id>u** - show the attribute identified with <id> in unsigned integer form (in case RADIUS authentication was used; "0" otherwise)
  - **radius<id>-<vnd-id>** - show the attribute identified with <id> and vendor ID <vnd-id> in text string form (in case RADIUS authentication was used; "" otherwise)
  - **radius<id>-<vnd-id>u** - show the attribute identified with <id> and vendor ID <vnd-id> in unsigned integer form (in case RADIUS authentication was used; "0" otherwise)

Working with variables

`$(if <var_name>)` statements can be used in theses pages. Following content will be included, if value of <var_name> will not be an empty string. It is an equivalent to `$(if <var_name> != "")` It is possible to compare on equivalence as well: `$(if <var_name> == <value>)` These statements have effect until `$(elif <var_name>)`, `$(else)` or `$(endif)`. In general case it looks like this:

```
some content, which will always be displayed
$(if username == john)
Hey, your username is john
$(elif username == dizzy)
Hello, Dizzy! How are you? Your administrator.
$(elif ip == 10.1.2.3)
You are sitting at that crappy computer, which is damn slow...
$(elif mac == 00:01:02:03:04:05)
This is an ethernet card, which was stolen few months ago...
$(else)
I don't know who you are, so lets live in peace.
```

```
$(endif)
other content, which will always be displayed
```

Only one of those expressions will be shown. Which one - depends on values of those variables for each client.

Customizing Error Messages

All error messages are stored in the **errors.txt** file within the respective HotSpot servlet directory. You can change and translate all these messages to your native language. To do so, edit the **errors.txt** file. You can also use variables in the messages. All instructions are given in that file.

Multiple Versions of HotSpot Pages

Multiple HotSpot page sets for the same HotSpot server are supported. They can be chosen by user (to select language) or automatically by JavaScript (to select PDA/regular version of HTML pages).

To utilize this feature, create subdirectories in HotSpot HTML directory, and place those HTML files, which are different, in that subdirectory. For example, to translate everything in Latvian, subdirectory "lv" can be created with login.html, logout.html, status.html, alogin.html, radvert.html and errors.txt files, which are translated into Latvian. If the requested HTML page can not be found in the requested subdirectory, the corresponding HTML file from the main directory will be used. Then main login.html file would contain link to "/lv/login?dst=$(link-orig-esc)", which then displays Latvian version of login page: `<a href="/lv/login?dst=$(link-orig-esc)">Latviski</a>` . And Latvian version would contain link to English version: `<a href="/login?dst=$(link-orig-esc)">English</a>`

Another way of referencing directories is to specify 'target' variable:

```
        <a href="$(link-login-only)?dst=$(link-orig-esc)&target=lv">Latviski</a>
        <a href="$(link-login-only)?dst=$(link-orig-esc)&target=%2F">English</a>
```

After preferred directory has been selected (for example, "lv"), all links to local HotSpot pages will contain that path (for example, `$(link-status) = "http://hotspot.mt.lv/lv/status"`). So, if all HotSpot pages reference links using "$(link-xxx)" variables, then no more changes are to be made - each client will stay within the selected directory all the time.

Notes

If you want to use HTTP-CHAP authentication method it is supposed that you include the **doLogin()** function (which references to the **md5.js** which must be already loaded) before the **Submit** action of the login form. Otherwise, CHAP login will fail.

The resulting password to be sent to the HotSpot gateway in case of HTTP-CHAP method, is formed MD5-hashing the concatenation of the following: chap-id, the password of the user and chap-challenge (in the given order)

In case variables are to be used in link directly, then they must be escaped accordingly. For example, in login page, <a href="https://login.example.com/login?mac=$(mac)&user=$(username)">link</a> will not work as intended, if username will be "123&456=1 2". In this case instead of $(user), its escaped version must be used: $(user-esc): <a href="https://login.server.serv/login?mac=$(mac-esc)&user=$(user-esc)">link</a>. Now the same username will be converted to "123%26456%3D1+2", which is the valid representation of "123&456=1 2" in URL. This trick may be used with any variables, not only with $(username).

There is a boolean parameter "erase-cookie" to the logout page, which may be either "on" or "true" to delete user cookie on logout (so that the user would not be automatically logged on when he/she opens a browser next time.

Example

With basic HTML language knowledge and the examples below it should be easy to implement the ideas described above.

- To provide predefined value as username, in login.html change:

```
<type="text" value="$(username)>
```

to this line:

```
<input type="hidden" name="username" value="hsuser">
```

(where **hsuser** is the username you are providing)

- To provide predefined value as password, in login.html change:

```
<input type="password">
```

to this line:

```
<input type="hidden" name="password" value="hspass">
```

(where **hspass** is the password you are providing)

- To send client's MAC address to a registration server in form of:

```
https://www.example.com/register.html?mac=XX:XX:XX:XX:XX:XX
```

change the Login button link in login.html to:

```
https://www.example.com/register.html?mac=$(mac)
```

(you should correct the link to point to your server)

- To show a banner after user login, in alogin.html after

```
$(if popup == 'true')
```

add the following line:

```
open('http://www.example.com/your-banner-page.html', 'my-banner-name','');
```

(you should correct the link to point to the page you want to show)

- To choose different page shown after login, in login.html change:

```
<input type="hidden" name="dst" value="$(link-orig)">
```

to this line:

```
<input type="hidden" name="dst" value="http://www.example.com">
```

(you should correct the link to point to your server)

- To erase the cookie on logoff, in the page containing link to the logout (for example, in status.html) change:

```
open('$(link-logout)', 'hotspot_logout', ...
```

to this:

```
open('$(link-logout)?erase-cookie=on', 'hotspot_logout', ...
```

or alternatively add this line:

```
<input type="hidden" name="erase-cookie" value="on">
```

before this one:

```
<input type="submit" value="log off">
```

An another example is making HotSpot to authenticate on a remote server (which may, for example, perform creditcard charging):

- Allow direct access to the external server in walled-garden (either HTTP-based, or IP-based)
- Modify login page of the HotSpot servlet to redirect to the external authentication server. The external server should modify RADIUS database as needed
  Here is an example of such a login page to put on the HotSpot router (it is redirecting to https://auth.example.com/login.php, replace with the actual address of an external authentication server):

```
<html>
<title>...</title>
<body>
<form name="redirect" action="https://auth.example.com/login.php" method="post">
<input type="hidden" name="mac" value="$(mac)">
<input type="hidden" name="ip" value="$(ip)">
<input type="hidden" name="username" value="$(username)">
<input type="hidden" name="link-login" value="$(link-login)">
<input type="hidden" name="link-orig" value="$(link-orig)">
<input type="hidden" name="error" value="$(error)">
</form>
<script language="JavaScript">
<!--
        document.redirect.submit();
//-->
</script>
</body>
</html>
```

- The external server can log in a HotSpot client by redirecting it back to the original HotSpot servlet login page, specifying the correct username and password
  Here is an example of such a page (it is redirecting to https://hotspot.example.com/login, replace with the actual address of a HotSpot router; also, it is displaying www.mikrotik.com after successful login, replace with what needed):

```
<html>
<title>Hotspot login page</title>
<body>
<form name="login" action="https://hotspot.example.com/login" method="post">
<input type="text" name="username" value="demo">
<input type="password" name="password" value="none">
<input type="hidden" name="domain" value="">
<input type="hidden" name="dst" value="http://www.mikrotik.com/">
<input type="submit" name="login" value="log in">
</form>
</body>
</html>
```

- Hotspot will ask RADIUS server whether to allow the login or not. If not allowed, alogin.html page will be displayed (it can be modified to do anything). If not allowed, flogin.html (or login.html) page will be displayed, which will redirect client back to the external authentication server.
- Note: as shown in these examples, HTTPS protocol and POST method can be used to secure communications.

Possible Error Messages

Description

There are two kinds of errors: fatal and non-fatal. Fatal errors are shown on a separate HTML page called error.html. Non-fatal errors are basically indicating incorrect user actions and are shown on the login form.

General non-fatal errors:

- **You are not logged in** - trying to access the status page or log off while not logged in. Solution: log in
- **already authorizing, retry later** - authorization in progress. Client already has issued an authorization request which is not yet complete. Solution: wait for the current request to be completed, and then try again
- **chap-missing = web browser did not send challenge response (try again, enable JavaScript)** - trying to log in with HTTP-CHAP method using MD5 hash, but HotSpot server does not know the challenge used for the hash. This may happen if you use BACK buttons in browser; if JavaScript is not enabled in web browser; if login.html page is not valid; or if challenge value has expired on server (more than 1h of inactivity). Solution: instructing browser to reload (refresh) the login page usually helps if JavaScript is enabled and login.html page is valid
- **invalid username ($(username)): this MAC address is not yours** - trying to log in using a MAC address username different from the actual user's MAC address. Solution: no - users with usernames that look like a MAC address (eg., 12:34:56:78:9a:bc) may only log in from the MAC address specified as their user name
- **session limit reached ($(error-orig))** - depending on licence number of active HotSpot clients is limited to some number. The error is displayed when this limit is reached. Solution: try to log in later when there will be less concurrent user sessions, or buy an another license that allows more simultaneous sessions
- **hotspot service is shutting down** - RouterOS is currently being restarted or shut down. Solution: wait until the service will be available again

General fatal errors:

- **internal error ($(error-orig))** - this should never happen. If it will, error page will be shown displaying this error message (error-orig will describe what has happened). Solution: correct the error reported
- **configuration error ($(error-orig))** - the HotSpot server is not configured properly (error-orig will describe what has happened). Solution: correct the error reported
- **cannot assign ip address - no more free addresses from pool** - unable to get an IP address from an IP pool as there is no more free IP addresses in that pool. Solution: make sure there is a

sufficient amount of free IP addresses in IP pool

Local HotSpot user database non-fatal errors:

- **invalid username or password** - self-explanatory
- **user $(username) is not allowed to log in from this MAC address** - trying to log in from a MAC address different from specified in user database. Solution: log in from the correct MAC address or take out the limitation
- **user $(username) has reached uptime limit** - self-explanatory
- **user $(username) has reached traffic limit** - either limit-bytes-in or limit-bytes-out limit is reached
- **no more sessions are allowed for user $(username)** - the shared-users limit for the user's profile is reached. Solution: wait until someone with this username logs out, use different login name or extend the shared-users limit

RADIUS client non-fatal errors:

- **invalid username or password** - RADIUS server has rejected the username and password sent to it without specifying a reason. Cause: either wrong username and/or password, or other error. Solution: should be clarified in RADIUS server's log files
- **<error_message_sent_by_radius_server>** - this may be any message (any text string) sent back by RADIUS server. Consult with your RADIUS server's documentation for further information

RADIUS client fatal errors:

- **RADIUS server is not responding** - user is being authenticated by RADIUS server, but no response is received from it. Solution: check whether the RADIUS server is running and is reachable from the HotSpot router

Application Examples

Description

This section will focus on some simple examples of how to use your HotSpot system, as well as give some useful ideas.

Setting up HTTPS authorization

At first certificate must be present with decrypted private key:

```
[admin@MikroTik] > /certificate print
Flags: K - decrypted-private-key, Q - private-key, R - rsa, D - dsa
 0 KR name="hotspot.example.net"
      subject=C=LV,L=Riga,O=MT,OU=dev,CN=hotspot.example.net,
              emailAddress=admin@hotsot.example.net
      issuer=C=LV,L=Riga,O=MT,OU=dev,CN=hotsot.example.net,
             emailAddress=admin@hotsot.example.net
      serial-number="0" email=admin@hotsot.example.net
      invalid-before=oct/27/2004 11:43:22 invalid-after=oct/27/2005 11:43:22
      ca=yes
```

Then we can use that certificate for HotSpot:

```
/ip hotspot profile set default login-by=cookie,http-chap,https \
ssl-certificate=hotsot.example.net
```

After that we can see, that HTTPS is running on HotSpot interface:

```
[admin@MikroTik] > /ip hotspot print
Flags: X - disabled, I - invalid, S - HTTPS
 #   NAME                        INTERFACE    ADDRESS-POOL PROFILE IDLE-TIMEOUT
 0 S hs-local                    local                     default 00:05:00
```

Bypass HotSpot for some devices in HotSpot network

All IP binding entries with **type** property set to **bypassed**, will not be asked to authorize - it means that they will have login-free access:

```
[admin@MikroTik] ip hotspot ip-binding> print
Flags: X - disabled, P - bypassed, B - blocked
 #   MAC-ADDRESS       ADDRESS         TO-ADDRESS      SERVER
 0 P                   10.11.12.3
```

If all fields has been filled in the ip-binding table and **type** has been set to **bypassed**, then the IP address of this entry will be accessible from public interfaces immediately:

```
[admin@MikroTik] ip hotspot ip-binding> print
Flags: X - disabled, P - bypassed, B - blocked
 #   MAC-ADDRESS       ADDRESS         TO-ADDRESS      SERVER
 0 P                   10.11.12.3
 1 P 00:01:02:03:04:05 10.11.12.3      10.11.12.3      hs-local
[admin@MikroTik] ip hotspot ip-binding> .. host print
Flags: S - static, H - DHCP, D - dynamic, A - authorized, P - bypassed
 #    MAC-ADDRESS       ADDRESS         TO-ADDRESS      SERVER   IDLE-TIMEOUT
 0  P 00:01:02:03:04:05 10.11.12.3      10.11.12.3      hs-local
```

Web Proxy

Document revision 1.5 (December 12, 2007, 11:44 GMT)

This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS implements the following proxy server features:

- Regular HTTP proxy
- Transparent proxy. Can be transparent and regular at the same time
- Access list by source, destination, URL and requested method
- Cache access list (specifies which objects to cache, and which not)
- Direct Access List (specifies which resources should be accessed directly, and which - through another proxy server)
- Logging facility

Quick Setup Guide

To set up a 1 GiB large web cache, which will listen on port 8000, do the following:

```
[admin@MikroTik] ip proxy> set enabled=yes port=8000 max-cache-size=1048576
[admin@MikroTik] ip proxy> print
                 enabled: yes
             src-address: 0.0.0.0
                    port: 8000
            parent-proxy: 0.0.0.0
       parent-proxy-port: 0
             cache-drive: system
     cache-administrator: "webmaster"
          max-cache-size: 1048576KiB
           cache-on-disk: no
  max-client-connections: 600
  max-server-connections: 600
          max-fresh-time: 3d
   serialize-connections: no
       always-from-cache: no
          cache-hit-dscp: 4
[admin@MikroTik] ip proxy>
```

Remember to secure your proxy by preventing unauthorized access to it, otherwise it may be used as an open proxy. Also you need to setup destination NAT in order to utilize transparent proxying facility:

```
[admin@MikroTik] ip firewall nat> add chain=dstnat protocol=tcp dst-port=80
action=redirect to-ports=8000
[admin@MikroTik] ip firewall nat>
```

Specifications

Packages required: ***web-proxy***
License required: ***level3***
Home menu level: ***/ip web-proxy***
Standards and Technologies: ***HTTP/1.0***, ***HTTP/1.1***, ***FTP***
Hardware usage: *uses memory and disk space, if available (see description below)*

Description

Web proxy performs Internet object cache function by storing requested Internet objects, i.e., data available via HTTP and FTP protocols on a system positioned closer to the recipient than the site the data is originated from. Here 'closer' means increased path reliability, speed or both. Web browsers can then use the local proxy cache to speed up access and external reduce bandwidth consumption.

When setting up Web proxy, make sure it serves only your clients, and is not misused as relay. Please read the security notice in the Access List Section!

Note that it may be useful to have Web proxy running even with no cache when you want to use it as something like HTTP and FTP firewall (for example, denying access to mp3 files) or to redirect requests to external proxy with large cache drives transparently.

Setup

Home menu level: ***/ip proxy***

Property Description

always-from-cache (yes | no; default: **no**) - ignore client refresh requests if the content is considered fresh

cache-administrator (*text*; default: **webmaster**) - administrator's e-mail displayed on proxy error page

cache-drive (*systemname*; default: **system**) - specifies the target disk drive to be used for storing cached objects. You can use console completion to see the list of available drives

cache-hit-dscp (*integer*: 0..63) - automatically mark cache hit with the provided DSCP value

cache-on-disk (yes | no; default: **no**) - whether to store cache files on disk or in RAM filesystem

enabled (yes | no; default: **no**) - specifies whether the web proxy is enabled

max-cache-size (*none* | *unlimitedinteger*: 0..4294967295; default: **none**) - specifies the maximal disk cache size, measured in kibibytes

max-client-connections (*integer*; default: **600**) - maximum number of concurrent client connections accepted by the proxy. All further connections will be rejected

max-fresh-time (*time*; default: **3d**) - an upper limit on how long objects without an explicit expiry time will be considered fresh

max-server-connections (*integer*; default: **600**) - maximum number of concurrent proxy connections to external servers. All further connections will be put on hold until some of the existing server connections will terminate

parent-proxy (*IP addressport*; default: **0.0.0.0**) - IP address of the upper-level (parent) proxy

parent-proxy-port (*port*) - TCP port the parent proxy is active on

port (*port*; default: **3128**) - specifies the port(s) the web proxy will be listening on

serialize-connections (yes | no; default: **no**) - Do not make multiple connections to server for multiple client connections, if possible (i.e. server supports persistent HTTP connections). Clients will be served on FIFO principle; next client is processed when response transfer to the previous one is completed. If a client is idle for too long (max 5 seconds by default), it will give up waiting and open another connection to the server

src-address (*IP address*; default: **0.0.0.0**) - the web-proxy will use this address connecting to the parent proxy or web site.

- **0.0.0.0** - appropriate src-address will be automatically taken from the routing table (preferred source of the respective route)

Notes

The web proxy listens to all IP addresses that the router has in its IP address list.

