

Terminal Console

Document revision NaN (Tue Apr 20 16:17:53 GMT 2004)

This document applies to V2.8

Table of Contents

[Table of Contents](#)

[Summary](#)

[Specifications](#)

[Related Documents](#)

[Common Console Functions](#)

[Description](#)

[Example](#)

[Lists and Item Names](#)

[Description](#)

[Notes](#)

[Example](#)

[Quick Typing](#)

[Description](#)

[Notes](#)

[Additional Information](#)

[Description](#)

[General Commands](#)

[Description](#)

[Command Description](#)

[Safe Mode](#)

[Description](#)

General Information

Summary

The Terminal Console is used for accessing the MikroTik Router's configuration and management features using text terminals, *id est* remote terminal clients or locally attached monitor and keyboard. The Terminal Console is also used for writing scripts. This manual describes the general console operation principles. Please consult the Scripting Manual on some advanced console commands and on how to write scripts.

Specifications

Packages required: *system*

License required: *level1*

Hardware usage: *Not significant*

Related Documents

- [Scripting Host and Complementary Tools](#)

Common Console Functions

Description

The console allows configuration of the router's settings using text commands. Although the command structure is similar to the Unix shell, you can get additional information about the command structure in the **Scripting Host and Complementary Tools** manual. Since there is a lot of available commands, they are split into groups organized in a way of hierarchical menu levels. The name of a menu level reflects the configuration information accessible in the relevant section, *exempli gratia* **/ip hotspot**.

In general, all menu levels hold the same commands. The difference is expressed mainly in command parameters.

Example

For example, you can issue the **/ip route print** command:

```
[admin@MikroTik] > /ip route print
Flags: X - disabled, I - invalid, D - dynamic, J - rejected,
C - connect, S - static, r - rip, o - ospf, b - bgp
#   DST-ADDRESS      G GATEWAY      DISTANCE INTERFACE
0   S 0.0.0.0/0       r 192.168.2.1  1           WAN
1   DC 192.168.124.0/24 r 0.0.0.0      0           LAN
2   DC 192.168.2.0/24  r 0.0.0.0      0           WAN
3   DC 192.168.0.0/24  r 0.0.0.0      0           LAN

[admin@MikroTik] >
```

Instead of typing **ip route** path before each command, the path can be typed only once to move into this particular branch of menu hierarchy. Thus, the example above could also be executed like this:

```
[admin@MikroTik] > ip route
[admin@MikroTik] ip route> print
Flags: X - disabled, I - invalid, D - dynamic, J - rejected,
C - connect, S - static, r - rip, o - ospf, b - bgp
#   DST-ADDRESS      G GATEWAY      DISTANCE INTERFACE
0   S 0.0.0.0/0       r 192.168.2.1  1           WAN
1   DC 192.168.124.0/24 r 0.0.0.0      0           LAN
2   DC 192.168.2.0/24  r 0.0.0.0      0           WAN
3   DC 192.168.0.0/24  r 0.0.0.0      0           LAN

[admin@MikroTik] ip route>
```

Notice that the prompt changes in order to reflect where you are located in the menu hierarchy at the moment. To move to the top level again, type **/**:

```
[admin@MikroTik] > /ip route
[admin@MikroTik] ip route> /
[admin@MikroTik] >
```

To move up one command level, type **..**:

```
[admin@MikroTik] ip route> ..
[admin@MikroTik] ip>
```

You can also use **/** and **..** to execute commands from other menu levels without changing the current level:

```
[admin@MikroTik] ip route> /ping 10.0.0.1
10.0.0.1 ping timeout
```

```

2 packets transmitted, 0 packets received, 100% packet loss
[admin@MikroTik] ip route> .. firewall print
# NAME                                POLICY
0 input                                accept
1 forward                              accept
2 output                              accept
3 ;;; Limit unauthorized HS clients
  hs-temp                              none
4 ;;; account auth HS clients
  hotspot                              none
[admin@MikroTik] ip route>

```

Lists and Item Names

Description

Lists

Many of the command levels operate with arrays of items: interfaces, routes, users etc. Such arrays are displayed in similarly looking lists. All items in the list have an item number followed by its parameter values.

To change parameters of an item, you have to specify it's number to the set command.

Item Names

Some lists have items that have specific names assigned to each. Examples are **interface** or **user** levels. There you can use item names instead of item numbers.

You do not have to use the **print** command before accessing items by name. As opposed to numbers, names are not assigned by the console internally, but are one of the items' properties. Thus, they would not change on their own. However, there are all kinds of obscure situations possible when several users are changing router's configuration at the same time. Generally, item names are more "stable" than the numbers, and also more informative, so you should prefer them to numbers when writing console scripts.

Notes

Item numbers are assigned by **print** command and are not constant - it is possible that two successive **print** commands will order items differently. But the results of last **print** commands are memorized and thus, once assigned, item numbers can be used even after **add**, **remove** and **move** operations (after **move** operation item numbers are moved with the items). Item numbers are assigned on per session basis, they will remain the same until you quit the console or until the next **print** command is executed. Also, numbers are assigned separately for every item list, so **ip address print** would not change numbers for **interface** list.

