Application Note

Command Line Interface

Advantech Czech s.r.o., Sokolska 71, 562 04 Usti nad Orlici, Czech Republic
Document No. APP-0127-EN, revised on February 3, 2026.

Used symbols

Danger – Information regarding user safety or potential damage to the router.

Attention – Problems that can arise in specific situations.

Information – Useful tips or information of special interest.

Firmware Version

This manual applies to firmware version **6.6.0 (December 17, 2025)**. Features introduced after this version may not be covered.

Contents

1. Document Introduction **1**

1.1 Document Contents . . . 1
1.2 Command Line Access . . . 2
1.3 Permissions Notes . . . 2

2. Commands **3**

2.1 HW Control Commands . . . 4
2.1.1 cdmaat . . . 4
2.1.2 cdmapwr . . . 5
2.1.3 gsminfo . . . 6
2.1.4 gsmat . . . 7
2.1.5 gsmat2 . . . 7
2.1.6 gsmpwr . . . 8
2.1.7 gsmpwr2 . . . 8
2.1.8 gsmsms . . . 8
2.1.9 hwclock . . . 9
2.1.10 io . . . 10
2.1.11 led . . . 11
2.1.12 lpm . . . 12
2.1.13 mac . . . 13
2.1.14 port1 . . . 13
2.1.15 port2 . . . 14
2.1.16 portd . . . 15
2.1.17 pse . . . 17
2.1.18 reboot . . . 18
2.1.19 report . . . 19
2.1.20 sms . . . 19
2.1.21 status . . . 20
2.1.22 stty . . . 22
2.1.23 tpm2 . . . 23
2.1.24 xbus . . . 24
2.2 File/Directory Management Commands . . . 25
2.2.1 basename . . . 25
2.2.2 cat . . . 25
2.2.3 cd . . . 26
2.2.4 chdir . . . 26
2.2.5 cmp . . . 26
2.2.6 cp . . . 27
2.2.7 cut . . . 28
2.2.8 dd . . . 29
2.2.9 decode . . . 29
2.2.10 dirname . . . 30
2.2.11 find . . . 30
2.2.12 grep . . . 31
2.2.13 gunzip . . . 32
2.2.14 gzip . . . 32

2.2.15 head . . . 33
2.2.16 jq . . . 34
2.2.17 less . . . 35
2.2.18 ln . . . 36
2.2.19 ls . . . 37
2.2.20 mkdir . . . 38
2.2.21 mv . . . 38
2.2.22 pwd . . . 39
2.2.23 readlink . . . 39
2.2.24 realpath . . . 40
2.2.25 rm . . . 40
2.2.26 rmdir . . . 41
2.2.27 sed . . . 41
2.2.28 shred . . . 42
2.2.29 split . . . 43
2.2.30 sync . . . 43
2.2.31 tail . . . 44
2.2.32 tar . . . 45
2.2.33 touch . . . 46
2.2.34 vi . . . 47
2.2.35 wc . . . 47
2.2.36 xxd . . . 48
2.2.37 zcat . . . 49
2.3 System Commands . . . 50
2.3.1 adduser . . . 50
2.3.2 backup . . . 51
2.3.3 chmod . . . 52
2.3.4 chown . . . 52
2.3.5 chpasswd . . . 53
2.3.6 clog . . . 54
2.3.7 date . . . 54
2.3.8 deluser . . . 55
2.3.9 df . . . 56
2.3.10 dmesg . . . 56
2.3.11 doas . . . 57
2.3.12 faillock . . . 58
2.3.13 free . . . 59
2.3.14 fwupdate . . . 60
2.3.15 id . . . 61
2.3.16 kill . . . 62
2.3.17 killall . . . 62
2.3.18 klog . . . 63
2.3.19 logger . . . 63
2.3.20 losetup . . . 64
2.3.21 mount . . . 65
2.3.22 openssl . . . 66
2.3.23 passwd . . . 67
2.3.24 pidof . . . 68
2.3.25 ps . . . 68
2.3.26 restore . . . 69

2.3.27 rlog . . . 70
2.3.28 slog . . . 70
2.3.29 service . . . 70
2.3.30 sudo . . . 70
2.3.31 sysctl . . . 71
2.3.32 times . . . 71
2.3.33 top . . . 72
2.3.34 umask . . . 73
2.3.35 umount . . . 74
2.3.36 umupdate . . . 74
2.4 Network Commands . . . 75
2.4.1 arp . . . 75
2.4.2 brctl . . . 76
2.4.3 bridge . . . 77
2.4.4 conntrack . . . 79
2.4.5 curl . . . 80
2.4.6 dhcrelay . . . 81
2.4.7 ebtables . . . 83
2.4.8 email . . . 85
2.4.9 ether-wake . . . 86
2.4.10 ethtool . . . 87
2.4.11 ftpput . . . 88
2.4.12 ifconfig . . . 89
2.4.13 ip . . . 90
2.4.14 ipcalc . . . 92
2.4.15 iptables . . . 93
2.4.16 iw . . . 94
2.4.17 nc . . . 95
2.4.18 net-snmpinform . . . 96
2.4.19 net-snmptrap . . . 97
2.4.20 netstat . . . 98
2.4.21 ntpdate . . . 98
2.4.22 ping . . . 99
2.4.23 ping6 . . . 100
2.4.24 route . . . 101
2.4.25 scp . . . 102
2.4.26 sipcalc . . . 104
2.4.27 snmpget . . . 105
2.4.28 snmpset . . . 106
2.4.29 snmptrap . . . 107
2.4.30 ssh . . . 108
2.4.31 tc . . . 110
2.4.32 tcpdump . . . 111
2.4.33 telnet . . . 111
2.4.34 traceroute . . . 112
2.4.35 traceroute6 . . . 113
2.4.36 vconfig . . . 114
2.4.37 wget . . . 115
2.5 Scripting/Shell Commands . . . 116
2.5.1 awk . . . 116

2.5.2 break . . . 117
2.5.3 continue . . . 117
2.5.4 echo . . . 117
2.5.5 eval . . . 118
2.5.6 exec . . . 118
2.5.7 exit . . . 118
2.5.8 export . . . 118
2.5.9 hash . . . 119
2.5.10 inc . . . 120
2.5.11 let . . . 121
2.5.12 local . . . 122
2.5.13 nohup . . . 122
2.5.14 printf . . . 123
2.5.15 read . . . 124
2.5.16 readonly . . . 124
2.5.17 return . . . 124
2.5.18 shlock . . . 124
2.5.19 set . . . 125
2.5.20 shift . . . 125
2.5.21 sleep . . . 125
2.5.22 source . . . 126
2.5.23 test . . . 127
2.5.24 trap . . . 128
2.5.25 type . . . 129
2.5.26 unset . . . 129
2.5.27 wait . . . 129
2.5.28 xargs . . . 129

3. Related Documents **130**

Appendix: Command Index **131**

List of Figures

List of Tables

1 Description of GSM information . . . 6
2 gsmat options . . . 7
3 gsmat2 options . . . 7
4 hwclock options . . . 9
5 io options . . . 10
6 led options . . . 11
7 led commands . . . 11
8 lpm options . . . 12
9 port1 options . . . 13
10 port2 options . . . 14