Example

To enable proxy on port 8080 with maximal available cache size:

```
[admin@MikroTik] ip proxy> set enabled=yes port=8080 \
\...  max-cache-size=unlimited
[admin@MikroTik] ip proxy> print
                 enabled: yes
             src-address: 0.0.0.0
                    port: 8000
            parent-proxy: 0.0.0.0
       parent-proxy-port: 0
             cache-drive: system
     cache-administrator: "webmaster"
          max-cache-size: 21000KiB
           cache-on-disk: no
  max-client-connections: 600
  max-server-connections: 600
          max-fresh-time: 3d
   serialize-connections: no
       always-from-cache: no
          cache-hit-dscp: 4
[admin@MikroTik] ip proxy>
```

Note how the **max-cache-size** value has been calculated from the **unlimited** to an accurate value in kibibytes

Proxy Monitoring

Command name: ***/ip proxy monitor***

Property Description

cache-used (*read-only: integer*) - the amount of disk (or RAM if the cache is stored only in RAM) used by the cache

free-disk-space (*read-only: integer*) - the amount of free space on the cache drive

hits (*read-only: integer*) - number of client requests resolved from the cache

hits-sent-to-clients (*read-only: integer*) - the amount of cache hits sent to client

received-from-servers (*read-only: integer*) - total amount of data received from the external servers

requests (*read-only: integer*) - total number of client requests to the proxy

sent-to-clients (*read-only: integer*) - total amount of data sent to the clients

status (*read-only: text*; default: **stopped**) - display status information of the proxy server

- **stopped** - proxy is disabled and is not running
- **running** - proxy is enabled and running
- **formatting-disk** - the cache drive is being formatted
- **checking-disk** - the cache drive is being checked for errors and cache inconsistencies
- **invalid-address** - proxy is enabled, but not running because of invalid address (you should change address or port)

total-disk-size (*read-only: integer*) - size of the cache drive

total-ram-used (*read-only: integer*) - the amount of memory used by the proxy (excluding RAM cache size)

uptime (*read-only: time*) - the time since the proxy has been started last time

Access List

Home menu level: ***/ip proxy access***

Description

Access list is configured in the same way as MikroTik RouterOS firewall rules. Rules are processed from the top to the bottom. First matching rule specifies decision of what to do with this connection. There is a total of 6 classifiers that specify matching constraints. If none of these classifiers is specified, the particular rule will match any connection.

If connection is matched by a rule, **action** property of this rule specifies whether connection will be allowed or not. If some connection does not match any rule, it will be allowed.

Property Description

action (*allow* | *deny*; default: **allow**) - specifies whether to pass or deny matched packets

dst-address (*IP addressnetmask*) - destination address of the IP packet

dst-host (*wildcard*) - IP address or DNS name used to make connection the target server (this is the string user wrote in his/her browser before specifying port and path to a particular web page)

dst-port (*port*) - a list or range of ports the packet is destined to

hits (*read-only: integer*) - the number of requests that were policed by this rule

local-port (*port*) - specifies the port of the web proxy via which the packet was received. This value should match one of the ports web proxy is listening on.

method (*any* | *connect* | *delete* | *get* | *head* | *options* | *post* | *put* | *trace*) - HTTP method used in the request (see HTTP Methods section at the end of this document)

path (*wildcard*) - name of the requested page within the target server (i.e. the name of a particular web page or document without the name of the server it resides on)

redirect-to (*text*) - in case access is denied by this rule, the user shall be redirected to the URL specified here

src-address (*IP addressnetmask*) - source address of the IP packet

Notes

It is strongly recommended to deny all IP addresses except those behind the router as the proxy still may be used to access your internal-use-only (intranet) web servers. Also, consult examples in Firewall Manual on how to protect your router.

Wildcard property **url** matches a complete string (i.e., they will not match "example.com" if they are set to "example"). Available wildcards are '*' (match any number of any characters) and '?' (match any one character). Regular expressions are also accepted here, but if the property should be treated as a regular expression, it should start with a colon (':').

Small hits in using regular expressions:

- **** symbol sequence is used to enter **** character in console
- **\.** pattern means **.** only (in regular expressions single dot in pattern means any symbol)
- to show that no symbols are allowed before the given pattern, we use **^** symbol at the beginning of the pattern
- to specify that no symbols are allowed after the given pattern, we use **$** symbol at the end of the pattern
- to enter **[** or **]** symbols, you should escape them with backslash **\.**

Direct Access List

Home menu level: ***/ip proxy direct***

Description

If **parent-proxy** property is specified, it is possible to tell the proxy server whether to try to pass the request to the parent proxy or to resolve it connecting to the requested server directly. Direct Access List is managed just like Proxy Access List described in the previous chapter except the **action** argument.

Property Description

action (*allow* | *deny*; default: **allow**) - specifies the action to perform on matched packets

- **allow** - always resolve matched requests directly bypassing the parent router
- **deny** - resolve matched requests through the parent proxy. If no one is specified this has the same effect as allow

dst-address (*IP addressnetmask*) - destination address of the IP packet

dst-host (*wildcard*) - IP address or DNS name used to make connection the target server (this is the string user wrote in his/her browser before specifying port and path to a particular web page)

dst-port (*port*) - a list or range of ports the packet is destined to

local-port (*port*) - specifies the port of the web proxy via which the packet was received. This value should match one of the ports web proxy is listening on.

method (*any* | *connect* | *delete* | *get* | *head* | *options* | *post* | *put* | *trace*) - HTTP method used in the request (see HTTP Methods section in the end of this document)

path (*wildcard*) - name of the requested page within the target server (i.e. the name of a particular web page or document without the name of the server it resides on)

src-address (*IP addressnetmask*) - source address of the IP packet

Notes

Unlike the access list, the direct proxy access list has default action equal to **deny**. It takes place when no rules are specified or a particular request did not match any rule.

Cache Management

Home menu level: ***/ip proxy cache***

Description

Cache access list specifies, which requests (domains, servers, pages) have to be cached locally by web proxy, and which not. This list is implemented exactly the same way as web proxy access list. Default action is to cache object (if no matching rule is found).

Property Description

action (*allow* | *deny*; default: **allow**) - specifies the action to perform on matched packets

- **allow** - cache objects from matched request
- **deny** - do not cache objects from matched request

dst-address (*IP addressnetmask*) - destination address of the IP packet

dst-port (*port*) - a list or range of ports the packet is destined to

local-port (*port*) - specifies the port of the web proxy via which the packet was received. This value should match one of the ports web proxy is listening on.

method (*any* | *connect* | *delete* | *get* | *head* | *options* | *post* | *put* | *trace*) - HTTP method used in the request (see HTTP Methods section in the end of this document)

path (*wildcard*) - name of the requested page within the target server (i.e. the name of a particular web page or document without the name of the server it resides on)

path (*wildcard*) - name of the requested page within the target server (i.e. the name of a particular web page or document without the name of the server it resides on)

src-address (*IP addressnetmask*) - source address of the IP packet

Connection List

Home menu level: ***/ip proxy connections***

Description

This menu contains the list of current connections the proxy is serving

Property Description

dst-address (*read-only: IP address*) - IP address of to which data are passed via this proxy

protocol (*read-only: text*) - protocol name

rx-bytes (*read-only: integer*) - the amount of bytes received from the remote end

src-address (*read-only: IP address*) - IP address of the remote end of the connection

state (*read-only: connecting | idle | resolving | rx-body | rx-header | tx-body | tx-header*) - opened connection state

- **connecting** - establishing connection with server
- **idle** - waiting for next client to serve
- **resolving** - resolving server's DNS name
- **rx-body** - receiving HTTP body
- **rx-header** - receiving HTTP header; or waiting for next request from client
- **tx-body** - transmitting HTTP body
- **tx-header** - transmitting HTTP header

tx-bytes (*read-only: integer*) - the amount of bytes sent to the remote end

Cache Contents

Home menu level: ***/ip proxy cache-contents***

Description

This menu lists all the files stored in the cache

Property Description

file-size (*read-only: integer*) - size of the stored file

last-accessed (*read-only: date*) - date of the last access to the resource

last-accessed-time (*read-only: time*) - time of the last access to the resource

last-modified (*read-only: date*) - modification date

last-modified-time (*read-only: time*) - modification time

uri (*read-only: text*) - full resource name

Cache inserts

Home menu level: ***/ip proxy inserts***

Description

This menu shows statistics on objects stored in cache (cache inserts)

Property Description

denied (*read-only: integer*) - number of inserts denied by the caching list

errors (*read-only: integer*) - number of disk or other system-related errors

no-memory (*read-only: integer*) - number of objects not stored because there was not enough memory

successes (*read-only: integer*) - number of successfull cache inserts

too-large (*read-only: integer*) - number of objects too large to store

Cache Lookups

Home menu level: ***/ip proxy lookups***

Description

This menu shows statistics on objects read from cache (cache lookups)

Property Description

denied (*read-only: integer*) - number of requests denied by the access list

expired (*read-only: integer*) - number of requests found in cache, but expired, and, thus, requested from an external server

no-expiration-info (*read-only: integer*) - conditional request received for a page that does not have the information to compare the request with

non-cacheable (*read-only: integer*) - number of requests requested from the external servers unconditionally (as their caching is denied by the cache access list)

not-found (*read-only: integer*) - number of requests not found in the cache, and, thus, requested from an external server (or parent proxy if configured accordingly)

successes (*read-only: integer*) - number of requests found in the cache

Complementary Tools

Description

Web proxy has additional commands to handle non-system drive used for caching purposes and to recover the proxy from severe file system errors.

Command Description

check-drive - checks non-system cache drive for errors

clear-cache - deletes existing cache and creates new cache directories

format-drive - formats non-system cache drive and prepairs it for holding the cache

Transparent Mode

Description

Transparent proxy feature performs request caching invisibly to the end-user. This way the user does not notice that his connection is being processed by the proxy and therefore does not need to perform any additional configuration of the software he is using.

This feature may as well be combined with bridge to simplify deployment of web proxy in the existing infrastructure.

To enable the transparent mode, place a firewall rule in destination NAT, specifying which connections, *id est* traffic coming to which ports should be redirected to the proxy.

Notes

Only HTTP traffic is supported in transparent mode of the web proxy. HTTPS and FTP protocols are not going to work this way.

Example

To configure the router to transparently redirect all connections coming from **ether1** interface to port **80** to the web proxy listening on port **8080**, then add the following destination NAT rule:

```
[admin@MikroTik] > /ip firewall nat add in-interface=ether1 dst-port=80 \
\... protocol=tcp action=redirect to-ports=8080 chain=dstnat
[admin@MikroTik] > /ip firewall nat print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=dstnat protocol=tcp in-interface=ether1 dst-port=80 action=redirect
     to-ports=8080
[admin@MikroTik] >
```

Be aware, that you will not be able to access the router's web page after addition of the rule above unless you will change the port for the **www** service under **/ip service** submenu to a different value or explicitly exclude router's IP address from those to be matched, like:

```
 /ip firewall nat add in-interface=ether1 dst-port=80 \
\... protocol=tcp action=redirect to-ports=8080 chain=dstnat dst-address=!1.1.1.1/32
```

It is assumed that the router's address is **1.1.1.1/32**.

HTTP Methods

Description

OPTIONS

This method is a request of information about the communication options available on the chain between the client and the server identified by the **Request-URI**. The method allows the client to determine the options and (or) the requirements associated with a resource without initiating any resource retrieval

GET

This method retrieves whatever information identified by the **Request-URI**. If the **Request-URI** refers to a data processing process than the response to the **GET** method should contain data produced by the process, not the source code of the process procedure(-s), unless the source is the result of the process.

The **GET** method can become a *conditional* **GET** if the request message includes an **If-Modified-Since, If-Unmodified-Since, If-Match, If-None-Match**, or **If-Range** header field. The conditional **GET** method is used to reduce the network traffic specifying that the transfer of the entity should occur only under circumstances described by conditional header field(-s).

The **GET** method can become a *partial* **GET** if the request message includes a **Range** header field. The partial **GET** method intends to reduce unnecessary network usage by requesting only parts of entities without transferring data already held by client.

The response to a **GET** request is cacheable if and only if it meets the requirements for HTTP caching.

HEAD

This method shares all features of **GET** method except that the server must not return a message-body in the response. This retrieves the metainformation of the entity implied by the request which leads to a wide usage of it for testing hypertext links for validity, accessibility, and recent modification.

The response to a **HEAD** request may be cacheable in the way that the information contained in the response may be used to update previously cached entity identified by that **Request-URI**.

POST

This method requests that the origin server accept the entity enclosed in the request as a new subordinate of the resource identified by the **Request-URI**.

The actual action performed by the **POST** method is determined by the origin server and usually is **Request-URI** dependent.

Responses to **POST** method are not cacheable, unless the response includes appropriate **Cache-Control** or **Expires** header fields.

PUT

This method requests that the enclosed entity be stored under the supplied **Request-URI**. If another entity exists under specified **Request-URI**, the enclosed entity should be considered as updated (newer) version of that residing on the origin server. If the **Request-URI** is not pointing to an existing resource, the origin server should create a resource with that URI.

If the request passes through a cache and the **Request-URI** identifies one or more currently cached entities, those entries should be treated as stale. Responses to this method are not cacheable.

TRACE

This method invokes a remote, application-layer loop-back of the request message. The final recipient of the request should reflect the message received back to the client as the entity-body of a 200 (OK) response. The final recipient is either the origin server or the first proxy or gateway to receive a **Max-Forwards** value of **0** in the request. A **TRACE** request must not include an entity.

Responses to this method MUST NOT be cached.

IP Pools

Document revision 0.1 (January 14, 2008, 9:50 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

IP pools are used to define range of IP addresses that is used for DHCP server and Point-to-Point servers

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip pool***
Standards and Technologies: ***none***
Hardware usage: ***Not significant***

Description

IP pools simply group IP addresses for further usage. It is a single configuration point for all features that assign IP addresses to clients.

Notes

Whenever possible, the same ip address is given out to each client (OWNER/INFO pair).

Setup

Home menu level: ***/ip pool***

Property Description

name (*name*) - the name of the pool

next-pool (*name*) - when address is acquired from pool that has no free addresses, and next-pool property is set to another pool, then next IP address will be acquired from next-pool

ranges (*IP address*) - IP address list of non-overlapping IP address ranges in form of: from1-to1,from2-to2,...,fromN-toN. For example, 10.0.0.1-10.0.0.27,10.0.0.32-10.0.0.47

Example

To define a pool named **ip-pool** with the **10.0.0.1-10.0.0.125** address range excluding gateway's address **10.0.0.1** and server's address **10.0.0.100**, and the other pool **dhcp-pool**, with the **10.0.0.200-10.0.0.250** address range:

```
[admin@MikroTik] ip pool> add name=ip-pool ranges=10.0.0.2-10.0.0.99,10.0.0.101
10.0.0.126
[admin@MikroTik] ip pool> add name=dhcp-pool ranges=10.0.0.200-10.0.0.250
[admin@MikroTik] ip pool> print
  # NAME                                        RANGES
  0 ip-pool                                     10.0.0.2-10.0.0.99
                                                10.0.0.101-10.0.0.126
  1 dhcp-pool                                   10.0.0.200-10.0.0.250

[admin@MikroTik] ip pool>
```

Used Addresses from Pool

Home menu level: ***/ip pool used***

Description

Here you can see all used IP addresses from IP pools.

Property Description

address (*read-only: IP address*) - IP address that is assigned to client form the pool

info (*read-only: name*) - name of the interface to which the client is connected to

owner (*read-only: MAC address*) - MAC address of the client

pool (*read-only: name*) - name of the IP pool

Example

See used addresses from pool:

```
[admin@MikroTik] ip pool used> print
POOL  ADDRESS         OWNER                        INFO
local 192.168.0.100   00:0C:42:03:1F:60            test
local 192.168.0.99    00:0C:42:03:21:0F            test
```

SOCKS Proxy Server

Document revision 1.4 (January 14, 2008, 11:23 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

This manual discusses the SOCKS proxy server which is implemented in RouterOS. MikroTik RouterOS supports SOCKS version 4.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip socks***
Standards and Technologies: ***SOCKS version 4***
Hardware usage: ***Not significant***

Description

SOCKS is a proxy server that allows TCP based application data to relay across the firewall, even if the firewall would block the packets. The SOCKS protocol is independent from application protocols, so it can be used for many services, e.g, WWW, FTP, TELNET, and others.

At first, an application client connects to the SOCKS proxy server, then the proxy server looks in its **access** list to see whether the client is permited to access the remote application resource or not, if it is permitted, the proxy server relies the packet to the application server and creates a connection between the application server and client.

Notes

Remember to configure your application client to use SOCKS version 4.

You should secure the SOCKS proxy using its access list and/or firewall to disallow access from outisde. Failing to secure the proxy server may introduce security issues to your network, and may provide a way for spammers to send junk mail through the router.

Additional Documents

- Information about SOCKS

SOCKS Configuration

Description

In this section you will learn how to enable the SOCKS proxy server and do its configuration.

Property Description

connection-idle-timeout (*time*; default: **2m**) - time after which idle connections are terminated

enabled (yes | no; default: **no**) - whether to enable or no the SOCKS proxy

max-connections (*integer*: 1..500; default: **200**) - maxumum number of simultaneous connections

port (*integer*: 1..65535; default: **1080**) - TCP port on which the SOCKS server listens for connections

Example

To enable SOCKS:

```
[admin@MikroTik] ip socks> set enabled=yes
[admin@MikroTik] ip socks> print
                    enabled: yes
                       port: 1080
    connection-idle-timeout: 2m
            max-connections: 200
[admin@MikroTik] ip socks>
```

Access List

Home menu level: ***/ip socks access***

Description

In the SOCKS access list you can add rules which will control access to SOCKS server. This list is similar to firewall lists.