Example

```

[admin@MikroTik] interface> set 0 mtu=1200
ERROR: item number must be assigned by a print command
use print command before using an item number in a command
[admin@MikroTik] interface> print
Flags: X - disabled, D - dynamic, R - running

```

```

#      NAME      TYPE      RX-RATE  TX-RATE  MTU
0  R Public      ether      0         0        1500
1  R Local      ether      0         0        1500
2  R wlan1      wlan       0         0        1500
[admin@MikroTik] interface> set 0
disabled mtu name rx-rate tx-rate
[admin@MikroTik] interface> set 0 mtu=1200
[admin@MikroTik] interface> set wlan1 mtu=1300
[admin@MikroTik] interface> print
Flags: X - disabled, D - dynamic, R - running
#      NAME      TYPE      RX-RATE  TX-RATE  MTU
0  R Public      ether      0         0        1200
1  R Local      ether      0         0        1500
2  R wlan1      wlan       0         0        1300
[admin@MikroTik] interface>

```

Quick Typing

Description

There are two features in the console that help entering commands much quicker and easier - the [Tab] key completions, and abbreviations of command names. Completions work similarly to the **bash** shell in UNIX. If you press the [Tab] key after a part of a word, console tries to find the command within the current context that begins with this word. If there is only one match, it is automatically appended, followed by a space:

```
/inte[Tab]_ becomes /interface _
```

If there is more than one match, but they all have a common beginning, which is longer than that what you have typed, then the word is completed to this common part, and no space is appended:

```
/interface set e[Tab]_ becomes /interface set ether_
```

If you've typed just the common part, pressing the tab key once has no effect. However, pressing it for the second time shows all possible completions in compact form:

```

[admin@MikroTik] > interface set e[Tab]_
[admin@MikroTik] > interface set ether[Tab]_
[admin@MikroTik] > interface set ether[Tab]_
ether1 ether5
[admin@MikroTik] > interface set ether_

```

The [Tab] key can be used almost in any context where the console might have a clue about possible values - command names, argument names, arguments that have only several possible values (like names of items in some lists or name of protocol in firewall and NAT rules). You cannot complete numbers, IP addresses and similar values.

Another way to press fewer keys while typing is to abbreviate command and argument names. You can type only beginning of command name, and, if it is not ambiguous, console will accept it as a full name. So typing:

```

[admin@MikroTik] > pi 10.1 c 3 s 100
equals to:
[admin@MikroTik] > ping 10.0.0.1 count 3 size 100

```

Notes

Pressing [Tab] key while entering IP address will do a DNS lookup, instead of completion. If what is typed before cursor is a valid IP address, it will be resolved to a DNS name (reverse resolve), otherwise it will be resolved directly (i.e. to an IP address). To use this feature, DNS server must be configured and working. To avoid input lockups any such lookup will timeout after half a second, so you might have to press [Tab] several times, before the name is actually resolved.

It is possible to complete not only beginning, but also any distinctive substring of a name: if there is no exact match, console starts looking for words that have string being completed as first letters of a multiple word name, or that simply contain letters of this string in the same order. If single such word is found, it is completed at cursor position. For example:

```
[admin@MikroTik] > interface x[TAB]_  
[admin@MikroTik] > interface export _
```

```
[admin@MikroTik] > interface mt[TAB]_  
[admin@MikroTik] > interface monitor-traffic _
```

Additional Information

Description

Built-in Help

The console has a built-in help, which can be accessed by typing ?. General rule is that help shows what you can type in position where the ? was pressed (similarly to pressing [Tab] key twice, but in verbose form and with explanations).

Internal Item Numbers

You can specify multiple items as targets to some commands. Almost everywhere, where you can write the number of item, you can also write a list of numbers:

```
[admin@MikroTik] > interface print  
Flags: X - disabled, D - dynamic, R - running  
#   NAME           TYPE      MTU  
0   R ether1        ether     1500  
1   R ether2        ether     1500  
2   R ether3        ether     1500  
3   R ether4        ether     1500  
[admin@MikroTik] > interface set 0,1,2 mtu=1460  
[admin@MikroTik] > interface print  
Flags: X - disabled, D - dynamic, R - running  
#   NAME           TYPE      MTU  
0   R ether1        ether     1460  
1   R ether2        ether     1460  
2   R ether3        ether     1460  
3   R ether4        ether     1500  
[admin@MikroTik] >
```

General Commands

Description

There are some commands that are common to nearly all menu levels, namely: **print, set, remove, add, find, get, export, enable, disable, comment, move**. These commands have similar behavior throughout different menu levels.