11 traceroute options . . . 16
12 port options . . . 17
13 command options . . . 17
14 reboot options . . . 18
15 report options . . . 19
16 sms arguments . . . 19
17 status options . . . 21
18 stty options . . . 22
19 tpm2 subcommands . . . 24
20 xbus commands . . . 24
21 cat options . . . 25
22 cd options . . . 26
23 cmp options . . . 26
24 cp options . . . 27
25 cut options . . . 28
26 dd options . . . 29
27 find expressions . . . 30
28 grep options . . . 31
29 gunzip options . . . 32
30 gzip options . . . 32
31 head options . . . 33
32 jq options . . . 34
33 less options . . . 35
34 ln options . . . 36
35 ls options . . . 37
36 mkdir options . . . 38
37 mv options . . . 38
38 readlink options . . . 39
39 rm options . . . 40
40 sed options . . . 41
41 shred options . . . 42
42 split options . . . 43
43 sync options . . . 43
44 tail options . . . 44
45 tar options . . . 45
46 touch options . . . 46
47 vi options . . . 47
48 wc options . . . 47
49 xxd options . . . 48
50 adduser options . . . 50
51 backup options . . . 51
52 chmod options . . . 52
53 chown options . . . 52
54 chpasswd options . . . 53
55 date options . . . 54
56 deluser options . . . 55
57 df options . . . 56
58 dmesg options . . . 56
59 doas options . . . 57
60 faillock options . . . 58

61 fwupdate options . . . 60
62 id options . . . 61
63 kill options . . . 62
64 killall options . . . 62
65 logger options . . . 63
66 losetup options . . . 64
67 mount flags . . . 65
68 mount options . . . 65
69 passwd options . . . 67
70 pidof options . . . 68
71 restore options . . . 69
72 slog options . . . 70
73 sysctl options . . . 71
74 touch options . . . 72
75 touch keys . . . 72
76 umask options . . . 73
77 umount options . . . 74
78 umupdate options . . . 74
79 arp options . . . 75
80 brctl commands . . . 76
81 Global bridge options . . . 77
82 bridge link commands . . . 77
83 bridge vlan commands . . . 78
84 conntrack comands . . . 79
85 conntrack tables . . . 79
86 conntrack options . . . 79
87 expectation options . . . 80
88 conntrack and expectation options . . . 80
89 dhcrelay options available in DHCPv4 mode only . . . 81
90 dhcrelay options available for both DHCPv4 and DHCPv6 . . . 81
91 dhcrelay options available in DHCPv6 mode only: . . . 82
92 ebtables commands . . . 83
93 ebtables options . . . 84
94 email options . . . 85
95 ether-wake options . . . 86
96 ftpput options . . . 88
97 ifconfig options . . . 89
98 ip options . . . 90
99 ip objects . . . 90
100 ipcalc options . . . 92
101 iw options . . . 94
102 iw commands . . . 94
103 nc options . . . 95
104 net-snmpinform options . . . 96
105 net-snmptrap options . . . 97
106 netstat options . . . 98
107 ntpdate options . . . 98
108 ping options . . . 99
109 ping6 options . . . 100
110 route options . . . 101

111 scp options . . . 102
112 sipcalc – global options . . . 104
113 sipcalc – IPv4 options . . . 104
114 sipcalc – IPv6 options . . . 104
115 snmpget options . . . 105
116 snmpset options . . . 106
117 Type options . . . 106
118 snmptrap options . . . 107
119 ssh options . . . 108
120 tc options . . . 110
121 traceroute options . . . 112
122 traceroute6 options . . . 113
123 vconfig commands and options . . . 114
124 wget options . . . 115
125 awk options . . . 116
126 echo options . . . 117
127 hash options . . . 119
128 nohup options . . . 122
129 printf format specifiers . . . 123
130 Commonly used tests . . . 127
131 Common signals . . . 128
132 xargs options . . . 129

1. Document Introduction

1.1 Document Contents

This document provides a complete list of available **console commands** for Advantech routers. Commands are grouped into the following categories based on their usage:

- **HW Control Commands** — see Chapter 2.1.
- **File/Directory Management Commands** — see Chapter 2.2.
- **System Commands** — see Chapter 2.3.
- **Network Commands** — see Chapter 2.4.
- **Scripting/Shell Commands** — see Chapter 2.4.

You may find some commands available on the system that are not documented in this manual. These commands are not intended for regular user operations, and their incorrect use may cause system malfunction.

For a comprehensive list of all commands, please refer to Appendix *Command Index*.

1.2 Command Line Access

The command-line interface (CLI) is a non-GUI alternative that allows you to manage the router. It provides an interactive interface enabling you to perform advanced configurations and troubleshoot the device directly.

SSH Connection

You can use an SSH (Secure Shell) connection to access the router's console. A commonly used application for this is *PuTTY*. To connect securely without requiring credentials, you can use Passwordless Console Login with a public key. For detailed instructions, refer to the Configuration Manual, Chapter *Administration* → *Modify User* → *Passwordless Console Login*.

Note: Ensure that SSH is enabled under *Configuration* → *Services* → *SSH*.

Web Terminal Router App

The *Web Terminal* Router App allows you to access the router's console directly from the web GUI. This Router App is available for free on the *Engineering Portal*.

1.3 Permissions Notes

Please note that only a user with the `admin` role is permitted to access the router's console; a user with the `user` role does not have this access.

Commands that modify device status or configuration require privileged mode. When you log in to the console on the `flexible` platform, you are already in privileged mode, indicated by the # symbol on the prompt line.

2. Commands

In the vast and complex ecosystem of Unix-like operating systems, commands serve as the essential tools through which users interact with the system, automate tasks, manage resources, and configure services. Whether you are a system administrator, a network engineer, or a software developer, understanding and mastering these commands is crucial to effectively harnessing the full potential of the system.

This chapter delves into an array of commands integral to daily operations and specialized tasks alike. From scripting and shell command essentials that form the backbone of system automation and task scheduling, to administration commands that ensure system security and integrity, each command is a piece of the larger puzzle of system management. Network commands provide the keys to configuring, analyzing, and securing network communications, while router management commands focus on the specifics of networking device configuration. Additionally, file and directory management commands offer the means to navigate and manipulate the filesystem, ensuring data organization and accessibility. Lastly, the text processing and analysis commands section reveals the power of Unix-like systems in handling textual data, enabling the user to edit, search, and process text in various complex ways.

Structured to cater to both novice users seeking foundational knowledge and advanced users aiming to refine their expertise, this chapter aims to equip you with the understanding and practical know-how needed to navigate the intricacies of Unix-like operating systems with confidence. Through concise descriptions, usage examples, and categorized presentation, we invite you on a journey to mastery over your system's capabilities, one command at a time.

2.1 HW Control Commands

These commands are specific to router and network device management, offering functionalities to configure interfaces, manage routing tables, and control device-specific features.

2.1.1 cdmaat

This command is not supported by routers of *v1* production line.

This command interfaces with a CDMA module to send AT commands, facilitating direct communication and control over the module's functionality. This utility is especially useful for developers and administrators working with CDMA technology for diagnostics, configuration, or testing purposes.

Synopsis:

`cdmaat <AT command>`

Description:

`cdmaat` sends specified AT commands to the connected CDMA module, allowing for a wide range of actions, from querying the module status to configuring its settings. The command's ability to interact directly with the CDMA module makes it a powerful tool for advanced device management.

Examples:

Check the signal quality of the CDMA module:

`cdmaat AT+CSQ`

Reset the CDMA module:

`cdmaat ATZ`

2.1.2 cdmapwr

This command is not supported by routers of *v1* production line.