Property Description

action (*allow* | *deny*; default: **allow**) - action to be performed for this rule

- **allow** - allow packets, matching this rule, to be forwarded for further processing
- **deny** - deny access for packets, matching this rule

dst-address (*IP addressnetmask*) - destination (server's) address

dst-port (*port*) - destination TCP port

src-address (*IP addressnetmask*) - source (client's) address for a packet

src-port (*port*) - source TCP port

Active Connections

Home menu level: ***/ip socks connections***

Description

The Active Connection list shows all established TCP connections, which are maintained through the SOCKS proxy server.

Property Description

dst-address (*read-only: IP address*) - destination (application server) IP address

rx (*read-only: integer*) - bytes received

src-address (*read-only: IP address*) - source (application client) IP address

tx (*read-only: integer*) - bytes sent

type (*read-only: in* | *out* | *unknown*) - connection type

- **in** - incoming connection
- **out** - outgoing connection
- **unknown** - connection has just been initiated

Example

To see current TCP connections:

```
[admin@MikroTik] ip socks connections> print
 # SRC-ADDRESS                DST-ADDRESS                TX         RX
 0 192.168.0.2:3242           159.148.147.196:80         4847       2880
 1 192.168.0.2:3243           159.148.147.196:80         3408       2127
 2 192.168.0.2:3246           159.148.95.16:80           10172      25207
 3 192.168.0.2:3248           194.8.18.26:80             474        1629
 4 192.168.0.2:3249           159.148.95.16:80           6477       18695
 5 192.168.0.2:3250           159.148.95.16:80           4137       27568
 6 192.168.0.2:3251           159.148.95.16:80           1712       14296
 7 192.168.0.2:3258           80.91.34.241:80            314        208
 8 192.168.0.2:3259           80.91.34.241:80            934        524
 9 192.168.0.2:3260           80.91.34.241:80            930        524
10 192.168.0.2:3261           80.91.34.241:80            312        158
11 192.168.0.2:3262           80.91.34.241:80            312        158
```

```
[admin@MikroTik] ip socks connections>
```

Application Examples

FTP service through SOCKS server

Let us consider that we have a network **192.168.0.0/24** which is masqueraded, using a router with a public IP **10.1.0.104/24** and a private IP **192.168.0.1/24**. Somewhere in the network is an FTP server with IP address **10.5.8.8**. We want to allow access to this FTP server for a client in our local network with IP address **192.168.0.2/24**.

We have already masqueraded our local network:

```
[admin@MikroTik] ip firewall nat> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=srcnat action=masquerade src-address=192.168.0.0/24
[admin@MikroTik] ip firewall nat>
```

And the access to public FTP servers is denied in firewall:

```
[admin@MikroTik] ip firewall filter> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=forward action=drop src-address=192.168.0.0/24 dst-port=21 protocol=tcp
[admin@MikroTik] ip firewall filter>
```

We need to enable the SOCKS server:

```
[admin@MikroTik] ip socks> set enabled=yes
[admin@MikroTik] ip socks> print
                    enabled: yes
                       port: 1080
    connection-idle-timeout: 2m
            max-connections: 200
[admin@MikroTik] ip socks>
```

Add access to a client with an IP address **192.168.0.2/32** to SOCKS access list, allow data transfer from FTP server to client (allow destionation ports from 1024 to 65535 for any IP address), and drop everything else:

```
[admin@MikroTik] ip socks access> add src-address=192.168.0.2 dst-port=21 \
\... action=allow
[admin@MikroTik] ip socks access> add dst-port=1024-65535 action=allow
[admin@MikroTik] ip socks access> add action=deny
[admin@MikroTik] ip socks access> print
Flags: X - disabled
 0   src-address=192.168.0.2 dst-port=21 action=allow
 1   dst-port=1024-65535 action=allow
 2   action=deny
[admin@MikroTik] ip socks access>
```

That's all - the SOCKS server is configured. To see active connections and data transmitted and received:

```
[admin@MikroTik] ip socks connections> print
 # SRC-ADDRESS                DST-ADDRESS                TX         RX
 0 192.168.0.2:1238           10.5.8.8:21                1163       4625
 1 192.168.0.2:1258           10.5.8.8:3423              0          3231744
[admin@MikroTik] ip socks connections>
```

Note! In order to use SOCKS proxy server, you have to specify its IP address and port in your FTP client. In this case IP address would be **192.168.0.1** (local IP address of the router/SOCKS server) and TCP port **1080**.

UPnP

Document revision 2.3 (January 14, 2008, 11:56 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik RouterOS supports Universal Plug and Play architecture for transparent peer-to-peer network connectivity of personal computers and network-enabled intelligent devices or appliances. UPnP builds enables these devices to automatically connect with one another and work together to make networking possible for more people.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip upnp***
Standards and Technologies: ***TCP/IP***, ***HTTP***, ***XML***, ***IGD***
Hardware usage: ***Not significant***

Description

UPnP enables data communication between any two devices under the command of any control device on the network. Universal Plug and Play is completely independent of any particular physical medium. It supports networking with automatic discovery without any initial configuration, whereby a device can dynamically join a network. DHCP and DNS servers are optional and will be used if available on the network. UPnP implements simple yet powerfull NAT traversal solution, that enables the client to get full two-way peer-to-peer network support from behind the NAT.

There are two interface types for UPnP: internal (the one local clients are connected to) and external (the one the Internet is connected to). A router may only have one external interface with a 'public' IP address on it, and as many internal interfaces as needed, all with source-NATted 'internal' IP addresses.

The UPnP protocol is used for many modern applications, like most of DirectX games, as well as for various Windows Messenger features (remote asisstance, application sharing, file transfer, voice, video) from behind a firewall.

Additional Documents

Enabling Universal Plug-n-Play

Home menu level: ***/ip upnp***

Property Description

allow-disable-external-interface (yes | no; default: **yes**) - whether or not should the users be allowed to disable router's external interface. This functionality (for users to be able to turn the router's external interface off without any authentication procedure) is required by the standard, but as it is sometimes not expected or unwanted in UPnP deployments which the standard was not designed for (it was designed mostly for home users to establish their ownlocal networks), you can disable this behavior

enabled (yes | no; default: **no**) - whether UPnP feature is enabled

show-dummy-rule (yes | no; default: **yes**) - this is to enable a workaround for some broken implementations, which are handling the absense of UPnP rules incorrectly (for example, popping up error messages). This option will instruct the server to install a dummy (meaningless) UPnP rule that can be observed by the clients, which refuse to work correctly otherwise

Notes

CAUTION: if you do not disable the **allow-disable-external-interface**, any user from the local network will be able (without any authentication procedures) to disable the router's external interface.

Example

To enable UPnP feature:

```
[admin@MikroTik] ip upnp> set enable=yes
[admin@MikroTik] ip upnp> print
                             enabled: yes
    allow-disable-external-interface: yes
                     show-dummy-rule: yes
[admin@MikroTik] ip upnp>
```

UPnP Interfaces

Home menu level: ***/ip upnp interfaces***

Property Description

interface (*name*) - interface name UPnP will be run on

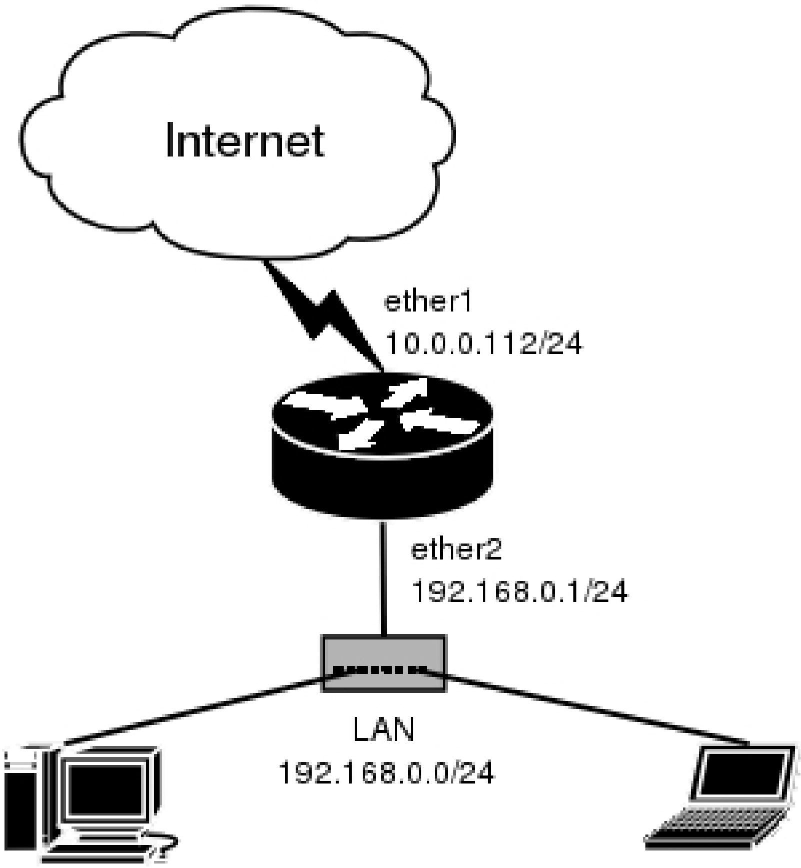
type (*external* | *internal*) - interface type, one of the:

- **external** - the interface a global IP address is assigned to
- **internal** - router's local interface the clients are connected to

Notes

It is highly recommended to upgrade DirectX runtime libraries to version DirectX 9.0c or higher and Windows Messenger to version Windows Messenger 5.0 or higher in order to get UPnP to work properly.

Example



We have masquerading already enabled on our router:

```
[admin@MikroTik] ip upnp interfaces> /ip firewall src-nat print
Flags: X - disabled, I - invalid, D - dynamic
  0   chain=srcnat action=masquerade out-interface=ether1
[admin@MikroTik] ip upnp interfaces>
```

Now all we have to do is to add interfaces and enable UPnP:

```
[admin@MikroTik] ip upnp interfaces> add interface=ether1 type=external
[admin@MikroTik] ip upnp interfaces> add interface=ether2 type=internal
```

```
[admin@MikroTik] ip upnp interfaces> print
Flags: X - disabled
  #   INTERFACE TYPE
  0 X ether1    external
  1 X ether2    internal

[admin@MikroTik] ip upnp interfaces> enable 0,1
[admin@MikroTik] ip upnp interfaces> .. set enabled=yes
[admin@MikroTik] ip upnp interfaces>
```

Certificate Management

Document revision 2.4 (January 23, 2008, 14:31 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

SSL (Secure Socket Layer) is a security technology to ensure encrypted transactions over a public network. To protect the data, an encryption key should be negotiated. SSL protocol is using Certificates to negotiate a key for data encryption.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/certificate***
Standards and Technologies: ***SSLv2***, ***SSLv3***, ***TLS***
Hardware usage: *high CPU usage*

Description

SSL technology was first introduced by Netscape to ensure secure transactions between browsers and web servers. When a browser requests a secure web page (usually on TCP port 443), a web server first sends a Certificate, which contains a public key for the encryption key negotiation to take place. After the encryption key is negotiated, the web server will send the requested page encrypted using this key to the browser (and also the browser will be able to submit its data securely to the server)

SSL Certificate confirms the web server identity. The Certificate contains information about its holder (like DNS name and Country), issuer (the entity has signed the Certificate) and also the public key used to negotiate the encryption key. In order a Certificate to play its role, it should be signed by a third party (Certificate Authority) which both parties trust. Modern browsers that support SSL protocol have a list of the Certificate Authorities they trust (the most known and trusted CA is VeriSign, but that is not the only one)

To use a Certificate (which contain a public key), server needs a private key. One of the keys is used for encryption, and the other - for decryption. It is important to understand, that both keys can encrypt and decrypt, but what is encrypted by one of them can be decrypted **only** by the another. Private key must be kept securely, so that nobody else can get it and use this certificate. Usually private key is encrypted with a passphrase.

Most trusted Certificate Authorities sell the service of signing Certificates (Certificates also have a finite validity term, so you will have to pay regularly). It is also possible to create a self-signed Certificate (you can create one on most UNIX/Linux boxes using openssl toolkit; all Root Certificate Authorities have self-signed Certificates), but if it is not present in a browser's database, the browser will pop up a security warning, saying that the Certificate is not trusted (note also that most browsers support importing custom Certificates to their databases).

Certificates

Home menu level: ***/certificate***

Description

MikroTik RouterOS can import Certificates for the SSL services it provides (only HotSpot for now). This submenu is used to manage Certificates for this services.

Property Description

alias (*read-only: text*) - alias (comment) used for generating the certificate

ca (yes | no; default: **yes**) - whether the certificate is used for building or verifying certificate chains (as Certificate Authority)

email (*read-only: text*) - e-mail address of the holder

invalid-after (*read-only: date*) - date the certificate is valid until

invalid-before (*read-only: date*) - date the certificate is valid from

issuer (*read-only: text*) - issuer of the certificate

name (*name*) - reference name

serial-number (*read-only: text*) - serial number of the certificate

subject (*read-only: text*) - holder (subject) of the certificate

Command Description

create-certificate-request - creates an RSA certificate request to be signed by a Certificate Authority. After this, download both private key and certificate request files from the router. When you receive your signed certificate from the CA, upload it and the private key (that is made by this command) to a router and use /certificate import command to install it

- **certificate request file name** - name for the certificate request file (if it already exists, it will be overwritten). This is the original certificate that will be signed by the Certificate Authority
- **file name** - name of private key file. If such file does not exist, it will be created during the next step. Private key is used to encrypt the certificate
- **passphrase** - the passphrase that will be used to encrypt generated private key file. You must

enter it twice to be sure you have not made any typing errors

- **rsa key bits** - number of bits for RSA (encryption) key. Longer keys take more time to generate. 4096 bit key takes about 30 seconds on Celeron 800 system to generate
- **country name** - (C) ISO two-character country code (e.g., LV for Latvia)
- **state or province name** - (ST) full name of state or province
- **locality name** - (L) locality (e.g. city) name
- **organization name** - (O) name of the organization or company
- **organization unit name** - (OU) organization unit name
- **common name** - (CN) the server's common name. For SSL web servers this must be the fully qualified domain name (FQDN) of the server that will use this certificate (like www.example.com). This is checked by web browsers
- **email address** - (Email) e-mail address of the person responsible for the certificate
- **challenge password** - the challenge password. It's use depends on your CA. It may be used to revoke this certificate
- **unstructured address** - unstructured address (like street address). Enter only if your CA accepts or requires it

decrypt - decrypt and cache public keys

- **passphrase** - passphrase for the found encrypted private key
- **keys-decrypted** - how many keys were successfully decrypted and cached

import - install new certificates

- **file-name** - import only this file (all files are searched for certificates by default)
- **passphrase** - passphrase for the found encrypted private key
- **certificates-imported** - how many new certificates were successfully imported
- **private-keys-imported** - how many private keys for existing certificates were successfully imported
- **files-imported** - how many files contained at least one item that was successfully imported
- **decryption-failures** - how many files could not be decrypted
- **keys-with-no-certificate** - how many public keys were successfully decrypted, but did not have matching certificate already installed

reset-certificate-cache - delete all cached decrypted public keys and rebuild the certificate cache

Notes

Server certificates may have **ca** property set to **no**, but Certificate Authority certificates must have it set to **yes**

Certificates and encrypted private keys are imported from and exported to the router's FTP server. Public keys are not stored on a router in unencrypted form. Cached decrypted private keys are stored in encrypted form, using key that is derived from the router ID. Passphrases are not stored on router.

Configuration backup does not include cached decrypted private keys. After restoring backup all certificates with private keys must be decrypted again, using **decrypt** command with the correct passphrase.

No other certificate operations are possible while generating a key.

When making a certificate request, you may leave some of the fields empty. CA may reject your certificate request if some of these values are incorrect or missing, so please check what are the requirements of your CA

Example

To import a certificate and the respective private key already uploaded on the router:

```
[admin@MikroTik] certificate> import
passphrase: xxxx
       certificates-imported: 1
       private-keys-imported: 1
              files-imported: 2
         decryption-failures: 0
    keys-with-no-certificate: 1
[admin@MikroTik] certificate> print
Flags: K - decrypted-private-key, Q - private-key, R - rsa, D - dsa
  0 QR name="cert1" subject=C=LV,ST=.,O=.,CN=cert.example.com
       issuer=C=LV,ST=.,O=.,CN=third serial-number="01"
       invalid-before=sep/17/2003 11:56:19 invalid-after=sep/16/2004 11:56:19
       ca=yes

[admin@MikroTik] certificate> decrypt
passphrase: xxxx
    keys-decrypted: 1
[admin@MikroTik] certificate> print
Flags: K - decrypted-private-key, Q - private-key, R - rsa, D - dsa
  0 KR name="cert1" subject=C=LV,ST=.,O=.,CN=cert.example.com
       issuer=C=LV,ST=.,O=.,CN=third serial-number="01"
       invalid-before=sep/17/2003 11:56:19 invalid-after=sep/16/2004 11:56:19
       ca=yes

[admin@MikroTik] certificate>
```

Now the certificate may be used by HotSpot servlet:

```
[admin@MikroTik] ip service> print
Flags: X - disabled, I - invalid
  #   NAME                                 PORT  ADDRESS             CERTIFICATE
  0   telnet                               23    0.0.0.0/0
  1   ftp                                  21    0.0.0.0/0
  2   www                                  8081  0.0.0.0/0
  3   hotspot                              80    0.0.0.0/0
  4   ssh                                  22    0.0.0.0/0
  5   hotspot-ssl                          443   0.0.0.0/0           none

[admin@MikroTik] ip service> set hotspot-ssl certificate=
cert1  none
[admin@MikroTik] ip service> set hotspot-ssl certificate=cert1
[admin@MikroTik] ip service> print
Flags: X - disabled, I - invalid
  #   NAME                                 PORT  ADDRESS             CERTIFICATE
  0   telnet                               23    0.0.0.0/0
  1   ftp                                  21    0.0.0.0/0
  2   www                                  8081  0.0.0.0/0
  3   hotspot                              80    0.0.0.0/0
  4   ssh                                  22    0.0.0.0/0
  5   hotspot-ssl                          443   0.0.0.0/0           cert1

[admin@MikroTik] ip service>
```

DDNS Update Tool

Document revision 1.3 (January 23, 2008, 14:31 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Dynamic DNS Update Tool gives a way to keep domain name pointing to dynamic IP address. It works by sending domain name system update request to name server, which has a zone to be updated. Secure DNS updates are also supported.