Command Description

print - shows all information that's accessible from particular command level. Thus, /system clock print shows system date and time, /ip route print shows all routes etc. If there's a list of items in current level and they are not read-only, i.e. you can change/remove them (example of read-only item list is /system history, which shows history of executed actions), then print command also assigns numbers that are used by all commands that operate with items in this list. - applicable only to lists of items. The action is performed with all items in this list in the same order in which they are given. - forces the print command to use tabular output form - specifies what parameters to include in printout - forces the print command to use property=value output form - shows the number of items - prints the contents of the specific submenu into a file. This file will be available in the router's ftp - shows the output from the print command for every interval seconds - prints the oid value, which is useful for SNMP - prints the output without paging, to see printed output which does not fit in the screen, use [Shift]+[PgUp] key combination

set - allows you to change values of general parameters or item parameters. The set command has arguments with names corresponding to values you can change. Use ? or double [Tab] to see list of all arguments. If there is a list of items in this command level, then set has one action argument that accepts the number of item (or list of numbers) you wish to set up. This command does not return anything.

add - this command usually has all the same arguments as set, except the action number argument. It adds a new item with values you have specified, usually to the end of list (in places where order is relevant). There are some values that you have to supply (like the interface for a new route), other values are set to defaults unless you explicitly specify them. - Copies an existing item. It takes default values of new item's properties from another item. If you do not want to make exact copy, you can specify new values for some properties. When copying items that have names, you will usually have to give a new name to a copy - add command returns internal number of item it has added - places a new item before an existing item with specified position. Thus, you do not need to use the move command after adding an item to the list - controls disabled/enabled state of the newly added item(-s) - holds the description of a newly created item

remove - removes item(-s) from a list - contains number(-s) or name(-s) of item(-s) to remove.

move - changes the order of items in list where one is relevant. Item numbers after move command are left in a consistent, but hardly intuitive order, so it's better to resync them by using print after each move command. - first argument. Specifies the item(-s) being moved. - second argument. Specifies the item before which to place all items being moved (they are placed at the end of the list if the second argument is omitted).

find - The find command has the same arguments as set, and an additional from argument which works like the from argument with the print command. Plus, find command has flag arguments like disabled, invalid that take values yes or no depending on the value of respective flag. To see all flags and their names, look at the top of print command's output. The find command returns internal numbers of all items that have the same values of arguments as specified.

edit - this command is in every place that has set command, it can be used to edit values of properties, exempli gratia:

```
[admin@ID] ip route> print
```

```

Flags: X - disabled, I - invalid, D - dynamic, J - rejected,
C - connect, S - static, r - rip, o - ospf, b - bgp
#   DST-ADDRESS      G GATEWAY      DISTANCE INTERFACE
0   S 0.0.0.0/0      r 1.1.1.1      1          ether1
1   DC 10.10.11.0/24 r 0.0.0.0      0          ether1
2   DC 10.1.0.0/24   r 0.0.0.0      0          ether2
3   DC 1.1.1.0/24    r 0.0.0.0      0          ether1
[admin@ID] ip route> edit 0 gateway

```

Safe Mode

Description

It is possible to change router configuration in a way that will make it not accessible except from local console. Usually this is done by accident, but there is no way to undo last change when connection to router is already cut. Safe mode can be used to minimize such risk.

Safe mode is entered by pressing **[Ctrl]+[X]**. To quit safe mode, press **[Ctrl]+[X]** again.

```

[admin@MikroTik] ip firewall rule input> [Ctrl]+[X]
[Safe Mode taken]
[admin@MikroTik] ip firewall rule input<SAFE>

```

Message **Safe Mode taken** is displayed and prompt changes to reflect that session is now in safe mode. All configuration changes that are made (also from other login sessions), while router is in safe mode, are automatically undone if safe mode session terminates abnormally. You can see all such changes that will be automatically undone tagged with an **F** flag in system history:

```

[admin@MikroTik] ip firewall rule input<SAFE> add
[admin@MikroTik] ip firewall rule input<SAFE> /system history print
Flags: U - undoable, R - redoable, F - floating-undo
ACTION      BY      POLICY
F rule added admin   write
[admin@MikroTik] ip firewall rule input<SAFE>

```

Now, if telnet connection is cut, then after a while (TCP timeout is **9** minutes) all changes that were made while in safe mode will be undone. Exiting session by **[Ctrl]+[D]** **emphasis>** also undoes all safe mode changes, while **/quit** does not.

If another user tries to enter safe mode, he's given following message:

```

[admin@MikroTik] >
Hijacking Safe Mode from someone - unroll/release/don't take it [u/r/d]:

```

- **[u]** - undoes all safe mode changes, and puts the current session in safe mode.
- **[d]** - leaves everything as-is.
- **[r]** - keeps all current safe mode changes, and puts current session in a safe mode. Previous owner of safe mode is notified about this:

```

[admin@MikroTik] ip firewall rule input
[Safe mode released by another user]

```

If too many changes are made while in safe mode, and there's no room in history to hold them all (currently history keeps up to **100** most recent actions), then session is automatically put out of the safe mode, no changes are automatically undone. Thus, it is best to change configuration in small steps, while in safe mode. Pressing **[Ctrl]+[X]** twice is an easy way to empty safe mode action list.