This command controls the power supply to the CDMA module, enabling or disabling it as required. This command is crucial for managing the module's power state, especially for conserving energy or resetting the module.

Synopsis:

```
cdmapwr [on | off]
```

Description:

By specifying `on` or `off` , the `cdmapwr` command toggles the power state of the CDMA module. Turning the module off can be particularly useful for saving battery life in portable devices or when the module is not in use. Conversely, powering on the module restores its operational state, ready for communication or configuration.

Examples:

Power on the CDMA module:

```
cdmapwr on
```

Power off the CDMA module:

```
cdmapwr off
```

2.1.3 gsminfo

This command is designed to retrieve and display detailed information about the GSM module's signal quality and connection status. This tool is invaluable for diagnosing connectivity issues, optimizing signal reception, and ensuring reliable GSM network services.

Synopsis:

`gsminfo`

Description:

`gsminfo` provides a concise overview of the current GSM module's status, including the signal strength, network operator, and the communication channel. This information aids in assessing the quality of the GSM connection and troubleshooting network problems.

Options:

Option	Description
`PLMN`	Displays the code of the operator to which the GSM module is currently connected.
`Cell`	Shows the identifier of the cell to which the router is currently connected.
`Channel`	Indicates the channel number used for communication by the GSM module.
`Level`	Provides an assessment of the signal quality from the connected cell.
`Neighbours`	Lists signal quality information from neighboring cells that are within hearing range.
`Uptime`	Reports the duration since the PPP (Point-to-Point Protocol) connection was established.

Table 1: Description of GSM information

Examples:

Check the signal quality of the GSM module:

`gsminfo`

This example would display various GSM module information including signal quality, operator code, and more, directly in the console output.

2.1.4 gsmat

This program can be used to send an AT command to the cellular module.

Synopsis:

```
gsmat [-t <timeout>] <AT command>
```

Options:

Option	Description
-t	The timeout for the response from the cellular module. If not specified, the default value is 10 seconds.

Table 2: gsmat options

Examples:

Determine the type and firmware version of GSM module.

```
gsmat ATI
```

Determine the IMEI code of module.

```
gsmat AT+GSN
```

2.1.5 gsmat2

This command is not supported by routers of *v1* production line.

This program can be used to send an AT command to the second cellular module if the module is installed in the router.

Synopsis:

```
gsmat2 [-t <timeout>] <AT command>
```

Options:

Option	Description
-t	The timeout for the response from the cellular module. If not specified, the default value is 10 seconds.

Table 3: gsmat2 options

Examples:

See the `gsmat` command.

2.1.6 gsmpwr

This command is not supported by routers of *v1* production line.

This program can be used to control the power supply of the cellular module.

Synopsis:

```
gsmpwr [on | off | shutdown]
```

Examples:

Turn on the power for the cellular module.

```
gsmpwr on
```

Turn off the power for the cellular module.

```
gsmpwr off
```

Shutdown the cellular module by an AT command and turn off the power for it. Please note that execution of this command may take a few seconds.

```
gsmpwr shutdown
```

It is highly recommended to use the *gsmpwr shutdown* command prior to use the *gsmpwr off* command to increase the lifetime of the cellular module.

2.1.7 gsmpwr2

This command is not supported by routers of *v1* production line.

This program can be used to control the power supply of the second cellular module in case this module is installed. For usage examples see the gsmpwr command.

Synopsis:

```
gsmpwr2 [on | off | shutdown]
```

It is highly recommended to use the *gsmpwr2 shutdown* command prior to use the *gsmpwr2 off* command to increase the lifetime of the cellular module.

2.1.8 gsmsms

This program has been deprecated and was removed starting with firmware version 6.6.x. Please use the `sms` command instead, as described in Chapter 2.1.20.

2.1.9 hwclock

This program can be used to query and set the hardware clock (RTC).

Synopsis:

```
hwclock [-r] [-s] [-w] [-u] [-l]
```

Options:

Option	Description
-r	Read hardware clock a print result
-s	Set the System Time from the Hardware Clock
-w	Set the Hardware Clock to the current System Time
-u	The hardware clock is kept in coordinated universal time
-l	The hardware clock is kept in local time

Table 4: hwclock options

Examples:

Set the hardware clock to the current system time.

```
hwclock -w -u
```

2.1.10 io

This program can be used to read binary inputs and to control binary outputs of the router. If installed, it also supports an expansion ports of the router.

Synopsis:

```
io [get <pin>] | [set <pin> <value>]
```

Options:

Option	Description
get	Get the state of input
set	Set the state of output

Table 5: io options

Examples:

Get the state of digital input BIN0.

```
io get bin0
```

Get the state of analog input AN1 on expansion port XC-CNT.

```
io get an1
```

Get the state of counter input CNT1 on expansion port XC-CNT.

```
io get cnt1
```

Set the state of binary output OUT0 to 1.

```
io set out0 1
```

2.1.11 led

This command is not supported by routers of *v1* production line.

This program can be used to control the USR or PWR LED of the router.

Synopsis:

```
led [-p] [-u] <command>
```

Options:

Option	Description
`-p`	Control of PWR LED
`-u`	Control of USR LED (default)

Table 6: led options

Commands:

Command	Description
`on`	Power on the LEDn
`off`	Power off the LED
`slow`	Start blinking the LED slowly
`fast`	Start blinking the LED fast

Table 7: led commands

Examples:

Turn on the USR LED.

```
led on
```

Start blinking slowly with the USR LED.

```
led slow
```

2.1.12 lpm

This command is not supported by routers of *v1*, v2, ICR-2000, ICR-2400, ICR-2500, and ICR-2600 production lines.

This command switches the router into Low Power Mode (LPM) to minimize power consumption. The router can be woken from this state by one of two events: the expiration of a specified time interval or a signal change on a binary input. If both wake-up conditions are configured, the router will wake up as soon as the first event occurs.

Synopsis:

```
lpm [-b] [-i <interval>]
```

Options:

Option	Description
`-b`	Wakes up the router upon activation of a binary input. The specific input used depends on the router platform: • *BIN1* for *SmartFlex*, *SmartMotion*, *ICR-2800*, *ICR-4200*, and *ICR-4400* platforms. • *BIN0* for *SmartStart* and *ICR-3200* platforms. **Note:** This option is not supported by routers of the *ICR-2700* production line.
`-i`	Wakes up the router after the specified time interval has elapsed. The interval is defined in seconds, ranging from 1 to 16,777,215.

Table 8: lpm options

Examples:

Put the router to sleep for five minutes.

```
lpm -i 300
```

Put the router to sleep, to be woken up by the activation of the binary input.

```
lpm -b
```

Put the router to sleep for a maximum of five minutes. It will wake up sooner if the binary input is activated.

```
lpm -b -i 300
```

2.1.13 mac

This program can be used to display the MAC address of eth0.

Synopsis:

```
mac [<separator>]
```

Examples:

Display the MAC address of eth0. Will be used as the separator character "-" instead of ":".

```
mac -
```

2.1.14 port1

This program can be used to control the first expansion port.

Synopsis:

```
port1 [on|off|auto|rs232|rs485]
```

Options:

Option	Description
`on`	Turn on the first expansion port.
`off`	Turn off the first expansion port.
`auto`	Turn on the first expansion port and set the flow control (CTS signal) to RS232 or RS485 mode depending on the type of the expansion board.
`rs232`	Turn on the first expansion port and set the flow control (CTS signal) to RS232 mode.
`rs485`	Turn on the first expansion port and set the flow control (CTS signal) to RS485 mode.