The DNS update tool supports only one algorithm - **hmac-md5**. It's the only proposed algorithm for signing DNS messages.

Specifications

Packages required: ***advanced-tools***
License required: ***level1***
Command name: ***/tool dns-update***
Standards and Technologies: ***Dynamic Updates in the DNS (RFC 2136)***, ***Secure DNS Dynamic Update (RFC 3007)***
Hardware usage: ***Not significant***

Description

Dynamic DNS Update is a tool that should be manually run to update dynamic DNS server.

Note that you have to have a DNS server that supports DNS updates properly configured.

Additional Documents

- DNS related RFCs

Dynamic DNS Update

Command name: ***/tool dns-update***

Property Description

address (*IP address*) - defines IP address associated with the domain name

dns-server (*IP address*) - DNS server to send update to

key (*text*; default: **""**) - authorization key (password of a kind) to access the server

key-name (*text*; default: **""**) - authorization key name (username of a kind) to access the server

name (*text*) - name to attach with the IP address

ttl (*integer*; default: **0**) - time to live for the item (in seconds)

zone (*text*) - DNS zone where to update the domain name in

Notes

Note that the system clock time on your router can't differ from the DNS server's time more than 5 minutes. Otherwise the DNS server will ignore this request.

Example

To tell **23.34.45.56** DNS server to (re)associate **mydomain** name in the **myzone.com** zone with **68.42.14.4** IP address specifying that the name of the key is **dns-update-key** and the actual key is **update**:

```
[admin@MikroTik] tool> dns-update dns-server=23.34.45.56 name=mydomain \
\... zone=myzone.com address=68.42.14.4 key-name=dns-update-key key=update
```

GPS Synchronization

Document revision 2.1 (January 23, 2008, 14:31 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Global Positioning System (GPS) receiver can be used by MikroTik RouterOS to get the precise location and time (which may be used as NTP time source)

Specifications

Packages required: ***gps***
License required: ***level1***
Home menu level: ***/system gps***
Standards and Technologies: ***GPS, NMEA 0183, Simple Text Output Protocol***
Hardware usage: ***Not significant***

Description

Global Positioning System (GPS) is used for determining precise location of a GPS receiver. There are two types of GPS service:

- Precise Positioning Service (PPS) that is used only by U. S. and Allied military, certain U. S. Government agencies, and selected civil users specifically approved by the U. S. Government. Its accuracy is 22m horizontally, 27.7m vertically and 200ns of time
- Standard Positioning Service (SPS) can be used by civil users worldwide without charge or restrictions except that SPS accuracy is intentionally degradated to 100m horizontally, 156m vertically and 340ns of time

GPS system is based on 24 satellites rotating on 6 different orbital planes with 12h orbital period. It makes that at least 5, but usually 6 or more satellites are visible at any time anywhere on the Earth. GPS receiver calculates more or less precise position (latitude, longitude and altitude), speed and time based on signals received from 4 satellites (three are used to determine 2D position and fourth is used to correct time and calculate altitude - 3D position), which are broadcasting information needed to calculate their exact current positions (using ephemeris - satelite own precise orbit - and almanach - information and coarse orbit of all satelites in the system) and UTC time (using GPS time and UTC deviation correction).

MikroTik RouterOS can communicate with many GPS receivers which are able to send the positioning and time via asynchronous serial line or USB using NMEA 0183, NMEA/RTCM or Simple Text Output Protocol. Note that you might need to configure the router's serial port in order to work with your device. For example, many GPS receivers work on 4800bit/s bitrate, to the same should be set in the **/port** menu for the respective serial port.

Precise time is mainly intended to be used by built-in NTP server, which can use it as a time source without any additional configuration if GPS is configured to set system time.

Additional Documents

- Global Positioning System - How it Works

Synchronizing with a GPS Receiver

Home menu level: ***/system gps***

Property Description

enabled (*yes* | *no*) - whether the router will communicate with a GPS receiver or not

port (*name*) - the port that will be used to communicate with a GPS receiver

set-system-time (*time*) - whether to set the system time to the value received from a GPS receiver or not

Notes

If you are synchronizing system time with a GPS device, you should correctly choose time zone if it is different from UTC as satellites are broadcasting time bound to UTC +00:00 timezone.

Example

To enable GPS communication through serial0 port:

```
[admin@MikroTik] system gps> print
            enabled: no
               port: (unknown)
    set-system-time: yes
[admin@MikroTik] system gps> set enabled=yes port=serial0
[admin@MikroTik] system gps> print
            enabled: yes
               port: serial0
    set-system-time: yes
[admin@MikroTik] system gps>
```

GPS Monitoring

Home menu level: ***/system gps monitor***

Description

This command is used for monitoring the data received from a GPS receiver.

Property Description

altitude (*read-only: text*) - altitude of the current location

date-and-time (*read-only: text*) - UTC date and time received from GPS server

latitude (*read-only: text*) - latitude of the current location

longitude (*read-only: text*) - longitude of the current location

speed (*read-only: text*) - mean velocity

valid (*read-only: yes | no*) - whether the received information is valid or not (e.g. you can set a GPS receiver to the demo mode to test the connection, in which case you will receive information, but it will not be valid)

Example

```
[admin@MikroTik] system gps> monitor
    date-and-time: jul/23/2003 12:25:00
        longitude: "E 24 8' 17''"
         latitude: "N 56 59' 22''"
         altitude: "-127.406400m"
            speed: "0.001600 km/h"
            valid: yes

[admin@MikroTik] system gps>
```

LCD Management

Document revision 2.6 (February 6, 2008, 4:17 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

LCDs are used to display system information.

The MikroTik RouterOS supports the following LCD hardware:

- Crystalfontz (http://www.crystalfontz.com) Intelligent Serial LCD Module 632 (16x2 characters) and 634 (20x4 characters)
- Powertip (http://www.powertip.com.tw) PC1602 (16x2 characters), PC1604 (16x4 characters), PC2002 (20x2 characters), PC2004 (20x4 characters), PC2402 (24x2 characters) and PC2404 (24x4 characters)
- Portwell (http://www.portwell.com.tw) EZIO-100 (16x2 characters)

Specifications

Packages required: ***lcd***
License required: ***level1***
Home menu level: ***/system lcd***
Standards and Technologies: ***None***
Hardware usage: ***Not significant***

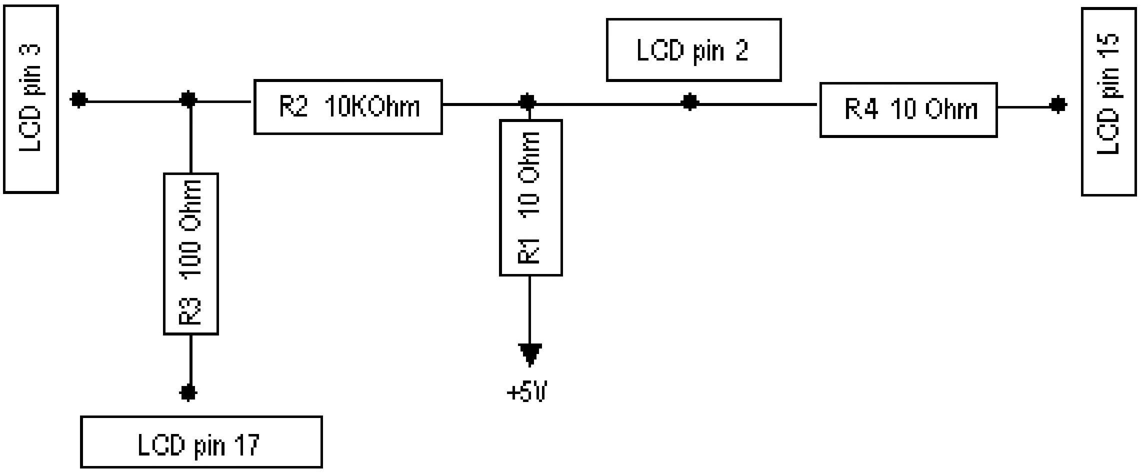
Description

How to Connect PowerTip LCD to a Parallel Port

Data signals are connected that way:

DB25m	Signal	LCD Panel
1	Enable (Strobe)	6
2	Data 0	7
3	Data 1	8
4	Data 2	9
5	Data 3	10
6	Data 4	11
7	Data 5	12
8	Data 6	13
9	Data 7	14
14	Register Select	4
18-25, GND	Ground	1, 5, 16

Powering:



As there are only 16 pins for the PC1602 modules, you need not connect power to the 17th pin.

GND and +5V can be taken from computer's internal power supply (use black wire for GND and red wire for +5V)

WARNING! Be very careful connecting power supply. We do not recommend using external power supplies. In no event shall MikroTik liable for any hardware damages.

Note that there are some PowerTip PC2404A modules that have different pin-out. Compare:

- From www.powertip.com.tw (probably newer one)
- From www.actron.de (probably older one)

Some LCDs may be connected without resistors:

DB25m	Signal	LCD Panel

18-25, GND	**Ground**	**1, 3, 4, 16**
+5V	**Power**	**2, 15**

Crystalfontz LCD Installation Notes

Before connecting the LCD, please check the availability of ports, their configuration, and free the desired port resource, if required:

```
[admin@MikroTik] port> print
  # NAME                      USED-BY                       BAUD-RATE
  0 serial0                      Serial Console               9600
  1 serial1                                                       9600
[admin@MikroTik] port>
```

The baud rate should be set to 9600 for use with the Crystalfontz serial LCD modules.

Portwell Installation Notes

The baud rate must be set to 2400 for Portwell LCD modules. The flow control must be set to **none**. The wiring for the DB9 to 10-pin female header cable is:

DB9 female	10-pin female header
2	**2**
3	**3**
5	**5**

Please note that the actual traces may not correspond to any of the documents coming from the manufacturer. It seems that all pin numbers of J2 are printed on the silkscreen in a "mirrored" way. Thus, the #1 pin is where the "5" is printed (the wiring above lists actual pin numbers, not the ones printed on the board).

Configuring the LCD's Settings

Home menu level: ***/system lcd***

Property Description

contrast (*integer*: 0..255; default: **0**) - contrast setting, sent to the LCD, if it supports contrast regulations

enabled (*yes* | *no*; default: **no**) - turns the LCD on or off

port (*nameparallel*; default: **parallel**) - name of the port where the LCD is connected. May be either one of the serial ports, or the first parallel

type (*16x2* | *16x4* | *20x2* | *20x4* | *24x2* | *24x4* | *mtb-134*; default: **24x4**) - sets the type of the LCD

- **mtb-134** - Portwell EZIO-100

Example

To enable Powertip parallel port LCD:

```
[admin@MikroTik] system lcd> print
   enabled: no
      type: 24x4
      port: parallel
  contrast: 0
[admin@MikroTik] system lcd> set enabled=yes
[admin@MikroTik] system lcd> print
   enabled: yes
      type: 24x4
      port: parallel
  contrast: 0
[admin@MikroTik] system lcd>
```

To enable Crystalfontz serial LCD on **serial1**:

```
[admin@MikroTik] system lcd> set port=serial1
[admin@MikroTik] system lcd> print
   enabled: yes
      type: 24x4
      port: serial1
  contrast: 0
[admin@MikroTik] system lcd>
```

LCD Information Display Configuration

Home menu level: ***/system lcd page***

Description

The submenu is used for configuring LCD information display: what pages and how long will be shown.

Property Description

description (*read-only: text*) - page description

display-time (*time*; default: **5s**) - how long to display the page

Notes

You cannot neither add your own pages (they are created dynamically depending on the configuration) nor change pages' description.

Example

To enable displaying all the pages:

```
[admin@MikroTik] system lcd page> print
Flags: X - disabled
  #   DISPLAY-TIME    DESCRIPTION
  0 X 5s            System date and time
  1 X 5s            System resources - cpu and memory load
  2 X 5s            System uptime
  3 X 5s            Aggregate traffic in packets/sec
  4 X 5s            Aggregate traffic in bits/sec
  5 X 5s            Software version and build info
  6 X 5s            ether1
  7 X 5s            prism1
[admin@MikroTik] system lcd page> enable [find]
[admin@MikroTik] system lcd page> print
Flags: X - disabled
  #   DISPLAY-TIME    DESCRIPTION
  0   5s            System date and time
  1   5s            System resources - cpu and memory load
  2   5s            System uptime
  3   5s            Aggregate traffic in packets/sec
  4   5s            Aggregate traffic in bits/sec
  5   5s            Software version and build info
  6   5s            ether1
  7   5s            prism1
[admin@MikroTik] system lcd page>
```

To set "System date and time" page to be displayed for 10 seconds:

```
[admin@MikroTik] system lcd page> set 0 display-time=10s
[admin@MikroTik] system lcd page> print
Flags: X - disabled
  #   DISPLAY-TIME    DESCRIPTION
  0   10s            System date and time
  1   5s            System resources - cpu and memory load
  2   5s            System uptime
  3   5s            Aggregate traffic in packets/sec
  4   5s            Aggregate traffic in bits/sec
  5   5s            Software version and build info
  6   5s            ether1
  7   5s            prism1
[admin@MikroTik] system lcd page>
```

LCD Troubleshooting

Description

LCD doesn't work, cannot be enabled by the '/system lcd set enabled=yes' command.

Probably the selected serial port is used by PPP client or server, or by the serial console. Check the availability and use of the ports by examining the output of the **/port print** command. Alternatively, select another port for connecting the LCD, or free up the desired port by disabling the related resource

LCD doesn't work, does not show any information.

Probably none of the information display items have been enabled. Use the */system lcd page set* command to enable the display.

MNDP

Document revision 1.5 (January 23, 2008, 16:06 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The MikroTik Neighbor Discovery Protocol (MNDP) eases network configuration and management by enabling each MikroTik router to discover other connected MikroTik routers and learn information about the system along with features which are enabled. The MikroTik routers can automatically use learned information to set up some features with minimal or no configuration.

MNDP features:

- works on IP level connections
- works on all non-dynamic interfaces
- distributes basic information on the software version
- distributes information on configured features that should interoperate with other MikroTik routers

MikroTik RouterOS is able to discover both MNDP and CDP (Cisco Discovery Protocol) devices.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/ip neighbor***
Standards and Technologies: ***MNDP***
Hardware usage: ***Not significant***

Related Documents

- Package Management
- M3P

Description

MNDP basic function is to assist with automatic configuration of features that are only available between MikroTik routers. Currently this is used for the 'Packet Packer' feature. The 'Packet Packer' may be enabled on a per interface basis. The MNDP protocol will then keep information about what routers have enabled the 'unpack' feature and the 'Packet Packer' will be used for traffic between these routers.

Specific features

- works on interfaces that support IP protocol and have at least one IP address and on all ethernet-like interfaces even without IP addresses
- is enabled by default for all new Ethernet-like interfaces -- Ethernet, wireless, EoIP, IPIP tunnels, PPTP-static-server
- when older versions on the RouterOS are upgraded from a version without discovery to a version with discovery, current Ethernet like interfaces will not be automatically enabled for MNDP
- uses UDP protocol port 5678
- a UDP packet with router info is broadcasted over the interface every 60 seconds
- every 30 seconds, the router checks if some of the neighbor entries are not stale
- if no info is received from a neighbor for more than 180 seconds the neighbor information is discarded

Setup

Home menu level: ***/ip neighbor discovery***

Property Description

discover (yes | no; default: **yes**) - specifies whether the neighbour discovery is enabled or not

name (*read-only: name*) - interface name for reference

Example

To disable MNDP protocol on Public interface:

```
[admin@MikroTik] ip neighbor discovery> set Public discover=no
[admin@MikroTik] ip neighbor discovery> print
  # NAME      DISCOVER
  0 Public    no
  1 Local     yes
```

Neighbour List

Home menu level: ***/ip neigbor***

Description

This submenu allows you to see the list of neighbours discovered

Property Description

address (*read-only: IP address*) - IP address of the neighbour router

age (*read-only: time*) - specifies the record's age in seconds (time from the last update)

identity (*read-only: text*) - system identity of the neighbour router

interface (*read-only: name*) - local interface name the neighbour is connected to

mac-address (*read-only: MAC address*) - MAC address of the neighbour router

platform (*read-only: text*) - hardware/software platform type of neighbour router

softwate-id (*read-only: text*) - Software ID of the neighbout MikroTik RouterOS router

unpack (*read-only: none* | *simple* | *compress-headers* | *compress-all*) - identifies if the interface of the neighbour router is unpacking packets packed with M3P

uptime (*read-only: time*) - uptime of the neighbour router

version (*read-only: text*) - operating system or firmware version of the neighbour router

Example

To view the table of discovered neighbours:

```
[admin@MikroTik] ip neighbor> pri
 # INTERFACE ADDRESS         MAC-ADDRESS       IDENTITY   VERSION
 0 ether2    10.1.0.113      00:0C:42:00:02:06 ID         2.9beta5
 1 ether2    1.1.1.3         00:0C:42:03:02:ED MikroTik   2.9beta5
[admin@MikroTik] ip neighbor>
```

System Clock and Simple SNTP Client

Document revision .NaN (January 23, 2008, 14:30 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

System Clock

Summary

System clock allows router to track current date and time.