Table 9: port1 options

2.1.15 port2

This program can be used to control the second expansion port.

Synopsis:

```
port2 [on|off|auto|rs232|rs485]
```

Options:

Option	Description
`on`	Turn on the second expansion port.
`off`	Turn off the second expansion port.
`auto`	Turn on the second expansion port and set the flow control (CTS signal) to RS232 or RS485 mode depending on the type of the expansion board.
`rs232`	Turn on the second expansion port and set the flow control (CTS signal) to RS232 mode.
`rs485`	Turn on the second expansion port and set the flow control (CTS signal) to RS485 mode.

Table 10: port2 options

2.1.16 portd

This program facilitates the transparent transfer of data between a serial line and a network using TCP or UDP protocols. It can operate in two primary modes: as a server that waits for incoming connections, or as a client that connects to a specified host.

Synopsis:

```
portd -c <device> [-b <baudrate>] [-d <databits>] [-p <parity>] [-s <stopbits>]
[-l <split timeout>] [-4] [-h <hostname>] [-o <proto>] -t <port> [-k <keepalive time>]
[-i <keepalive interval>] [-j <inactivity timeout>] [-r <keepalive probes>] [-u <user>]
[-n] [-x] [-z] [-f]
```

Options:

Item	Description
`-c <device>`	Specifies the serial line device path, e.g., `/dev/ttyS0` . This option is mandatory.
`-b <baudrate>`	Sets the communication speed in bits per second (baud rate). Default is 9600.
`-d <databits>`	Sets the number of data bits. Valid values are 5, 6, 7, or 8. Default is 8.
`-p <parity>`	Sets the parity. Use 'N' for None, 'E' for Even, or 'O' for Odd. Default is None.
`-s <stopbits>`	Sets the number of stop bits. Valid values are 1 or 2. Default is 1.
`-l <split timeout>`	Sets the timeout in milliseconds for data packetization. If the time between consecutive characters received from the serial port exceeds this value, the buffered data is sent as a single network packet. Default is 20 ms.
`-4`	Forces RS485 mode. This typically involves managing the RTS signal for controlling the line driver.
`-h <hostname>`	Specifies the hostname or IP address of the remote server to connect to. When this option is used, `portd` operates in client mode. If omitted, it runs in server mode, listening for incoming connections.
`-o <proto>`	Sets the communication protocol. Specify `tcp` or `udp` . Default is TCP.
`-t <port>`	Specifies the TCP or UDP port number to listen on (server mode) or connect to (client mode). This option is mandatory.
`-k <time>`	Sets the TCP keepalive time in seconds. This is the duration of inactivity before the first keepalive probe is sent. A value of 0 disables keepalive.
`-i <interval>`	Sets the interval in seconds between individual TCP keepalive probes. Default is 10 seconds.
`-j <timeout>`	Sets an inactivity timeout in seconds. If no data is transferred over the network socket for this duration, the connection is closed. A value of 0 disables this feature.
`-n`	Rejects new connections in server mode. If a client is already connected, any new incoming connection attempts will be rejected until the current one is closed.
`-r <probes>`	Sets the number of unacknowledged TCP keepalive probes to send before considering the connection to be dead. Default is 5 probes.
`-u <user>`	Specifies a user to drop privileges to after the program has been initialized (e.g., after binding to a privileged port).

Continued on the next page

Continued from previous page

Item	Description
`-x`	Use Carrier Detect (CD) signal to indicate TCP connection status. The router's DTR pin will be asserted (high) when a TCP connection is active and de-asserted (low) when disconnected. This can be used to signal connection status to an attached device.
`-z`	Use Data Terminal Ready (DTR) signal to control the TCP connection. The program monitors the serial port's CD pin, and if it is de-asserted by the connected device, the TCP connection is terminated.
`-f`	Enables hardware flow control (RTS/CTS). The RTS pin is asserted when a network connection is active.

Table 11: traceroute options

Examples:

Run a TCP server listening on port 1000. After a client establishes a TCP connection, the program will transparently transfer data from the serial port `/dev/ttyS0` , which is configured for 115200 bit/s, 8 data bits, no parity, and 1 stop bit (8N1). The process runs in the background.

```
portd -c /dev/ttyS0 -b 115200 -t 1000 &
```

Run as a TCP client connecting to a server at IP address 192.168.1.100 on port 2000. Data from the local serial port `/dev/ttyS1` (at 9600, 8N1) will be forwarded to the remote server. The connection will have TCP keepalive enabled with a 60-second idle time.

```
portd -c /dev/ttyS1 -b 9600 -h 192.168.1.100 -t 2000 -k 60 &
```

2.1.17 pse

This program is only supported on models supporting the PoE PSE functionality.

This program can be used to enable/disable the PoE PSE functionality on a router.

Synopsis:

```
pse [port] [command]
```

port:

port	Description
`eth0`	Ethernet ETH0 interface (port).
`eth1`	Ethernet ETH1 interface (port), if equipped on the router.
`lanx`	Port of an Ethernet switch interface, if equipped on the router. Replace `x` with a port number.

Table 12: port options

command:

comm	Description
`on`	Enable the PoE PSE a on specified interface (port).
`off`	Disable the PoE PSE a on specified interface (port).

Table 13: command options

Example:

Enable the PoE PSE on *eth1* Ethernet interface.

```
pse eth1 on
```

Enable the PoE PSE on *lan2* port of Ethernet switch interface.

```
pse lan2 on
```

2.1.18 reboot

This program can be used to reboot the router.

Synopsis:

```
reboot [-d <delay>] [-n <nosync>] [-f <force>]
```

Options:

Option	Description
-d	Delay interval for rebooting
-n	No call to sync()
-f	Force reboot, do not call shutdown

Table 14: reboot options

Examples:

Reboot router after 10 second.

```
reboot -d 10
```

2.1.19 report

This command can be used to list the router report from the command line.

Sensitive data from the report are filtered out for security reasons.

Synopsis:

```
report [<options>]
```

Options:

Option	Description
no option	Report all sections except the configuration one.
`-a`	Report all sections.
`-s`	Report status section.
`-m`	Report router apps section.
`-l`	Report log section.
`-c`	Report configuration section.

Table 15: report options

2.1.20 sms

This command is used to send an SMS message from the router's cellular module.

Synopsis:

```
sms <phone_number> "<text>"
```

Arguments:

Argument	Description
`<phone_number>`	The recipient's telephone number. It is recommended to use the international format (e.g., `+420123456789`).
`<text>`	The content of the SMS message. The text must be enclosed in double quotes if it contains spaces or special characters.

Table 16: sms arguments

Example:

Send the SMS "Hello world" to the phone number +420123456789.

```
sms +420123456789 "Hello world"
```

2.1.21 status

This command displays the status of the router's interfaces and system components. It provides information equivalent to what is available on the *General Status* and *Mobile WAN Status* pages in the router's web administration interface.