Specifications

License required: ***level1***
Home menu level: ***/system clock***

Property Description

date (*text*) - date in format "mm/DD/YYY"

gmt-offset (*read-only: text*) - UTC timezone in format "+HH:MM" or "-HH:MM"

time (*time*) - time in format "HH:MM:SS"

time-zone-name (*text*; default: **manual**) - name of the timezone (usually, identified by a major city or a country). UTC offset and DST information of the selected location is used

- **manual** - UTC offset and DST activation is set manually

Notes

It is recommended that you reboot the router after time change to obviate the possible errors in time measurments and logging.

Date and time settings become permanent and effect BIOS settings.

If NTP update gives time shifted by 1 hour, although the time zone is set correctly, you may want to adjust the DST setting in **/system clock manual** menu.

Example

To view the current date and time settings:

```
[admin@Local] system clock> print
            time: 09:08:37
            date: nov/18/2007
  time-zone-name: "manual"
      gmt-offset: +00:00
[admin@Local] system clock>
```

To set the system date and time to EET:

```
[admin@Local] system clock> set date=nov/22/2022 time=11:10:21 time-zone-name=EET
[admin@Local] system clock> print
        time: 11:10:25
        date: nov/18/2007
  time-zone-name: "EET"
      gmt-offset: +02:00
[admin@Local] system clock>
```

Manual Time Zone Settings

Home menu level: ***/system clock manual***

Description

Usually the time zone and associated DST activation/deactivation time is properly configured when the major nearby location is specified in the **time-zone-name** parameter. In most countries, a Daylight Saving Time regime is activated in spring and deactivated in autumn. This configuration menu provides UTC timezone and DST adjustment facility, to set and drift the timezone according to your local legislation and practice if it is not set correctly by selecting an appropriate time zone name.

Property Description

dst-delta (*text*; default: **+01:00**) - UTC timezone drift in format "+HH:MM" or "-HH:MM" to be added to the local timezone during DST period

dst-end (*datetime*) - date and time when DST ends (when the delta is to be dropped).

dst-start (*datetime*) - date and time when DST begins (when the delta is to be applied).

time-zone (*text*) - UTC offset of the desired time zone in format "+HH:MM" or "-HH:MM"

Example

For EET timezone and DST zonechange active from **mar/27/2005 03:00:00** till **oct/30/2005 03:00:00**:

```
[admin@MikroTik] system clock> set time-zone-name=manual
[admin@MikroTik] system clock> manual set time-zone=+02:00 dst-delta=+01:00 \
\... dst-start="mar/27/2005 03:00:00" dst-end="oct/30/2005 03:00:00"
```

```
[admin@MikroTik] system clock> manual print
  time-zone: +02:00
  dst-delta: +01:00
  dst-start: mar/27/2005 03:00:00
    dst-end: oct/30/2005 03:00:00
[admin@MikroTik] system clock dst>
```

Simple SNTP Client

Home menu level: ***/system ntp client***
Standards and Technologies: ***SNTP version 4 (RFC 2030)***

Description

NTP protocol allows synchronizing time among computers in network. It is good if there is an internet connection available and local NTP server is synchronized to correct time source. List of public NTP servers is available at http://www.eecis.udel.edu/~mills/ntp/servers.html. SNTP is a simplified version of NTP, compatible with virtually all internet NTP servers, but lacks somevery high precision internal algorithms (and thus have significantly lower requirements and smaller memory footprint). There is also a full NTP client and server implementation for RouterOS available in a separate package (**ntp** package is available for download from www.mikrotik.com) with higher system requirements and more features, but the small SNTP client included in the **system** package is sufficient in most cases. Note that the software included in the **ntp** uses the same configuration menu, so you cannot use both NTP and SNTP at the same time.

Property Description

active-server (*read-only: IP address*) - server, the client is communicating with (unicast only)

enabled (*yes* | *no*; default: **no**) - whether the SNTP client is enabled or not

last-adjustment (*read-only: time*) - last time adjustment delta (difference between the local clock state and the received time during the last update)

last-bad-packet-before (*read-only: time*) - time since the last unaccepted NTP message has been received

last-bad-packet-from (*read-only: IP address*) - server address, which sent the last unaccepted message

last-bad-packet-reason (*read-only: text*) - reason that states why has the last unaccepted message been discarded

last-update-before (*read-only: time*) - time past since the last clock update

last-update-from (*read-only: IP address*) - IP address of the3 server that sent last accepted message, that was used to adjust clock

mode (*unicast* | *broadcast*; default: **broadcast**) - NTP client mode

- **broadcast** - NTP client listens for broadcast messages sent by any NTP server. After receiving first broadcast message, client synchronizes local clock using unicast mode, and afterwards does not send any packets to that particular NTP server, but rather waits for the next broadcast messages
- **unicast** - NTP client connects to the specified NTP server. IP address of NTP server must be set in ntp-server and/or second-ntp-server parameters. At first client synchronizes to NTP server. Afterwards client periodically (64..1024s) sends time requests to NTP server

poll-interval (*read-only: time*) - current interval between messages sent to server (unicast only)

primary-ntp (*IP address*; default: **0.0.0.0**) - specifies IP address of the primary NTP server

secondary-ntp (*IP address*; default: **0.0.0.0**) - specifies IP address of the secondary NTP server

Notes

CAUTION! Using **broadcast** mode is dangerous! Intruder (or simple user) can set up his/her own NTP server. If this new server will be chosen as time source for your router, it will be possible for this user to change time on your router at his/her will.

Example

To enable the NTP client to synchronize with the **159.148.60.11** server:

```
[admin@MikroTik] system ntp client> set enabled=yes primary-ntp=159.148.60.2 \
\... mode=unicast
[admin@MikroTik] system ntp client> print
             enabled: yes
                mode: unicast
         primary-ntp: 159.148.60.11
       secondary-ntp: 0.0.0.0
       poll-interval: 8m32s
       active-server: 159.148.60.11
    last-update-from: 159.148.60.11
  last-update-before: 1m38s120ms
     last-adjustment: 2ms562us
[admin@MikroTik] system ntp client>
```

NTP Server and Client

Document revision 1.1 (January 23, 2008, 14:31 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

NTP protocol allows synchronizing time among computers in network. It is good if there is an internet connection available and local NTP server is synchronized to correct time source. List of publec NTP servers is available at http://www.eecis.udel.edu/~mills/ntp/servers.html. Note that if NTP client is all you need, you may want to consider using SNTP client instead for it has lower system requirements and significantly smaller memory footprint. It is included in the **system** package and is activated once **ntp** package is disabled or uninstalled.

Specifications

Packages required: ***ntp***
License required: ***level1***
Home menu level: ***/system ntp***
Standards and Technologies: ***NTP version 3 (RFC 1305)***
Hardware usage: ***Not significant***

Description

Network Time Protocol (NTP) is used to synchronize time with some NTP servers in a network. MikroTik RouterOS provides both - NTP client and NTP server.

NTP server listens on UDP port 123

NTP client synchronizes local clock with some other time source (NTP server). There are 4 modes in which NTP client can operate at:

- **unicast** (Client/Server) mode - NTP client connects to the specified NTP server. IP address of NTP server must be set in ntp-server and/or second-ntp-server parameters. At first client synchronizes to NTP server. Afterwards client periodically (64..1024s) sends time requests to NTP server. Unicast mode is the only one which uses ntp-server and second-ntp-server parameters.

- **broadcast** mode - NTP client listens for broadcast messages sent by any NTP server. After receiving first broadcast message, client synchronizes local clock using unicast mode, and afterwards does not send any packets to that particular NTP server, but rather waits for the next broadcast messages.
- **multicast** mode - acts the same as broadcast mode, only instead of broadcast messages (IP address 255.255.255.255) multicast messages are received (IP address 224.0.1.1).
- **manycast** mode - actually is unicast mode only with unknown IP address of NTP server. To discover NTP server, client sends multicast message (IP 239.192.1.1). If NTP server is configured to listen for these multicast messages (manycast mode is enabled), it replies. After client receives reply, it enters unicast mode and synchronizes to that NTP server. But in parallel client continues to look for more NTP servers by sending multicast messages periodically.

Client

Home menu level: ***/system ntp client***

Property Description

enabled (*yes* | *no*; default: **no**) - whether the NTP client is enabled or not

mode (*unicast* | *broadcast* | *multicast* | *manycast*; default: **unicast**) - NTP client mode

primary-ntp (*IP address*; default: **0.0.0.0**) - specifies IP address of the primary NTP server

secondary-ntp (*IP address*; default: **0.0.0.0**) - specifies IP address of the secondary NTP server

status (*read-only: text*) - status of the NTP client:

- **stopped** - NTP is not running (NTP is disabled)
- **error** - there was some internal error starting NTP service (please, try to restart (disable and enable) NTP service)
- **started** - NTP client service is started, but NTP server is not found, yet
- **failed** - NTP server sent invalid response to our NTP client (NTP server is not synchronized to some other time source)
- **reached** - NTP server contacted. Comparing local clock to NTP server's clock (duration of this phase is approximately 30s)
- **timeset** - local time changed to NTP server's time (duration of this phase is approximately 30s)
- **synchronized** - local clock is synchronized to NTP server's clock. NTP server is activated
- **using-local-clock** - using local clock as time source (server enabled while client disabled)

Example

To enable the NTP client to synchronize with the **159.148.60.2** server:

```
[admin@MikroTik] system ntp client> set enabled=yes primary-ntp=159.148.60.2
[admin@MikroTik] system ntp client> print
          enabled: yes
             mode: unicast
      primary-ntp: 159.148.60.2
```

```
    secondary-ntp: 0.0.0.0
           status: synchronized
[admin@MikroTik] system ntp client>
```

Server

Home menu level: ***/system ntp server***

Property Description

broadcast (*yes* | *no*; default: **no**) - whether NTP broadcast message is sent to 255.255.255.255 every 64s

enabled (*yes* | *no*; default: **no**) - whether the NTP server is enabled

manycast (*yes* | *no*; default: **yes**) - whether NTP server listens for multicast messages sent to 239.192.1.1 and responds to them

multicast (*yes* | *no*; default: **no**) - whether NTP multicast message is sent to 224.0.1.1 every 64s

Notes

NTP server activities only when local NTP client is in **synchronized** or **using-local-clock** mode.

If NTP server is disabled, all NTP requests are ignored.

If NTP server is enabled, all individual time requests are answered.

CAUTION! Using **broadcast**, **multicast** and **manycast** modes is dangerous! Intruder (or simple user) can set up his own NTP server. If this new server will be chosen as time source for your server, it will be possible for this user to change time on your server at his will.

Example

To enable NTP server to answer unicast requests only:

```
[admin@MikroTik] system ntp server> set manycast=no enabled=yes
[admin@MikroTik] system ntp server> print
      enabled: yes
    broadcast: no
    multicast: no
     manycast: no
[admin@MikroTik] system ntp server>
```

Support Output File

Document revision 2.2 (January 23, 2008, 16:06 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The support file is used for debugging MikroTik RouterOS and to solve the support questions faster. All MikroTik Router information is saved in a binary file, which is stored on the router and can be downloaded from the router using ftp. This file does not contain RouterOS user paswords, so you are not disclosing access information by sending us this file.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/system***
Hardware usage: ***Not significant***

Generating Support Output File

Command name: ***/system sup-output***

Example

To make a Support Output File:

```
[admin@MikroTik] > system sup-output
creating supout.rif file, might take a while
...................
done
[admin@MikroTik] >
```

To see the files stored on the router:

```
[admin@MikroTik] > file print
 # NAME                           TYPE SIZE                 CREATION-TIME
 0 supout.rif                     .r.. 277042               jan/23/2008 18:03:29
[admin@MikroTik] >
```

Connect to the router using FTP and download the supout.rif file using BINARY file transfer mode. Send the supout.rif file to MikroTik Support support@mikrotik.com with detailed description of the problem.

System Resource Management

Document revision 2.4 (January 24, 2008, 11:16 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

MikroTik RouterOS offers several features for monitoring and managing the system resources.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/system***
Standards and Technologies: ***None***
Hardware usage: ***Not significant***

System Resource

Home menu level: ***/system resource***

Notes

In **monitor** command priotout the values for cpu usage and free memory are in percentage and kilobytes, respectively.

Example

To view the basic system resource status:

```
[admin@MikroTik] system resource> print
                   uptime: 5h26m12s
                  version: "3.0"
              free-memory: 17000kB
             total-memory: 30200kB
                    model: "RouterBOARD 500"
                      cpu: "MIPS 4Kc V0.10"
                cpu-count: 1
            cpu-frequency: 333MHz
                 cpu-load: 3
           free-hdd-space: 14208kB
          total-hdd-space: 61440kB
  write-sect-since-reboot: 1047
         write-sect-total: 379983
               bad-blocks: 0
[admin@MikroTik] system resource>
```

To view the current system CPU usage and free memory:

```
[admin@MikroTik] > system resource monitor
       cpu-used: 0
    free-memory: 115676

[admin@MikroTik] >
```

IRQ Usage Monitor

Command name: ***/system resource irq print***

Description

IRQ usage shows which IRQ (Interrupt requests) are currently used by hardware.

Example

```
[admin@MikroTik] > system resource irq print
Flags: U - unused
   IRQ OWNER
   1   keyboard
   2   APIC
 U 3
   4   serial port
   5   [Ricoh Co Ltd RL5c476 II (#2)]
 U 6
 U 7
 U 8
 U 9
 U 10
   11  ether1
   12  [Ricoh Co Ltd RL5c476 II]
 U 13
   14  IDE 1
[admin@MikroTik] >
```

IO Port Usage Monitor

Command name: ***/system resource io print***

Description

IO usage shows which IO (Input/Output) ports are currently used by hardware.

Example

```
[admin@MikroTik] > system resource io print
 PORT-RANGE        OWNER
 0x20-0x3F         APIC
 0x40-0x5F         timer
 0x60-0x6F         keyboard
 0x80-0x8F         DMA
 0xA0-0xBF         APIC
 0xC0-0xDF         DMA
 0xF0-0xFF         FPU
 0x1F0-0x1F7       IDE 1
 0x2F8-0x2FF       serial port
 0x3C0-0x3DF       VGA
 0x3F6-0x3F6       IDE 1
 0x3F8-0x3FF       serial port
 0xCF8-0xCFF       [PCI conf1]
 0x4000-0x40FF     [PCI CardBus #03]
 0x4400-0x44FF     [PCI CardBus #03]
 0x4800-0x48FF     [PCI CardBus #04]
 0x4C00-0x4CFF     [PCI CardBus #04]
 0x5000-0x500F     [Intel Corp. 82801BA/BAM SMBus]
 0xC000-0xC0FF     [Realtek Semiconductor Co., Ltd. RTL-8139/8139C/8139C+]
 0xC000-0xC0FF     [8139too]
 0xC400-0xC407     [Cologne Chip Designs GmbH ISDN network controller [HFC-PCI]
 0xC800-0xC87F     [Cyclades Corporation PC300/TE (1 port)]
 0xF000-0xF00F     [Intel Corp. 82801BA IDE U100]

[admin@MikroTik] >
```

USB Port Information

Command name: ***/system resource usb print***

Description

Shows all USB ports available for the router. Only available on the routers, supporting USB.

Property Description

device (*read-only: text*) - number of device

name (*read-only: text*) - name of the USB port

speed (*read-only: integer*) - bandwidth speed at which the port works

vendor (*read-only: text*) - vendor name of the USB device

Example

To list all available USB ports:

```
[admin@MikroTik] system resource usb> print
 # DEVICE VENDOR                  NAME                   SPEED
 0 1:1                            USB OHCI Root Hub      12 Mbps
[admin@MikroTik] system resource usb>
```

PCI Information

Command name: ***/system resource pci print***

Property Description

category (*read-only: text*) - device type

device (*read-only: text*) - number of device

device-id (*read-only: integer*) - hexadecimal device ID

irq (*read-only: integer*) - IRQ number which this device uses

memory (*read-only: integer*) - memory range this device uses

name (*read-only: text*) - name of the device

vendor (*read-only: text*) - vendor name of the device

vendor-id (*read-only: integer*) - hexadecimal vendor ID of the device

Example

To see PCI slot details:

```
[admin@MikroTik] system resource pci> print
 # DEVICE   VENDOR                       NAME                        IRQ
 0 00:13.0  Compaq                       ZFMicro Chipset USB (rev... 12
 1 00:12.5  National Semi                SC1100 XBus (rev: 0)
 2 00:12.4  National Semi                SC1100 Video (rev: 1)
 3 00:12.3  National Semi                SCx200 Audio (rev: 0)
 4 00:12.2  National Semi                SCx200 IDE (rev: 1)
 5 00:12.1  National Semi                SC1100 SMI (rev: 0)
 6 00:12.0  National Semi                SC1100 Bridge (rev: 0)
 7 00:0e.0  Atheros Communications       AR5212 (rev: 1)             10
 8 00:0d.1  Texas Instruments           PCI1250 PC card Cardbus ... 11
 9 00:0d.0  Texas Instruments           PCI1250 PC card Cardbus ... 11
10 00:0c.0  National Semi               DP83815 (MacPhyter) Ethe... 10
11 00:0b.0  National Semi               DP83815 (MacPhyter) Ethe... 9
12 00:00.0  Cyrix Corporation           PCI Master (rev: 0)
[admin@MikroTik] system resource pci>
```

Reboot

Command name: ***/system reboot***

Description

The system reboot is required when upgrading or installing new software packages. The packages are installed during the system shutdown.

The reboot process sends termination signal to all running processes, unmounts the file systems, and reboots the router.

Notes

Only users, which are members of groups with reboot privileges are permitted to reboot the router.

Reboot can be called from scripts, in which case it does not prompt for confirmation.

Example

```
[admin@MikroTik] > system reboot
Reboot, yes? [y/N]: y
system will reboot shortly
[admin@MikroTik] >
```

Shutdown

Command name: ***/system shutdown***

Description

Before turning the power off for the router, the system should be brought to halt. The shutdown process sends termination signal to all running processes, unmounts the file systems, and halts the router.