Synopsis:

```
status [-h] [-v] [eth | geoloc | gnss | vlan | mobile | module | mwan | sim | bard
| ports | security | sys | hw | wifi ap | wifi sta]
```

Options:

Item	Description
`-h`	Generates HTML output, which is used when the command is called by the web interface.
`-v`	Enables verbose mode, which provides more detailed information. Data amounts are shown in bytes only, not in dynamic units (e.g., KB, MB).
`eth`	Displays the status of Ethernet interfaces (e.g., `eth0`, `eth1`), including link state, speed, and data statistics.
`geoloc`	Displays the router's last known geographical coordinates (latitude and longitude), determined by the GNSS receiver.
`gnss`	Shows the current status of the GNSS receiver, including the UTC time, fix type (e.g., 2D, 3D), dilution of precision (HDOP), and the number of satellites in use versus in view.
`vlan`	Lists the status of configured Virtual LAN (VLAN) interfaces, including their names, associated physical interfaces, and VLAN IDs.
`mobile`	Shows the status of the mobile radio connection, including registration status, operator, network technology (e.g., LTE, 5G), and signal quality metrics.
`module`	Displays the status of the cellular module(s), which can be `module 1`, `module 2`, etc., if available.
`mwan`	Provides the status of the Mobile WAN network interface, including IP address, DNS servers, and connection uptime.
`status sim` or `status sim 1`	Reports daily statistics for the first SIM card, including data usage (RX/TX), connection count, signal history, cell changes, and network availability. `status sim` is an alias for `status sim 1`.
`status sim 2`	Reports the same daily statistics as above, but for the second SIM card.
`status sim 1 full` or `status sim 2 full`	Provides a comprehensive statistical report for the specified SIM card (1 or 2). This extended view includes data for multiple time intervals: Today, Yesterday, This Week, Last Week, This Period, and Last Period.
`bard`	Displays the status of the Backup and Recovery Daemon (BARD), indicating whether it is active and monitoring backup routes.
`ports`	Shows the status of available peripheral ports (e.g., serial ports, USB).
`security`	Shows recent user login activity, including the last successful login, the count of failed login attempts, and the source IP address for each event.
`sys`	Provides general system information, such as uptime, memory usage, and supply voltage.

Continued on the next page

Continued from previous page

Item	Description
`hw`	Displays key hardware information, such as the presence of GNSS or Wi-Fi modules and the product's designated region.
`wifi ap`	Displays the status of the WiFi Access Point.
`wifi sta`	Displays the status of the WiFi Station (client) connection.

Table 17: status options

Example:

Show verbose status of the mobile connection:

```
~ # status -v mobile
Registration        : Home Network
Operator            : Vodafone CZ
Technology          : LTE
PLMN                : 23003
Cell                : 10A804
TAC                 : 947C
Channel             : 1849
Band                : B3
Signal Strength     : -90 dBm
Signal Quality      : -7 dB

RSSI                : -63 dBm
RSRP                : -90 dBm
RSRQ                : -7 dB
SINR                : 18 dB
CSQ                 : 11
```

2.1.22 stty

This program can be used to print or to change terminal characteristics.

Synopsis:

```
stty [-a|g] [-F DEVICE] [SETTING]...
```

Options:

Option	Description
`-F DEVICE`	Open device instead of stdin
`-a`	Print all current settings in human-readable form
`-g`	Print in stty-readable form
`[SETTING]`	See manpage

Table 18: stty options

Examples:

To get current parameters of the first UART serial port.

```
stty -F /dev/ttyS0
```

To only get actual speed of the second UART serial port.

```
stty -F /dev/ttyS1 speed
```

To set parameters of the first UART serial port to:

- speed to 1200 bps
- character size to 7 bits
- 2 stop bits
- disable software output flow control
- reset parameters to system default raw mode

```
stty -F /dev/ttyS0 1200 cs7 cstopb -ixon raw
```

2.1.23 tpm2

TPM features are available only on products that are equipped with a TPM module. TPM support can be verified either in the Hardware Manual or directly in the router console, as described below.

This program can be used to work with the TPM 2.0 (Trusted Platform Module) chip mounted directly on the router mainboard.

Note: To verify whether TPM functionality is supported on your device, use one of the following procedures:

1. In the router console, execute the following command: `ls /dev/tpm0`
   - If the response is `No such file or directory`, the TPM chip is not available.
   - If the response is `/dev/tpm0`, the TPM chip is available.
2. In the router console, execute the TPM command to display fixed TPM properties: `tpm2 getcap properties-fixed`
   - If multiple error messages are displayed, the TPM chip is not available.
   - If a list of TPM properties is displayed, the TPM chip is available.

Synopsis:

`tpm2 <subcommand> [<options>...]`

The table below lists the most important `tpm2` subcommands. For more details about the subcommands, see https://github.com/tpm2-software/tpm2-tools/tree/5.1.X/man.

tpm2 subcommands

Subcommand	Description
`getcap`	Display TPM capabilities in a human readable form.
`create`	Create a child object.
`createek`	Generate TCG profile compliant endorsement key.
`createak`	Generate attestation key with given algorithm under the endorsement hierarchy.
`createprimary`	Create a primary key.
`load`	Load an object into the TPM.
`loadexternal`	Load an external object into the TPM.
`readpublic`	Read the public area of a loaded object.
`evictcontrol`	Make a transient object persistent or evict a persistent object.
`clear`	Clear lockout, endorsement and owner hierarchy authorization values.
`getrandom`	Retrieve random bytes from the TPM.
`hash`	Perform a hash operation with the TPM.
`hmac`	Perform an HMAC operation with the TPM.
`sign`	Sign a hash or message using the TPM.
`verifysignature`	Validate a signature using the TPM.
`encryptdecrypt`	Perform symmetric encryption or decryption.

Continued on the next page

Continued from previous page

Subcommand	Description
`ecdhzgen`	Recover the shared secret value (Z) from a public point and a specified private key.
`nvdefine`	Define a TPM Non-Volatile (NV) index.
`nvundefine`	Delete a Non-Volatile (NV) index.
`nvread`	Read the data stored in a Non-Volatile (NV)s index.
`nvwrite`	Write data to a Non-Volatile (NV) index.

Table 19: tpm2 subcommands

2.1.24 xbus

This program is primarily used for an internal process communication. It can be used as a watchdog of a user process.

Synopsis:

```
xbus [command]
```

xbus commands:

Command	Description
`subscribe <topic> [<script>]`	Subscribe to particular topic.
`publish <topic> <payload>`	Publish the payload for the topic[1].
`write <topic> <payload>`	Publish and store the payload for the topic[1].
`read <topic>`	Read stored payload of the topic.
`list`	List all the stored topics.

Table 20: xbus commands

Examples:

Set the watchdog for process "user_process" to 10 seconds. If this command is not re-executed within 10 seconds, the router will reboot.

```
xbus write watchdog/proc/user_process "Timeout:10"
```

Suspend the watchdog for process "user_process".

```
xbus write watchdog/proc/user_process "Timeout:0"
```

[1] Do not execute this command more than once per second.

2.2 File/Directory Management Commands

Essential for navigating and manipulating the filesystem, these commands allow users to create, list, modify, and remove files and directories, making them indispensable for day-to-day operations on a Unix-like system.

2.2.1 basename

This command can be used to strip directory path and .SUFFIX from FILE.

Synopsis:

```
basename FILE [SUFFIX]
```

Examples:

Strip directory from `/home/myfile.txt` path, including `.txt` extension.

```
basename /home/myfile.txt .txt
```

```
myfile
```

2.2.2 cat

This command concatenates files and print on the standard output.

Synopsis:

```
cat [-u] [<file>] ...
```

Options:

Option	Description
`-u`	Ignored since unbuffered I/O is always used.