For some systems, it is necessary to wait up to 30 seconds (but usually less than 10 seconds if there is no upgrade scheduled) for a safe power down.

Notes

Only users, which are members of groups with reboot privileges are permitted to shutdown the router.

Shutdown can be called from scripts, in which case it does not prompt for confirmation.

Example

```
[admin@MikroTik] > system shutdown
Shutdown, yes? [y/N]: y
system will shutdown promptly
[admin@MikroTik] >
```

Router Identity

Home menu level: ***/system identity***

Description

The router identity is displayed before the command prompt. It is also used for DHCP client as 'host name' parameter when reporting it to the DHCP server.

Example

To view the router identity:

```
[admin@MikroTik] > system identity print
    name: "MikroTik"
[admin@MikroTik] >
```

To set the router identity:

```
[admin@MikroTik] > system identity set name=Gateway
[admin@Gateway] >
```

Configuration Change History

Home menu level: Command name: ***/system history, /undo, /redo***

Description

The history of system configuration changes is held until the next router shutdown. The invoked commands can be 'undone' (in reverse order they have been invoked). The 'undone' commands may be 'redone' (in reverse order they have been 'undone').

Command Description

/redo - undoes previous '/undo' command

/system history print - print a list of last configuration changes, specifying whether the action can be undone or redone

/undo - undoes previous configuration changing command (except another '/undo' command)

Notes

Floating-undo actions are created within the current SAFE mode session. They are automatically converted to undoable and redoable when SAFE mode terminated successfully, and are all undone irreverively when SAFE mode terminated insuccessfully.

Undo command cannot undo commands past start of the SAFE mode.

Example

To show the list of configuration changes:

```
[admin@MikroTik] system history> print
Flags: U - undoable, R - redoable, F - floating-undo
   ACTION                                   BY                 POLICY
 U system time zone changed                 admin              write
 U system time zone changed                 admin              write
 U system time zone changed                 admin              write
 U system identity changed                  admin              write
[admin@MikroTik] system clock>
```

What the **/undo** command does:

```
[admin@MikroTik] system history> print
Flags: U - undoable, R - redoable, F - floating-undo
   ACTION                                   BY                 POLICY
 R system time zone changed                 admin              write
 U system time zone changed                 admin              write
 U system time zone changed                 admin              write
 U system identity changed                  admin              write
[admin@MikroTik] system clock>
```

System Note

Home menu level: ***/system note***

Description

System note feature allows you to assign arbitrary text notes or messages that will be displayed on each login right after banner. For example, you may distribute warnings between system administrators this way, or describe what does that particular router actually do. To configure system note, you may upload a plain text file named **sys-note.txt** on the router's FTP server, or, additionally, edit the settings in this menu

Property Description

note (*text*; default: **""**) - the note

show-at-login (yes | no; default: **yes**) - whether to show system note on each login

Notes

If you want to enter or edit multiline system note, you may need to use embedded text editor: `/system note edit note`

Bandwidth Test

Document revision 1.10 (January 24, 2008, 11:22 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The Bandwidth Tester can be used to monitor the throughput only to a remote MikroTik router (either wired or wireless) and thereby help to discover network "bottlenecks".

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/tool***
Standards and Technologies: ***TCP (RFC 793)***, ***UDP (RFC768)***
Hardware usage: *significant*

Description

Protocol Description

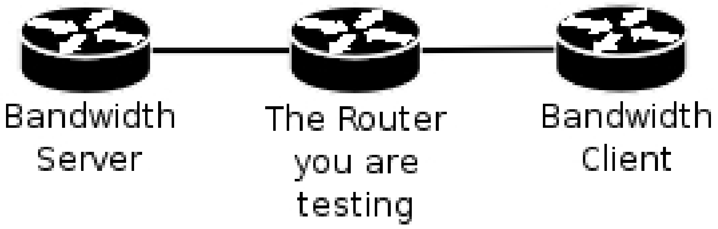
The TCP test uses the standard TCP protocol with acknowledgments and follows the TCP algorithm on how many packets to send according to latency, dropped packets, and other features in the TCP algorithm. Please review the TCP protocol for details on its internal speed settings and how to analyze its behavior. Statistics for throughput are calculated using the entire size of the TCP data stream. As acknowledgments are an internal working of TCP, their size and usage of the link are not included in the throughput statistics. Therefore this statistic is not as reliable as the UDP statistic when estimating throughput.

The UDP tester sends 110% or more packets than currently reported as received on the other side of the link. To see the maximum throughput of a link, the packet size should be set for the maximum MTU allowed by the links which is usually 1500 bytes. There is no acknowledgment required by UDP; this implementation means that the closest approximation of the throughput can be seen.

Usage Notes

Caution! Bandwidth Test uses all available bandwidth (by default) and may impact network usability.

Bandwidth Test uses much resources. If you want to test real throughput of a router, you should run bandwidth test through it not from or to it. To do this you need at least 3 routers connected in chain: the Bandwidth Server, the given router and the Bandwidth Client:



Note that if you use UDP protocol then Bandwidth Test counts IP header+UDP header+UDP data. In case if you use TCP then Bandwidth Test counts only TCP data (TCP header and IP header are not included).

Server Configuration

Home menu level: ***/tool bandwidth-server***

Property Description

allocate-udp-ports-from - allocate UDP ports from

authenticate (*yes* | *no*; default: **yes**) - communicate only with authenticated (by valid username and password) clients

enable (*yes* | *no*; default: **no**) - enable client connections for bandwidth test

max-sessions - maximal number of bandwidth-test clients

Notes

The list of current connections can be obtained in **session** submenu

Example

Bandwidth Server:

```
[admin@MikroTik] tool bandwidth-server> print
                  enabled: yes
             authenticate: yes
  allocate-udp-ports-from: 2000
             max-sessions: 10
[admin@MikroTik] tool>
```

Active sessions:

```
[admin@MikroTik] tool> bandwidth-server session print
  # CLIENT          PROTOCOL DIRECTION USER
  0 35.35.35.1      udp      send      admin
  1 25.25.25.1      udp      send      admin
  2 36.36.36.1      udp      send      admin

[admin@MikroTik] tool>
```

To enable **bandwidth-test** server without client authentication:

```
[admin@MikroTik] tool bandwidth-server> set enabled=yes authenticate=no
[admin@MikroTik] tool bandwidth-server> print
                  enabled: yes
             authenticate: no
  allocate-udp-ports-from: 2000
             max-sessions: 10
[admin@MikroTik] tool>
```

Client Configuration

Command name: ***/tool bandwidth-test***

Property Description

(*IP address*) - IP address of destination host

assume-lost-time (*time*; default: **0s**) - assume that connection is lost if Bandwidth Server is not responding for that time

direction (*receive*/*transmit*/*both*; default: **receive**) - the direction of the test

do (*name* | *string*; default: **""**) - script source

duration (*time*; default: **0s**) - duration of the test

- **0s** - test duration is not limited

interval (*time*: 20ms..5s; default: **1s**) - delay between reports (in seconds)

local-tx-speed (*integer*; default: **0**) - transfer test maximum speed (bits per second)

- **0** - no speed limitations

local-udp-tx-size (*integer*: 40..64000) - local transmit packet size in bytes

password (*text*; default: **""**) - password for the remote user

protocol (*udp* | *tcp*; default: **udp**) - protocol to use

random-data (*yes* | *no*; default: **no**) - if random-data is set to yes, the payload of the bandwidth test packets will have incompressible random data stream so that links that use data compression will not distort the results (this is CPU intensive and random-data should be set to no for low speed CPUs)

remote-tx-speed (*integer*; default: **0**) - receive test maximum speed (bits per second)

- **0** - no speed limitations

remote-udp-tx-size (*integer*: 40..64000) - remote transmit packet size in bytes

user (*name*; default: **""**) - remote user

Example

To run 15-second long bandwidth-test to the **10.0.0.211** host sending and receiving **1000**-byte UDP packets and using username **admin** to connect

```
[admin@MikroTik] tool> bandwidth-test 10.0.0.211 duration=15s direction=both \
\... size=1000 protocol=udp user=admin
                status: done testing
                duration: 15s
              tx-current: 3.62Mbps
    tx-10-second-average: 3.87Mbps
        tx-total-average: 3.53Mbps
              rx-current: 3.33Mbps
    rx-10-second-average: 3.68Mbps
        rx-total-average: 3.49Mbps

[admin@MikroTik] tool>
```

ICMP Bandwidth Test

Document revision 1.3 (January 24, 2008, 15:28 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The ICMP Bandwidth Tester (Ping Speed) can be used to approximately evaluate the throughput to **any** remote computer and thereby help to discover network 'bottlenecks'.

Specifications

Packages required: ***advanced-tools***
License required: ***level1***
Command name: ***/tool ping-speed***
Standards and Technologies: ***ICMP (RFC792)***
Hardware usage: ***Not significant***

ICMP Bandwidth Test

Description

The ICMP test uses two standard echo-requests per second. The time between these pings can be changed. Ping packet size variation makes it possible to approximately evaluate connection parameters and speed with different packet sizes. Statistics for throughput is calculated using the size of the ICMP packet, the interval between ICMP echo-request and echo-reply and the differences between parameters of the first and the second packet. Note that the tool can not estimate the channel throughput exactly and can show imprecise results.

Property Description

(*IP address*) - IP address to ping

do (*name*) - assigned name of the script to start

first-ping-size (*integer*: 32..64000; default: **32**) - first ICMP packet size

interval (*time*: 20ms..5s) - time interval between two ping repetitions

once - specifies that the ping will be performed only once

second-ping-size (*integer*: 32..64000; default: **1500**) - second ICMP packet size

time-between-pings (*integer*) - the time between the first and the second ICMP echo-requests in seconds. A new ICMP-packet pair will never be sent before the previous pair is completely sent and the algorithm itself will never send more than two requests in one second

Example

In the following example we will test the bandwidth to a host with IP address **159.148.60.2**. The interval between repetitions will be **1** second.

```
[admin@MikroTik] tool> ping-speed 159.148.60.2 interval=1s
    current: 2.23Mbps
    average: 2.61Mbps

[admin@MikroTik] tool>
```

Packet Sniffer

Document revision 1.6 (February 5, 2008, 15:52 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Packet sniffer is a feature that catches all the data travelling over the network, that it is able to get (when using switched network, a computer may catch only the data addressed to it or is forwarded through it).

Specifications

Packages required: ***system***
License required: ***level1***

Home menu level: ***/tool sniffer***
Standards and Technologies: ***none***
Hardware usage: ***Not significant***

Description

It allows you to "sniff" (listen and record) packets going through the router (and any other traffic that gets to the router, when there is no switching in the network) and view them using specific software.

Packet Sniffer Configuration

Home menu level: ***/tool sniffer***

Property Description

file-limit (*integer*; default: **10**) - the limit of the file in KB. Sniffer will stop after this limit is reached

file-name (*text*; default: **""**) - the name of the file where the sniffed packets will be saved to

filter-address1 (*IP addressnetmaskport*; default: **0.0.0.0/0:0-65535**) - criterion of choosing the packets to process

filter-address2 (*IP addressnetmaskport*; default: **0.0.0.0/0:0-65535**) - criterion of choosing the packets to process

filter-protocol (*all-frames* | *ip-only* | *mac-only-no-ip*; default: **ip-only**) - specific protocol group to filter

- **all-frames** - sniff all packets
- **ip-only** - sniff IP packets only
- **mac-only-no-ip** - sniff non-IP packets only

filter-stream (*yes* | *no*; default: **yes**) - whether to ignore sniffed packets that are destined to the stream server

interface (*name* | *all*; default: **all**) - the name of the interface that receives the packets

memory-limit (*integer*; default: **10**) - maximum amount of memory to use. Sniffer will stop after this limit is reached

only-headers (*yes* | *no*; default: **no**) - whether to save in the memory packets' headers only (not the whole packet)

running (*read-only: yes* | *no*; default: **no**) - if the sniffer is started then the value is yes otherwise no

streaming-enabled (*yes* | *no*; default: **no**) - whether to send sniffed packets to a remote server

streaming-server (*IP address*; default: **0.0.0.0**) - Tazmen Sniffer Protocol (TZSP) stream receiver

Notes

filter-address1 and **filter-address2** are used to specify the two participients in communication (i.e. they will match only in the case if one of them matches the source address and the other one matches the destination address of a packet). These properties are taken in account only if **filter-protocol** is **ip-only**.

Not only **Wireshark** (ex-Ethereal, http://www.wireshark.org) and **Packetyzer** (http://www.packetyzer.com) can receive the sniffer's stream but also MikroTik's program **trafr** http://www.mikrotik.com/download.html) that runs on any IA32 Linux computer and saves received packets **libpcap** file format.

Example

In the following example **streaming-server** will be added, streaming will be enabled, **file-name** will be set to *test* and packet sniffer will be started and stopped after some time:

```
[admin@MikroTik] tool sniffer>set streaming-server=10.0.0.241 \
\... streaming-enabled=yes file-name=test
[admin@MikroTik] tool sniffer> prin
            interface: all
         only-headers: no
         memory-limit: 10
            file-name: "test"
           file-limit: 10
    streaming-enabled: yes
     streaming-server: 10.0.0.241
        filter-stream: yes
      filter-protocol: ip-only
      filter-address1: 0.0.0.0/0:0-65535
      filter-address2: 0.0.0.0/0:0-65535
              running: no
[admin@MikroTik] tool sniffer>start
[admin@MikroTik] tool sniffer>stop
```

Running Packet Sniffer

Command name: ***/tool sniffer start, /tool sniffer stop, /tool sniffer save***

Description

The commands are used to control runtime operation of the packet sniffer. The **start** command is used to start/reset sniffering, **stop** - stops sniffering. To save currently sniffed packets in a specific file **save** command is used.

Example

In the following example the packet sniffer will be started and after some time - stopped:

```
[admin@MikroTik] tool sniffer> start
[admin@MikroTik] tool sniffer> stop
```

Below the sniffed packets will be saved in the file named *test*:

```
[admin@MikroTik] tool sniffer> save file-name=test
[admin@MikroTik] tool sniffer> /file print
  # NAME                              TYPE         SIZE       CREATION-TIME
  0 test                              unknown      1350       apr/07/2003 16:01:52

[admin@MikroTik] tool sniffer>
```

Sniffed Packets

Home menu level: ***/tool sniffer packet***

Description

The submenu allows to see the list of sniffed packets.

Property Description

data (*read-only: text*) - specified data inclusion in packets

dst-address (*read-only: IP address*) - destination IP address

dst-mac-address (*MAC address*) - destination MAC address

fragment-offset (*read-only: integer*) - IP fragment offset

identification (*read-only: integer*) - IP identification

interface (*read-only: name*) - name of the interface the packet has been captured on

ip-header-size (*read-only: integer*) - the size of IP header

ip-packet-size (*read-only: integer*) - the size of IP packet

ip-protocol (*ip* | *icmp* | *igmp* | *ggp* | *ipencap* | *st* | *tcp* | *egp* | *pup* | *udp* | *hmp* | *xns-idp* | *rdp* | *iso-tp4* | *xtp* | *ddp* | *idrp-cmtp* | *gre* | *esp* | *ah* | *rspf* | *vmtp* | *ospf* | *ipip* | *encap*) - the name/number of IP protocol

- **ip** - Internet Protocol
- **icmp** - Internet Control Message Protocol
- **igmp** - Internet Group Management Protocol
- **ggp** - Gateway-Gateway Protocol
- **ipencap** - IP Encapsulated in IP
- **st** - st datagram mode
- **tcp** - Transmission Control Protocol
- **egp** - Exterior Gateway Protocol
- **pup** - Parc Universal packet Protocol
- **udp** - User Datagram Protocol
- **hmp** - Host Monitoring Protocol
- **xns-idp** - Xerox ns idp
- **rdp** - Reliable Datagram Protocol
- **iso-tp4** - ISO Transport Protocol class 4
- **xtp** - Xpress Transfer Protocol
- **ddp** - Datagram Delivery Protocol
- **idpr-cmtp** - idpr Control Message Transport
- **gre** - General Routing Encapsulation
- **esp** - IPsec ESP protocol
- **ah** - IPsec AH protocol
- **rspf** - Radio Shortest Path First
- **vmtp** - Versatile Message Transport Protocol
- **ospf** - Open Shortest Path First
- **ipip** - IP encapsulation (protocol 4)
- **encap** - IP encapsulation (protocol 98)

protocol (*read-only: ip | arp | rarp | ipx | ipv6*) - the name/number of ethernet protocol

- **ip** - Internet Protocol
- **arp** - Address Resolution Protocol
- **rarp** - Reverse Address Resolution Protocol
- **ipx** - Internet Packet exchange protocol
- **ipv6** - Internet Protocol next generation

size (*read-only: integer*) - size of packet

src-address (*IP address*) - source address

src-mac-address (*MAC address*) - source MAC address

time (*read-only: time*) - time when packet arrived

tos (*read-only: integer*) - IP Type Of Service

ttl (*read-only: integer*) - IP Time To Live

Example

In the example below it's seen, how to get the list of sniffed packets:

```
[admin@MikroTik] tool sniffer packet> print
  # TIME    INTERFACE SRC-ADDRESS              DST-ADDRESS            IP-.. SIZE
  0 0.12    ether1    10.0.0.241:1839          10.0.0.181:23 (telnet) tcp   46
  1 0.12    ether1    10.0.0.241:1839          10.0.0.181:23 (telnet) tcp   40
  2 0.12    ether1    10.0.0.181:23 (telnet)   10.0.0.241:1839        tcp   78
  3 0.292   ether1    10.0.0.181               10.0.0.4               gre   88
  4 0.32    ether1    10.0.0.241:1839          10.0.0.181:23 (telnet) tcp   40
  5 0.744   ether1    10.0.0.144:2265          10.0.0.181:22 (ssh)    tcp   76
  6 0.744   ether1    10.0.0.144:2265          10.0.0.181:22 (ssh)    tcp   76
  7 0.744   ether1    10.0.0.181:22 (ssh)      10.0.0.144:2265        tcp   40
  8 0.744   ether1    10.0.0.181:22 (ssh)      10.0.0.144:2265        tcp   76
[admin@MikroTik] tool sniffer packet>
```

Packet Sniffer Protocols

Home menu level: ***/tool sniffer protocol***

Description

In this submenu you can see all kind of protocols that have been sniffed.