Table 21: cat options

Examples:

View the contents of file /proc/tty/driver/spear_serial (info about serial ports of v2 routers).

```
cat /proc/tty/driver/spear_serial
```

Copy the contents of the router configuration files in /tmp/my.cfg.

```
cat /etc/settings.* > /tmp/my.cfg
```

2.2.3 cd

This command can be used to change the current working directory.

Synopsis:

```
cd [-P] [-L] [<directory>]
```

Options:

Option	Description
`-P`	Do not follow symbolic links
`-L`	Follow symbolic links (default)

Table 22: cd options

Examples:

Move to home directory (/root).

```
cd
```

Move to directory /mnt.

```
cd /mmt
```

2.2.4 chdir

The `chdir` command, also known as `cd` in many shell environments, is utilized to change the current working directory of the shell session.

2.2.5 cmp

The *cmp* utility compares two files of any type and writes the results to the standard output.

Synopsis:

```
cmp [-l] [-s] <file1> <file2> [<skip1> [<skip2>]]
```

Options:

Option	Description
`-l`	Print the byte number (decimal) and the differing byte values (octal) for each difference.
`-s`	Print nothing for differing files; return exit status only.

Table 23: cmp options

By default, *cmp* is silent if the files are the same; if they differ, the byte and line number at which the first difference occurred is reported. Bytes and lines are numbered beginning with one. If *<file2>* is not specified, standard input is used instead.

The optional arguments *<skip1>* and *<skip2>* are the byte offsets from the beginning of *<file1>* and *<file2>* respectively, where the comparison will begin. The offset is decimal by default, but may be expressed as an hexadecimal or octal value.

2.2.6 cp

This command can be used to copy files and directories.

Synopsis:

```
cp [<option>] <source> <dest>
```

Options:

Option	Description
-a	Preserve the all attributes.
-R, -r	Copy directories recursively.
-d, -P	Never follow symbolic links.
-H, -L	Follow command-line symbolic links.
-p	Preserve the mode, ownership, timestamps attributes.
-f	If an existing destination file cannot be opened, remove it and try again.
-i	Prompt before overwrite.
-n	Don't overwrite.
-l, -s	Create (sym)links
-T	Refuse to copy if DEST is a directory
-t DIR	Copy all SOURCEs into DIR
-u	Copy only newer files

Table 24: cp options

Examples:

Copy the system log to directory /mnt.

```
cp /var/log/messages* /mnt
```

Copy configuration profile "Alternative 1" to profile "Standard".

```
cp -r /etc/alt1/* /etc
```

2.2.7 cut

This command prints selected fields from each input FILE to standard output.

Synopsis:

```
cut [OPTIONS] [FILE]...
```

Options:

Option	Description
`-b LIST`	Output only bytes from LIST
`-c LIST`	Output only characters from LIST
`-d CHAR`	Field delimiter for input (default -f TAB, -F run of whitespace)
`-O CHAR`	Field delimeter for output (default = -d for -f, one space for -F)
`-D`	Don't sort/collate sections or match -fF lines without delimeter
`-f LIST`	Print only these fields (-d is single char)
`-s`	Output only the lines containing delimiter
`-n`	Ignored

Table 25: cut options

Examples:

Display the 1st field of each line, using tab as the field separator.

```
cut -f1 file.txt
```

Displays the 2nd field of each line, using colon as the field separator.

```
echo a:b | cut -d: -f2
```

```
b
```

Display the 2nd and every later field.

```
echo a b c d | cut -d" " -f2-
```

```
b c d
```

Instead of fields, treats characters. Display the third and fourth character.

```
echo abcd | cut -c3,4
```

```
cd
```

2.2.8 dd

This command can be used to copy a file and convert the data format in the process, according to the operands specified.

Synopsis:

```
dd [if=FILE] [of=FILE] [bs=N] [count=N] [skip=N] [seek=N]
```

Options:

Option	Description
`if=FILE`	Read from FILE instead of stdin
`of=FILE`	Write to FILE instead of stdout
`bs=N`	Read and write N bytes at a time
`count=N`	Copy only N input blocks
`skip=N`	Skip N input blocks
`seek=N`	Skip N output blocks
`N`	May be suffixed by c (1), w (2), b (512), kB (1000), k (1024), MB, M, GB, G

Table 26: dd options

Examples:

Erase the microSD card, which is available as `/dev/sda`.

```
dd if=/dev/zero of=/dev/sda bs=4k
```

2.2.9 decode

This command is designed for decoding Base64-encoded values, making it an essential tool for working with configuration files or other data that utilize Base64 encoding for obfuscation or storage efficiency. The decoded value is printed to the standard output, allowing for direct observation or redirection to a file for further use.

Synopsis:

```
decode <base64>
```

Description:

The *decode* command takes a single argument, a Base64-encoded string, and prints its decoded value. This functionality is particularly useful in scenarios where configuration files or data are encoded to prevent manipulation or ensure compatibility.

Options: This proprietary command is straightforward and does not require additional options for its operation.

Examples:

Decode a Base64-encoded string.

```
decode SGVsbG8sIFdvcmxkIQ==
```

This example will output the decoded string of the provided Base64-encoded value, in this case, "Hello, World!".

2.2.10 dirname

This command strips non-directory suffix from FILENAME.

Synopsis:

`dirname FILENAME`

Examples:

Strip non-directory suffix from `/home/MyFolder/myfile.txt` path.

```
dirname /home/MyFolder/myfile.txt
/home/MyFolder
```

2.2.11 find

Command to search for files in a directory hierarchy.

Synopsis:

`find [<path> ...] [<expression>]`

Options:

The default path is the current directory, default expression is '-print'. Type `find ---help` for help or look up online man page for more detailed description. Expression may consist of:

Option	Description
`-follow`	Dereference symbolic links
`-name <pattern>`	File name (leading directories removed) matches <pattern>
`-print`	Print (default and assumed)
`-type X`	Filetype matches X (where X is one of: f,d,l,b,c,...)
`-perm <perms>`	Permissions match any of (+NNN); all of (-NNN); or exactly (NNN)
`-mtime <days>`	Modified time is greater than (+N); less than (-N); or exactly (N) days
`-mmin <mins>`	Modified time is greater than (+N); less than (-N); or exactly (N) minutes
`-exec <cmd>`	Execute command with all instances of {} replaced by the files matching <expression>

Table 27: find expressions

Examples:

Search for files in your home directory which have been modified in the last twenty-four hours.

`find $HOME -mtime 0`

Search for files which have read and write permission for their owner, and group, but which other users can read but not write to.

`find . -perm 664`

2.2.12 grep

This program searches the named input FILEs (or standard input if no files are named, or the file name – is given) for lines containing a match to the given PATTERN. By default, grep prints the matching lines.