Property Description

bytes (*integer*) - total number of data bytes

ip-protocol (*ip | icmp | igmp | ggp | ipencap | st | tcp | egp | pup | udp | hmp | xns-idp | rdp | iso-tp4 | xtp | ddp | idrp-cmtp | gre | esp | ah | rspf | vmtp | ospf | ipip | encap*) - the name/number of IP protocol

- **ip** - Internet Protocol
- **icmp** - Internet Control Message Protocol
- **igmp** - Internet Group Management Protocol
- **ggp** - Gateway-Gateway Protocol

- **ipencap** - IP Encapsulated in IP
- **st** - st datagram mode
- **tcp** - Transmission Control Protocol
- **egp** - Exterior Gateway Protocol
- **pup** - Parc Universal packet Protocol
- **udp** - User Datagram Protocol
- **hmp** - Host Monitoring Protocol
- **xns-idp** - Xerox ns idp
- **rdp** - Reliable Datagram Protocol
- **iso-tp4** - ISO Transport Protocol class 4
- **xtp** - Xpress Transfer Protocol
- **ddp** - Datagram Delivery Protocol
- **idpr-cmtp** - idpr Control Message Transport
- **gre** - General Routing Encapsulation
- **esp** - IPsec ESP protocol
- **ah** - IPsec AH protocol
- **rspf** - Radio Shortest Path First
- **vmtp** - Versatile Message Transport Protocol
- **ospf** - Open Shortest Path First
- **ipip** - IP encapsulation
- **encap** - IP encapsulation

packets (*integer*) - the number of packets

port (*name*) - the port of TCP/UDP protocol

protocol (*read-only: ip | arp | rarp | ipx | ipv6*) - the name/number of ethernet protocol

- **ip** - Internet Protocol
- **arp** - Address Resolution Protocol
- **rarp** - Reverse Address Resolution Protocol
- **ipx** - Internet Packet exchange protocol
- **ipv6** - Internet Protocol next generation

share (*integer*) - specific type of traffic share compared to all traffic in bytes

Example

```
[admin@MikroTik] tool sniffer protocol> print
  # PROTOCOL IP-PR... PORT          PACKETS    BYTES   SHARE
  0 ip                              77         4592    100 %
  1 ip       tcp                    74         4328    94.25 %
  2 ip       gre                    3          264     5.74 %
  3 ip       tcp      22 (ssh)      49         3220    70.12 %
  4 ip       tcp      23 (telnet)   25         1108    24.12 %

[admin@MikroTik] tool sniffer protocol>
```

Packet Sniffer Host

Home menu level: ***/tool sniffer host***

Description

The submenu shows the list of hosts that were participating in data excange you've sniffed.

Property Description

address (*read-only: IP address*) - IP address of the host

peek-rate (*read-only: integerinteger*) - the maximum data-rate received/transmitted

rate (*read-only: integerinteger*) - current data-rate received/transmitted

total (*read-only: integerinteger*) - total packets received/transmitted

Example

In the following example we'll see the list of hosts:

```
[admin@MikroTik] tool sniffer host> print
  # ADDRESS        RATE          PEEK-RATE           TOTAL
  0 10.0.0.4       0bps/0bps     704bps/0bps         264/0
  1 10.0.0.144     0bps/0bps     6.24kbps/12.2kbps   1092/2128
  2 10.0.0.181     0bps/0bps     12.2kbps/6.24kbps   2994/1598
  3 10.0.0.241     0bps/0bps     1.31kbps/4.85kbps   242/866

[admin@MikroTik] tool sniffer host>
```

Packet Sniffer Connections

Home menu level: ***/tool sniffer connection***

Description

Here you can get a list of the connections that have been watched during the sniffing time.

Property Description

active (*read-only: yes* | *no*) - if yes the find active connections

bytes (*read-only: integerinteger*) - bytes in the current connection

dst-address (*read-only: IP address*) - destination address

mss (*read-only: integerinteger*) - Maximum Segment Size

resends (*read-only: integerinteger*) - the number of packets resends in the current connection

src-address (*read-only: IP address*) - source address

Example

The example shows how to get the list of connections:

```
[admin@MikroTik] tool sniffer connection> print
Flags: A - active
  #   SRC-ADDRESS        DST-ADDRESS            BYTES     RESENDS   MSS
  0 A 10.0.0.241:1839    10.0.0.181:23 (telnet) 6/42      60/0      0/0
  1 A 10.0.0.144:2265    10.0.0.181:22 (ssh)    504/252   504/0     0/0

[admin@MikroTik] tool sniffer connection>
```

Sniff MAC Address

You can also see the source and destination MAC Addresses. To do so, at first stop the sniffer if it is running, and select a specific interface:

```
[admin@MikroTik] tool sniffer> stop
[admin@MikroTik] tool sniffer> set interface=bridge1
[admin@MikroTik] tool sniffer> start
[admin@MikroTik] tool sniffer> print
            interface: bridge1
         only-headers: no
         memory-limit: 10
            file-name:
           file-limit: 10
     streaming-enabled: no
      streaming-server: 0.0.0.0
         filter-stream: yes
       filter-protocol: ip-only
       filter-address1: 0.0.0.0/0:0-65535
       filter-address2: 0.0.0.0/0:0-65535
              running: yes
[admin@MikroTik] tool sniffer>
```

Now you have the source and destination MAC Addresses:

```
[admin@MikroTik] tool sniffer packet> print detail
 0 time=0 src-mac-address=00:0C:42:03:02:C7 dst-mac-address=00:30:4F:08:3A:E7
   interface=bridge1 src-address=10.5.8.104:1125
   dst-address=10.1.0.172:3987 (winbox-tls) protocol=ip ip-protocol=tcp
   size=146 ip-packet-size=146 ip-header-size=20 tos=0 identification=5088
   fragment-offset=0 ttl=126

 1 time=0 src-mac-address=00:30:4F:08:3A:E7 dst-mac-address=00:0C:42:03:02:C7
   interface=bridge1 src-address=10.1.0.172:3987 (winbox-tls)
   dst-address=10.5.8.104:1125 protocol=ip ip-protocol=tcp size=253
   ip-packet-size=253 ip-header-size=20 tos=0 identification=41744
   fragment-offset=0 ttl=64

 2 time=0.071 src-mac-address=00:0C:42:03:02:C7
   dst-mac-address=00:30:4F:08:3A:E7 interface=bridge1
   src-address=10.5.8.104:1125 dst-address=10.1.0.172:3987 (winbox-tls)
   protocol=ip ip-protocol=tcp size=40 ip-packet-size=40 ip-header-size=20
   tos=0 identification=5089 fragment-offset=0 ttl=126

 3 time=0.071 src-mac-address=00:30:4F:08:3A:E7
   dst-mac-address=00:0C:42:03:02:C7 interface=bridge1
   src-address=10.1.0.172:3987 (winbox-tls) dst-address=10.5.8.104:1125
   protocol=ip ip-protocol=tcp size=213 ip-packet-size=213 ip-header-size=20
   tos=0 identification=41745 fragment-offset=0 ttl=64

-- [Q quit|D dump|down]
```

Ping

Document revision .NaN (February 5, 2008, 15:52 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Ping uses Internet Control Message Protocol (ICMP) Echo messages to determine if a remote host is active or inactive and to determine the round-trip delay when communicating with it.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/, /tool mac-server ping***
Standards and Technologies: ***ICMP***
Hardware usage: ***Not significant***

Description

Ping sends ICMP echo (ICMP type 8) message to the host and waits for the ICMP echo-reply (ICMP type 0) from that host. The interval between these events is called round trip. If the response (that is called pong) has not come until the end of the interval, we assume it has timed out. The second significant parameter reported is ttl (Time to Live). Is is decremented at each machine in which the packet is processed. The packet will reach its destination only when the ttl is greater than the number of routers between the source and the destination.

The Ping Command

Command name: ***/ping***

Property Description

(*IP addressMAC address*) - IP or MAC address for destination host

count (*integer*; default: **0**) - how many times ICMP packets will be sent

- **0** - Ping continues till [Ctrl]+[C] is pressed

do-not-fragment - if added, packets will not be fragmented

interface (*name*) - ping, using ARP requests on this interface, instead of ICMP requests.

interval (*time*: 10ms..5s; default: **1s**) - delay between messages

size (*integer*: 28..65535; default: **64**) - size of the IP packet (in bytes, including the IP and ICMP headers)

src-address (*IP address*) - Source address for ping

ttl (*integer*: 1..255; default: **255**) - time To Live (TTL) value of the ICMP packet

Notes

If DNS service is configured, it is possible to ping by DNS address. To do it from **Winbox**, you should resolve DNS address first, pressing right mouse button over its address and choosing **Lookup Address**.

You cannot ping with packets larger that the MTU of that interface, so the packet **size** should always be equal or less than MTU. If 'pinging' by MAC address, minimal packet size iz 50 bytes.

Only neighbour MikroTik RouterOS routers with MAC-ping feature enabled can be 'pinged' by MAC address.

Example of ping command

An example of Ping command:

```
/pi 159.148.95.16 count=5 interval=500ms
159.148.95.16 64 byte ping: ttl=59 time=21 ms
159.148.95.16 ping timeout
159.148.95.16 ping timeout
159.148.95.16 ping timeout
159.148.95.16 64 byte ping: ttl=59 time=16 ms
5 packets transmitted, 2 packets received, 60% packet loss
round-trip min/avg/max = 16/18.5/21 ms
[admin@MikroTik] >
```

Resolve IP address:

To resolve IP address from a DNS name, type the command:

```
/ping www.google.lv
```

and press the [Tab] key:

```
[admin@MikroTik] > /ping 66.102.11.104
```

The DNS name **www.google.lv** changed to IP address 66.102.11.104!

'Ping', using arp requests:

To ping a host in our local network, using ARP requests instead of ICMP:

```
/ping 10.5.8.130 interface=local
10.5.8.130 with hw-addr 00:30:4F:14:AB:58 ping time=1 ms
10.5.8.130 with hw-addr 00:30:4F:14:AB:58 ping time=1 ms
10.5.8.130 with hw-addr 00:30:4F:14:AB:58 ping time=1 ms
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 1/1.0/1 ms
[admin@MikroTik] >
```

MAC Ping Server

Home menu level: ***/tool mac-server ping***

Property Description

enabled (yes | no; default: **yes**) - whether MAC pings to this router are allowed

Example

To disable MAC pings:

```
[admin@MikroTik] tool mac-server ping> set enabled=no
[admin@MikroTik] tool mac-server ping> print
    enabled: no
[admin@MikroTik] tool mac-server ping>
```

Torch (Realtime Traffic Monitor)

Document revision 1.9 (January 24, 2008, 15:28 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Realtime traffic monitor may be used to monitor the traffic flow through an interface.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/tool***
Standards and Technologies: ***none***
Hardware usage: ***Not significant***

Description

Realtime Traffic Monitor called also torch is used for monitoring traffic that is going through an interface. You can monitor traffic classified by protocol name, source address, destination address, port. Torch shows the protocols you have chosen and mean transmitted and received data rate for each of them.

The Torch Command

Command name: ***/tool torch***

Property Description

(*name*) - the name of the interface to monitor

dst-address (*IP addressnetmask*) - destination address and network mask to filter the traffic only with such an address, any destination address: 0.0.0.0/0

freeze-frame-interval (*time*) - time in seconds for which the screen output is paused

port (*name* | *integer*) - the name or number of the port

protocol (*any* | *any-ip* | *ddp* | *egp* | *encap* | *ggp* | *gre* | *hmp* | *icmp* | *idpr-cmtp* | *igmp* | *ipencap* | *ipip* | *ipsec-ah* | *ipsec-esp* | *iso-tp4* | *ospf* | *pup* | *rdp* | *rspf* | *st* | *tcp* | *udp* | *vmtp* | *xns-idp* | *xtp*) - the name or number of the protocol

- **any** - any ethernet or IP protocol
- **any-ip** - any IP protocol

src-address (*IP addressnetmask*) - source address and network mask to filter the traffic only with such an address, any source address: 0.0.0.0/0

Notes

If there will be specific port given, then only **tcp** and **udp** protocols will be filtered, i.e., the name of the **protocol** can be **any**, **any-ip**, **tcp**, **udp**.

Except TX and RX, there will be only the field you've specified in command line in the command's output (e.g., you will get **PROTOCOL** column only in case if **protocol** property is explicitly specified).

Example

The following example monitors the traffic that goes through the **ether1** interface generated by **telnet** protocol:

```
[admin@MikroTik] tool> torch ether1 port=telnet
 SRC-PORT                     DST-PORT                     TX         RX
 1439                         23 (telnet)                  1.7kbps    368bps

[admin@MikroTik] tool>
```

To see what IP protocols are going through the **ether1** interface:

```
[admin@MikroTik] tool> torch ether1 protocol=any-ip
 PRO.. TX         RX
 tcp   1.06kbps   608bps
 udp   896bps     3.7kbps
 icmp  480bps     480bps
 ospf  0bps       192bps

[admin@MikroTik] tool>
```

To see what IP protocols are interacting with **10.0.0.144/32** host connected to the **ether1** interface:

```
[admin@MikroTik] tool> torch ether1 src-address=10.0.0.144/32 protocol=any
 PRO.. SRC-ADDRESS     TX         RX
 tcp   10.0.0.144      1.01kbps   608bps
 icmp  10.0.0.144      480bps     480bps

[admin@MikroTik] tool>
```

To see what tcp/udp protocols are going through the **ether1** interface:

```
[admin@MikroTik] tool> torch ether1 protocol=any-ip port=any
 PRO.. SRC-PORT                  DST-PORT                  TX         RX
 tcp   3430                      22 (ssh)                  1.06kbps   608bps
 udp   2812                      1813 (radius-acct)        512bps     2.11kbps
 tcp   1059                      139 (netbios-ssn)         248bps     360bps
[admin@MikroTik] tool>
```

Traceroute

Document revision 1.10 (February 5, 2008, 15:52 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Traceroute determines how packets are being routed to a particular host.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/tool***
Standards and Technologies: ***ICMP***, ***UDP***, ***Traceroute***
Hardware usage: ***Not significant***

Description

Traceroute is a TCP/IP protocol-based utility, which allows user to determine how packets are being routed to a particular host. Traceroute works by increasing the time-to-live value of packets and seeing how far they get until they reach the given destination; thus, a lengthening trail of hosts passed through is built up.

Traceroute shows the number of hops to the given host address of every passed gateway. Traceroute utility sends packets three times to each passed gateway so it shows three timeout values for each gateway in ms.

The Traceroute Command

Command name: ***/tool traceroute***

Property Description

(*IP address*) - IP address of the host you are tracing the route to

dscp (*integer*: 0..63) - DSCP field value for the probe packets (in case the route varies depending on the DSCP priority)

max-hops (*integer*) - utmost hops through which packet can be reached

port (*integer*: 0..65535) - UDP port number

protocol (*UDP* | *ICMP*) - type of protocol to use. If one fails (for example, it is blocked by a firewall), try the other

size (*integer*: 28..1500; default: **64**) - packet size in bytes

src-address (*IP address*) - change the source address of the packet

timeout (*time*: 1s..8s; default: **1s**) - response waiting timeout, i.e. delay between messages

use-dns (*yes* | *no*; default: **no**) - specifies whether to use DNS server, which can be set in /ip dns menu

Notes

Traceroute session may be stopped by pressing [Ctrl]+[C].

Example

To trace the route to 216.239.39.101 host using ICMP protocol with packet size of 64 bytes, setting DSCP field to 8 and extending the timeout to 4 seconds:

```
[admin@MikroTik] tool> traceroute 216.239.39.101 protocol=icmp size=64 dscp=8
timeout=4s
     ADDRESS                                STATUS
   1 159.148.60.227   3ms    3ms     3ms
   2 195.13.173.221  80ms  169ms    14ms
   3 195.13.173.28   6ms    4ms     4ms
   4 195.158.240.21 111ms  110ms   110ms
   5 213.174.71.49 124ms  120ms   129ms
   6 213.174.71.134 139ms  146ms   135ms
   7 213.174.70.245 132ms  131ms   136ms
   8 213.174.70.58 211ms  215ms   215ms
   9 195.158.229.130 225ms  239ms      0s
  10 216.32.223.114 283ms  269ms   281ms
  11 216.32.132.14 267ms  260ms   266ms
  12 209.185.9.102 296ms  296ms   290ms
  13 216.109.66.1 288ms  297ms   294ms
  14 216.109.66.90 297ms  317ms   319ms
  15 216.239.47.66 137ms  136ms   134ms
  16 216.239.47.46 135ms  134ms   134ms
  17 216.239.39.101 134ms  134ms   135ms
[admin@MikroTik] tool>
```

System Watchdog

Document revision 1.3 (February 6, 2008, 4:08 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

System watchdog feature is needed to reboot the system in case of software failures.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/system watchdog***
Hardware usage: ***Not significant***

Hardware Watchdog Management

Home menu level: ***/system watchdog***

Description

This menu allows to configure system when an IP address does not respond, or in case the system has locked up. Software watchdog timer is used to provide the last option, but in very rare cases (caused by hardware malfunction) it can lock up by itself. There is a hardware watchdog device available in RouterBOARD hardware, which can reboot the system in any case.