Synopsis:

```
grep [<options> ...] <pattern> [<file> ...]
```

Options:

Option	Description
-H	Print the filename for each match
-h	Suppress the prefixing of filenames on output when multiple files are searched
-n	Prefix each line of output with the line number within its input file
-l	Suppress normal output; instead print the name of each input file from which output would normally have been printed
-L	Suppress normal output; instead print the name of each input file from which no output would normally have been printed
-c	Suppress normal output; instead print a count of matching lines for each input file
-o	Show only the matching part of line
-q	Quiet; do not write anything to standard output. Exit immediately with zero status if any match is found, even if an error was detected. Also see the *-s* or *--no-messages* option.
-v	Invert the sense of matching, to select non-matching lines
-s	Suppress error messages about nonexistent or unreadable files
-r	Recurse
-R	Recurse and dereference symlinks
-i	Ignore case distinctions
-w	Match whole words only
-x	Match whole lines only
-F	Interpret PATTERN as a list of fixed strings, separated by new lines, any of which is to be matched
-E	PATTERN is an extended regexp
-m N	Match up to N times per file
-e PTRN	Use PATTERN as the pattern; useful to protect patterns beginning with –
-f FILE	Obtain patterns from FILE, one per line

Table 28: grep options

Examples:

See all lines of system log in which occurs the word "error".

```
grep error /var/log/messages
```

View all processes whose name the contents of the string "ppp".

```
ps | grep ppp
```

2.2.13 gunzip

This program can be used to decompress FILE (or standard input if filename is '–').

Synopsis:

```
gunzip [-c] [-f] [-t] <filename>
```

Options:

Option	Description
-c	Write output on standard output
-f	Force decompression even if the file has multiple links or the corresp. file already exists, or if the compressed data is read from or written to a terminal.
-t	Test. Check the compressed file integrity.

Table 29: gunzip options

Examples:

Decompression of file test.tar.gz (creates file test.tar).

```
gunzip test.tar.gz
```

2.2.14 gzip

This program can be used to compress FILE with maximum compression.

Synopsis:

```
gzip [-c] [-d] [-f] <filename>
```

Options:

Option	Description
-c	Write output on standard output
-d	Decompress
-f	Force compression even if the file has multiple links or the corresponding file already exists, or if the compressed data is read from or written to a terminal

Table 30: gzip options

Examples:

Compression of file test.tar (creates file test.tar.gz).

```
gzip test.tar
```

2.2.15 head

This program prints first 10 lines of each file to standard output. With more than one file, precede each with a header giving the file name. With no file, or when file is a dash ("-"), read standard input.

Synopsis:

```
head [<option(s)>] [<file(s)>]
```

Options:

Option	Description
`-n NUM`	Print first NUM lines instead of first 10.
`-c NUM`	Output the first NUM bytes.
`-q`	Never output headers giving file names.
`-v`	Always output headers giving file names.

Table 31: head options

2.2.16 jq

This command is not supported by routers of *v1* and v2 production lines.

This command is a powerful tool for processing JSON inputs, applying filters to JSON text, and producing the output as JSON. It can manipulate, filter, and transform structured data with its directive.

Synopsis:

```
jq [options] <jq filter> [file...]
jq [options] --args <jq filter> [strings...]
jq [options] --jsonargs <jq filter> [JSON_TEXTS...]
```

Options:

Option	Description
`-r`	Output raw strings, not JSON texts.
`-c`	Produce compact output instead of pretty-printed output.
`-f`	Load a jq program from a file.
`-arg name value`	Pass argument to the filter.

Table 32: jq options

Examples:

`echo '{"foo": 0}' | jq .`
Outputs the JSON object unmodified, with formatting.

`echo '{"foo": 0, "bar": 1}' | jq '.bar'`
Extracts the value of the key `bar` from the JSON input.

`echo '[{"foo": 0}, {"foo": 1}]' | jq '.[] | .foo'`
Applies the filter to each element in the JSON array, outputting the value of the `foo` key.

`cat data.json | jq --arg keyName "myKey" --arg keyValue "myValue" '. + {$keyName: $keyValue}'`
Adds a new key-value pair to the JSON object read from `data.json` .

2.2.17 less

This command is a terminal pager program that displays the contents of a text file one screen at a time. Unlike most pager programs, `less` allows backward movement in the file as well as forward movement.

Synopsis:

```
less [options] file ...
```

Options:

For a complete list of options, refer to the less(1) — Linux manual page.
Some commonly used options include:

Option	Description
`-N`	Displays line numbers at the beginning of each line.
`-i`	Ignores the case when searching.
`-F`	Exits if the content can be displayed on one screen.
`-R`	Displays ANSI color escape sequences in raw form.

Table 33: less options

Examples:

`less file.txt`
Displays the contents of `file.txt` , allowing navigation through the file.

`less -N file.txt`
Displays `file.txt` with line numbers.

`grep 'search_term' file.txt | less -R`
Searches for `search_term` in `file.txt` and pipes the output to `less` , allowing the user to view the search results with ANSI color formatting.

`less +F log.txt`
Opens `log.txt` in `less` , automatically following the end of the file as new lines are added, similar to `tail -f` .

2.2.18 ln

This program can be used to make links between files.

Synopsis:

```
ln [ option ] < target > ... < link_name > | < directory >
```

Options:

Option	Description
-s	Make symbolic links instead of hard links
-f	Remove existing destination files
-n	No dereference symlinks – treat like normal file
-b	Make a backup of the target (if exists) before link operation
-S	Use suffix instead of ∼ when making backup files

Table 34: ln options

Examples:

Creating a symbolic link to file /var/log/messages called my.log.

```
ln -s /var/log/messages my.log
```

2.2.19 ls

This program can be used to list directory contents.

Synopsis:

```
ls [ option ] < filename > ...
```

Options:

Option	Description
-1	List files in a single column
-A	Do not list implied . and ..
-a	Do not hide entries starting with .
-C	List entries by columns
-c	With -l: show ctime
-d	List directory entries instead of contents
-e	List both full date and full time
-i	List the i-node for each file
-l	Use a long listing form
-n	List numeric UIDs and GIDs instead of names
-L	List entries pointed to by symbolic links
-r	Sort the listing in reverse order
-S	Sort the listing by file size
-s	List the size of each file, in blocks
-t	With -l: show modification time
-u	With -l: show access time
-v	Sort the listing by version
-x	List entries by lines instead of by columns
-X	Sort the listing by extension

Table 35: ls options

Examples:

View detailed content of directory /mnt.

```
ls -l /mnt
```

View list contents of actually directory.

```
ls
```

2.2.20 mkdir

This program used to make directories.

Synopsis:

```
mkdir [<option>] directory ...
```

Options:

Option	Description
-m	Set permission mode (as in chmod), not rwxrwxrwx – umask
-p	No error if existing, make parent directories as needed

Table 36: mkdir options

Examples:

Create directory /tmp/test/example.

```
mkdir -p /tmp/test/example
```

2.2.21 mv

This program can be used to move or rename files.

Synopsis:

```
mv [-f] [-fin] <source> ... <dest>
```

Options:

Option	Description
-f	Don't prompt before overwriting
-i	Interactive, prompt before overwrite
-n	Don't overwrite an existing file
-T	Refuse to move if DEST is a directory
-t DIR	Move all SOURCEs into DIR

Table 37: mv options

Examples:

Rename file abc.txt na def.txt.

```
mv abc.txt def.txt
```

Move all files with the extension txt to the directory /mnt.

```
mv *.txt /mnt
```

2.2.22 pwd

This program can be used to print the path of the current directory.

Synopsis:

`pwd`

Examples:

Print the path of the current directory, which is `/home/httpd` in this case.

`pwd` `/home/httpd`

2.2.23 readlink

`readlink` command displays the value of a symlink stored in a link file.

Synopsis:

`readlink [OPTIONS] [FILE]`

Options:

Option	Description
`-f`	Canonicalize by following all symlinks.
`-n`	Don't add newline to the output.
`-v`	Verbose output, report error messages if any.

Table 38: readlink options

Examples:

Display the symlink for sudo command.

```
readlink /usr/bin/sudo
/usr/bin/doas
```

2.2.24 realpath

This command returns the absolute pathname of given file name.