Property Description

auto-send-supout (yes | no; default: **no**) - after the support output file is automatically generated, it can be sent by email

automatic-supout (yes | no; default: **yes**) - when software failure happens, a file named "autosupout.rif" is generated automatically. The previous "autosupout.rif" file is renamed to "autosupout.old.rif"

no-ping-delay (*time*; default: **5m**) - specifies how long after reboot not to test and ping watch-address. The default setting means that if watch-address is set and is not reachable, the router will reboot about every 6 minutes.

send-email-from (*text*; default: "") - e-mail address to send the support output file from. If not set, the value set in /tool e-mail is used

send-email-to (*text*; default: "") - e-mail address to send the support output file to

send-smtp-server (*text*; default: "") - SMTP server address to send the support output file through. If not set, the value set in /tool e-mail is used

watch-address (*IP address*; default: **none**) - if set, the system will reboot in case 6 sequental pings to the given IP address (sent once per 10 seconds) will fail

- **none** - disable this option

watchdog-timer (yes | no; default: **no**) - whether to reboot if system is unresponsive for a minute

Example

To make system generate a support output file and sent it automatically to **support@example.com** throught the **192.0.2.1**in case of a software crash:

```
[admin@MikroTik] system watchdog> set auto-send-supout=yes \
\... send-to-email=support@example.com send-smtp-server=192.0.2.1
[admin@MikroTik] system watchdog> print
      watch-address: none
     watchdog-timer: yes
      no-ping-delay: 5m
   automatic-supout: yes
   auto-send-supout: yes
   send-smtp-server: 192.0.2.1
      send-email-to: support@example.com
[admin@MikroTik] system watchdog>
```

UPS Monitor

Document revision 2.3 (February 6, 2008, 4:08 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

The UPS monitor feature works with APC UPS units that support “smart” signaling over serial RS232 or USB connection. This feature enables the network administrator to monitor the UPS and set the router to ‘gracefully’ handle any power outage with no corruption or damage to the router. The basic purpose of this feature is to ensure that the router will come back online after an extended power failure. To do this, the router will monitor the UPS and set itself to hibernate mode when the utility power is down and the UPS battery is has less than 10% of its battery power left. The router will then continue to monitor the UPS (while in hibernate mode) and then restart itself after when the utility power returns. If the UPS battery is drained and the router loses all power, the router will power back to full operation when the ‘utility’ power returns.

The UPS monitor feature on the MikroTik RouterOS supports

- hibernate and safe reboot on power and battery failure
- UPS battery test and run time calibration test
- monitoring of all "smart" mode status information supported by UPS
- logging of power changes

Specifications

Packages required: ***ups***
License required: ***level1***
Home menu level: ***/system ups***
Standards and Technologies: ***APC's smart protocol***
Hardware usage: ***Not significant***

Description

Cabling

The serial APC UPS (BackUPS Pro or SmartUPS) requires a special serial cable (unless connected with USB). If no cable came with the UPS, a cable may be ordered from APC or one can be made "in-house". Use the following diagram:

Router Side (DB9f)	Signal	Direction	UPS Side (DB9m)
2	**Receive**	**IN**	**2**
3	**Send**	**OUT**	**1**
5	**Ground**		**4**
7	**CTS**	**IN**	**6**

Note that you may also connect with USB if available.

UPS Monitor Setup

Home menu level: ***/system ups***

Property Description

alarm-setting (*delayed* | *immediate* | *low-battery* | *none*; default: **immediate**) - UPS sound alarm setting:

- **delayed** - alarm is delayed to the on-battery event
- **immediate** - alarm immediately after the on-battery event
- **low-battery** - alarm only when the battery is low
- **none** - do not alarm

load (*read-only: percentage*) - the UPS's output load as a percentage of full rated load in Watts. The typical accuracy of this measurement is ±3% of the maximum of 105%

manufacture-date (*read-only: text*) - the UPS's date of manufacture in the format "mm/dd/yy" (month, day, year)

min-runtime (*time*; default: **5m**) - minimal run time remaining. After a 'utility' failure, the router will monitor the runtime-left value. When the value reaches the min-runtime value, the router will go to hibernate mode

- **0** - the router will go to hibernate mode when the "battery low" signal is sent indicating that the battery power is below 10%

model (*read-only: text*) - less than 32 ASCII character string consisting of the UPS model name (the words on the front of the UPS itself)

nominal-battery-voltage (*read-only: integer*) - the UPS's nominal battery voltage rating (this is not the UPS's actual battery voltage)

offline-after (*read-only: time*) - when will the router go offline

offline-time (*time*; default: **5m**) - how long to work on batteries. The router waits that amount of time and then goes into hibernate mode until the UPS reports that the 'utility' power is back

- **0** - the router will go into hibernate mode according the min-runtime setting and 10% of battery power event. In this case, the router will wait until the UPS reports that the battery power is below 10%

port (*name*) - communication port of the router

serial (*read-only: text*) - a string of at least 8 characters directly representing the UPS's serial number as set at the factory. Newer SmartUPS models have 12-character serial numbers

version (*read-only: text*) - UPS version, consists of three fields: SKU number, firmware revision, country code. The county code may be one of the following:

- **I** - 220/230/240 Vac
- **D** - 115/120 Vac
- **A** - 100 Vac
- **M** - 208 Vac
- **J** - 200 Vac

Notes

In order to enable UPS monitor, the serial port should be available.

Example

To enable the UPS monitor for port **serial1**:

```
[admin@MikroTik] system ups> add port=serial1 disabled=no
[admin@MikroTik] system ups> print
Flags: X - disabled, I - invalid
 0    name="ups" port=serial1 offline-time=5m min-runtime=5m
      alarm-setting=immediate model="SMART-UPS 1000" version="60.11.I"
      serial="QS0030311640" manufacture-date="07/18/00"
      nominal-battery-voltage=24V
[admin@MikroTik] system ups>
```

Runtime Calibration

Command name: ***/system ups rtc***

Description

The **rtc** command causes the UPS to start a run time calibration until less than 25% of full battery capacity is reached. This command calibrates the returned run time value.

Notes

The test begins only if the battery capacity is 100%.

Example

```
[admin@MikroTik] system ups> rtc 0
```

UPS Monitoring

Command name: ***/system ups monitor***

Property Description

battery-charge (*percentage*) - the UPS's remaining battery capacity as a percent of the fully charged condition

battery-voltage - the UPS's present battery voltage. The typical accuracy of this measurement is ±5% of the maximum value (depending on the UPS's nominal battery voltage)

frequency (*percentage*) - when operating on-line, the UPS's internal operating frequency is synchronized to the line within variations within 3 Hz of the nominal 50 or 60 Hz. The typical accuracy of this measurement is ±1% of the full scale value of 63 Hz

line-voltage - the in-line utility power voltage

load (*percentage*) - the UPS's output load as a percentage of full rated load in Watts. The typical accuracy of this measurement is ±3% of the maximum of 105%

low-battery - only shown when the UPS reports this status

on-battery (*yes* | *no*) - Whether UPS battery is supplying power

on-line (*yes* | *no*) - whether power is being provided by the external utility (power company)

output-voltage - the UPS's output voltage

overloaded-output - only shown when the UPS reports this status

replace-battery - only shown when the UPS reports this status

runtime-calibration-running - only shown when the UPS reports this status

runtime-left (*time*) - the UPS's estimated remaining run time in minutes. You can query the UPS when it is operating in the on-line, bypass, or on-battery modes of operation. The UPS's remaining run time reply is based on available battery capacity and output load

smart-boost-mode - only shown when the UPS reports this status

smart-ssdd-mode - only shown when the UPS reports this status

transfer-cause (*text*) - the reason for the most recent transfer to on-battery operation (only shown when the unit is on-battery)

Example

When running on utility power:

```
[admin@MikroTik] system ups> monitor 0
          on-line: yes
       on-battery: no
      RTC-running: no
     runtime-left: 20m
   battery-charge: 100%
  battery-voltage: 27V
     line-voltage: 226V
   output-voltage: 226V
             load: 45%
      temperature: 39C
        frequency: 50Hz
  replace-battery: no
      smart-boost: no
       smart-trim: no
         overload: no
      low-battery: no

[admin@MikroTik] system ups>
```

When running on battery:

```
[admin@MikroTik] system ups> monitor 0
          on-line: no
       on-battery: yes
   transfer-cause: "Line voltage notch or spike"
      RTC-running: no
     runtime-left: 19m
    offline-after: 4m46s
   battery-charge: 94%
  battery-voltage: 24V
     line-voltage: 0V
   output-voltage: 228V
             load: 42%
      temperature: 39C
        frequency: 50Hz
  replace-battery: no
      smart-boost: no
       smart-trim: no
         overload: no
      low-battery: no

      [admin@MikroTik] system ups>
```

VRRP

Document revision 1.6 (February 6, 2008, 4:08 GMT)
This document applies to MikroTik RouterOS V3.0

Table of Contents

General Information

Summary

Virtual Router Redundancy Protocol (VRRP) implementation in the MikroTik RouterOS is RFC2338 compliant. VRRP protocol is used to ensure constant access to some resources. Two or more routers (referred as VRRP Routers in this context) create a highly available cluster (also referred as Virtual routers) with dynamic fail over. Each router can participate in not more than 255 virtual routers per interface. Many modern routers support this protocol.

Network setups with VRRP clusters provide high availability for routers without using clumsy ping-based scripts.

Specifications

Packages required: ***system***
License required: ***level1***
Home menu level: ***/interface vrrp***
Standards and Technologies: ***VRRP***, ***AH***, ***HMAC-MD5-96 within ESP and AH***
Hardware usage: ***Not significant***

Description

Virtual Router Redundancy Protocol is an election protocol that provides high availability for routers. A number of routers may participate in one or more virtual routers. One or more IP addresses may be assigned to a virtual router. A node of a virtual router can be in one of the following states:

- **MASTER** state, when the node answers all the requests to the instance's IP addresses. There may only be one MASTER node in a virtual router. This node sends VRRP advertisement packets to all the backup routers (using multicast address) every once in a while (set in **interval** property).
- **BACKUP** state, when the VRRP router monitors the availability and state of the Master Router. It does not answer any requests to the instance's IP addresses. Should master become unavailable (if at least three sequential VRRP packets are lost), election process happens, and new master is proclaimed based on its priority. For more details on virtual routers, see RFC2338.

Notes

VRRP does not currently work on VLAN interfaces, as it is impossible to have the MAC address of a VLAN interface different from the MAC address of the physical interface it is put on.

VRRP Routers

Home menu level: ***/interface vrrp***

Description

A number of VRRP routers may form a virtual router. The maximal number of clusters on one network is 255 each having a unique VRID (Virtual Router ID). Each router participating in a VRRP cluster must have it priority set to a valid value. Each VRRP instance is configured like a virtual interface that bound to a real interface (in a similar manner VLAN is). VRRP addresses are then put on the virtual VRRP interface normally. The VRRP master has **running** flag enabled, making the address (and the associated routes and other configuration) active. A backup instance is not 'running', so all the settings attached to that interface is inactive.

Property Description

arp (*disabled* | *enabled* | *proxy-arp* | *reply-only*; default: **enabled**) - Address Resolution Protocol

authentication (*none* | *simple* | *ah*; default: **none**) - authentication method to use for VRRP advertisement packets

- **none** - no authentication
- **simple** - plain text authentication
- **ah** - Authentication Header using HMAC-MD5-96 algorithm

backup (*read-only: flag*) - whether the instance is in the backup state

interface (*name*) - interface name the instance is running on

interval (*integer*: 1..255; default: **1**) - VRRP update interval in seconds. Defines how frequently the master of the given cluster sends VRRP advertisement packets

mac-address (*MAC address*) - MAC address of the VRRP instance. According to the RFC, any VRRP instance should have its unique MAC address

master (*read-only: flag*) - whether the instance is in the master state

mtu (*integer*; default: **1500**) - Maximum Transmission Unit

name (*name*) - assigned name of the VRRP instance

on-backup (*name*; default: "") - script to execute when the node switch to backup state

on-master (*name*; default: "") - script to execute when the node switch to master state

password (*text*; default: "") - password required for authentication depending on method used can be ignored (if no authentication used), 8-character long text string (for plain-text authentication) or 16-character long text string (128-bit key required for AH authentication)

preemption-mode (yes | no; default: **yes**) - whether preemption mode is enabled

- **no** - a backup node will not be elected to be a master until the current master fail even if the backup node has higher priority than the current master
- **yes** - the master node always has the priority

priority (*integer*: 1..255; default: **100**) - priority of the current node (higher values mean higher priority)

- **255** - RFC requires that the router that owns the IP addresses assigned to this instance had the priority of 255

vrid (*integer*: 0..255; default: **1**) - Virtual Router Identifier (must be unique on one interface)

Notes

All the nodes of one cluster must have the same **vrid**, **interval**, **preemption-mode**, **authentication** and **password**.

Example

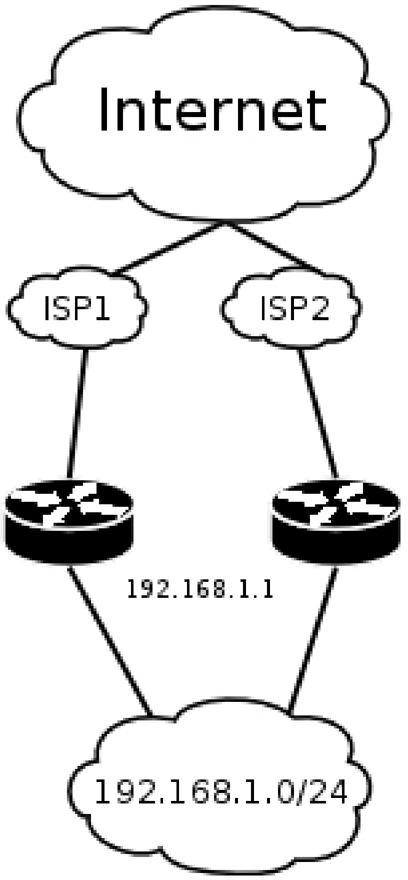
To add a VRRP instance on **ether1** interface, forming (because **priority** is **255**) a virtual router with **vrid** of **1**:

```
[admin@MikroTik] interface vrrp> add interface=ether1 vrid=1 priority=255
[admin@MikroTik] interface vrrp> print
Flags: X - disabled, I - invalid, R - running, M - master, B - backup
 0   RM name="vrrp1" mtu=1500 mac-address=00:00:5E:00:01:01 arp=enabled
        interface=ether1 vrid=1 priority=255 interval=1 preemption-mode=yes
        authentication=none password="" on-backup="" on-master=""
[admin@MikroTik] ip vrrp>
```

Note that the instance is active at once. This is because it has the priority of 255. The instance would wait in backup mode for a new master election process to complete in its favour before assuming the master role otherwise. This also means that there must not be other VRRP routers with the maximal priority

A simple example of VRRP fail over

Description



VRRP protocol may be used to make a redundant Internet connection with seamless fail-over. Let us assume that we have 192.168.1.0/24 network and we need to provide highly available Internet connection for it. This network should be NATted (to make fail-over with public IPs, use such dynamic routing protocols as BGP or OSPF together with VRRP). We have connections to two different Internet Service Providers (ISPs), and one of them is preferred (for example, it is cheaper or faster).

This example shows how to configure VRRP on the two routers shown on the diagram. The routers must have initial configuration: interfaces are enabled, each interface have appropriate IP address (note that each of the two interfaces should have an IP address), routing table is set correctly (it should have at least a default route). SRC-NAT or masquerading should also be configured before. See the respective manual chapters on how to make this configuration.

We will assume that the interface the 192.168.1.0/24 network is connected to is named **local** on both VRRP routers

Configuring Master VRRP router

First of all we should create a VRRP instance on this router. We will use the priority of 255 for this router as it should be preferred router.

```
[admin@MikroTik] interface vrrp> add interface=local priority=255
[admin@MikroTik] interface vrrp> print
Flags: X - disabled, I - invalid, R - running, M - master, B - backup
 0   RM name="vrrp1" mtu=1500 mac-address=00:00:5E:00:01:01 arp=enabled
        interface=local vrid=1 priority=255 interval=1 preemption-mode=yes
        authentication=none password="" on-backup="" on-master=""
[admin@MikroTik] interface vrrp>
```

Next the IP address should be added to this VRRP instance

```
[admin@MikroTik] ip address> add address=192.168.1.1/24 interface=vrrp1
[admin@MikroTik] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
  #   ADDRESS            NETWORK         BROADCAST       INTERFACE
  0   10.0.0.1/24        10.0.0.0        10.0.0.255      public
  1   192.168.1.2/24     192.168.1.0     192.168.1.255   local
  2   192.168.1.1/24     192.168.1.0     192.168.1.255   vrrp1
[admin@MikroTik] ip address>
```

Configuring Backup VRRP router

Now we will create VRRP instance with lower priority (we can use the default value of **100**), so this router will back up the preferred one:

```
[admin@MikroTik] interface vrrp> add interface=local
[admin@MikroTik] ip vrrp> print
Flags: X - disabled, I - invalid, R - running, M - master, B - backup
 0    B name="vrrp1" mtu=1500 mac-address=00:00:5E:00:01:01 arp=enabled
        interface=local vrid=1 priority=100 interval=1 preemption-mode=yes
        authentication=none password="" on-backup="" on-master=""
[admin@MikroTik] interface vrrp>
```

Now we should add the same address as was added to the master node:

```
[admin@MikroTik] ip address> add address=192.168.1.1/24 interface=vrrp1
```

Testing fail over

Now, when we will disconnect the master router, the backup one will switch to the master state after a few seconds:

```
[admin@MikroTik] interface vrrp> print
Flags: X - disabled, I - invalid, R - running, M - master, B - backup
 0   RM name="vrrp1" mtu=1500 mac-address=00:00:5E:00:01:01 arp=enabled
        interface=local vrid=1 priority=100 interval=1 preemption-mode=yes
        authentication=none password="" on-backup="" on-master=""
[admin@MikroTik] interface vrrp>
```