Synopsis:

```
realpath FILE
```

Examples:

Return the absolute pathname of `myfile.txt` located at current directory, here in `/home/MyFolder/`.

```
realpath myfile.txt
/home/MyFolder/myfile.txt
```

2.2.25 rm

This program can be used to remove files or directories.

Synopsis:

```
rm [-i] [-f] [-r] <file> ...
```

Options:

Option	Description
`-i`	Always prompt before removing each destination
`-f`	Remove existing destinations, never prompt
`-r`	Remove the contents of directories recursively

Table 39: rm options

Examples:

Remove all files with extension txt in the current directory.

```
rm *.txt
```

Remove directory /tmp/test and all subdirectories.

```
rm -rf /tmp/test
```

2.2.26 rmdir

This program can be used to remove empty directories.

Synopsis:

```
rmdir <filename>
```

Examples:

Remove empty directory /tmp/test.

```
rmdir /tmp/test
```

2.2.27 sed

This program can be used for filtering and transforming text.

Synopsis:

```
sed [ -e ] [ -f ] [ -i ] [ -n ] [ -r ] pattern [ -files ]
```

Options:

Option	Description
`-e`	Add the script to the commands to be executed
`-f`	Add script-file contents to the commands to be executed
`-i`	Edit files in place (makes backup if extension supplied)
`-n`	Suppress automatic printing of pattern space
`-r`	Use extended regular expression syntax

Table 40: sed options

If no `-e` or `-f` is given, the first non-option argument is taken as the sed script to interpret. All remaining arguments are names of input files; if no input files are specified, then the standard input is read. Source files will not be modified unless -i option is given.

Examples:

Change parameter PPP_APN in file /etc/settings.ppp to value "internet".

```
sed -e "s/\(PPP_APN=\).*/\1internet/" -i /etc/settings.ppp
```

2.2.28 shred

This program can be used to delete a file completely from a non-volatile memory. This command overwrites the contents of a file multiple times, using patterns chosen to maximize the destruction of the residual data, making it harder for even very expensive hardware probing to recover it.

Synopsis:

```
shred [OPTIONS] [FILE]
```

Options:

Option	Description
`-f`	Change permissions to allow writing if necessary.
`-n N`	Overwrite N times instead of the default (default is 3 time).
`-z`	Add a final overwrite with zeros to hide shredding.
`-u`	Truncate and remove file after overwriting.

Table 41: shred options

Examples:

Overwrite the data of `file1.txt`, and `file2.txt` using the default shredding methods.

```
shred file1.txt file2.txt
```

Overwrite the data of `file1.txt` using the default shredding methods and delete the file.

```
shred -u file1.txt
```

Overwrite the data of `file1.txt` 10 times.

```
shred -n 10 file1.txt
```

2.2.29 split

This program splits a single file (INPUT) into multiple files. A custom PREFIX for the name of the output files can by specified.

Synopsis:

```
split [OPTIONS] [INPUT [PREFIX]]
```

Options:

Option	Description
`-b N[k|m]`	Split input file by N (kilo|mega)bytes.
`-l N`	Split input file by N lines (by default 1000 lines).
`-a N`	Use N letters as suffix for the name of the output files (by default 2 letters).

Table 42: split options

Examples:

Split file *file.img* into files (*xaa*, *xab*, *xac*, ...) every with size of 50 kB:

```
split -b 50k file.img
```

Split file *file.txt* into files (*file_a*, *file_b*, *file_c*, ...) every containing of 200 lines:

```
split -l 200 -a 1 file.txt file_
```

2.2.30 sync

This command forces an immediate transfer of buffered data blocks in memory or in FILEs to the disk.

Synopsis:

```
sync [OPTIONS] [FILEs]...
```

Options:

Option	Description
`-d`	Avoid syncing metadata.
`-f`	Sync filesystems underlying FILEs.

Table 43: sync options

2.2.31 tail

This program can be used to output the last part of files.

Synopsis:

```
tail [ -n <number>] [ -f ]
```

Options:

Option	Description
-n	Print last N lines instead of last 10
-f	Output data as the file grows

Table 44: tail options

Examples:

Show last 30 lines of /var/log/messages.

```
tail -n 30 /var/log/messages
```

2.2.32 tar

This program can be used to create, extract or list files from a tar file.

Synopsis:

```
tar -[czxtv0] [ -f tarfile ] [ -C dir ] [ file ] ...
```

Options:

Option	Description
`c`	Create
`x`	Extract
`t`	List
`-f FILE`	Name of TARFILE or "-" for stdin
`-C DIR`	Change to directory DIR before operation
`-v`	Verbosely list files processed
`-O`	Extract to stdout
`-o`	Don't restore user:group
`-k`	Don't replace existing files
`-z`	(De)compress using gzip
`-a`	(De)compress based on extension
`-h`	Follow symlinks

Table 45: tar options

Examples:

Creating log.tar archive that contains files from the directory /var/log.

```
tar -cf log.tar /var/log
```

Extract files from the archive log.tar.

```
tar -xf log.tar
```

2.2.33 touch

This program can be used to update timestamp of file.

Synopsis:

```
touch [-c] <file> [<file> ...]
```

Options:

Option	Description
`-c`	Do not create any files
`-h`	Do not follow links

Table 46: touch options

Examples:

Create a file, respectively update timestamp of file /tmp/test.

```
touch /tmp/test
```

2.2.34 vi

This program can be used to edit and read text file.

Synopsis:

```
vi [-R] [<file> ...]
```

Options:

Option	Description
-R	Read only, do not write to the file

Table 47: vi options

Examples:

Open file /etc/rc.local in the text editor vi.

```
vi /etc/rc.local
```

2.2.35 wc

Print newline, word, and byte counts for each file, and a total line if more than one file is specified. With no file, or when file is a dash ("-"), read standard input.

Synopsis:

```
wc [<option(s)>] [<file(s)>]
```

Options:

Option	Description
-c	Print the byte counts.
-l	Print the newline counts.
-L	Print the length of the longest line.
-w	Print the word counts.

Table 48: wc options

2.2.36 xxd

This is a program is a command-line utility that creates a hex dump of a given file or standard input. It can also convert a hex dump back to its original binary form. This tool is commonly used for debugging, examining binary files, and performing binary file analysis.

Synopsis:

```
xxd [-pri] [-g N] [-c N] [-l LEN] [-s OFS] [-o OFS] [FILE]
```

Options:

Option	Description
`-g N`	Bytes per group
`-c N`	Bytes per line
`-p`	Show only hex bytes, assumes -c30
`-i`	C include file style
`-l LENGTH`	Show only first LENGTH bytes
`-s OFFSET`	Skip OFFSET bytes
`-o OFFSET`	Add OFFSET to displayed offset
`-r`	Reverse (with -p, assumes no offsets in input)

Table 49: xxd options

Examples:

`xxd file.txt`

This command generates a hex dump of `file.txt` , displaying both the hexadecimal values and their corresponding ASCII characters. It's commonly used for inspecting the contents of a file, especially in debugging or when dealing with binary data.

`xxd -p file.bin > file.hex`

In this scenario, the `-p` option is used to create a plain hex dump of a binary file (`file.bin`). The output is redirected to a new file (`file.hex`). This format is easier to read and manipulate, particularly useful in scenarios involving binary data analysis or modification.

`xxd -r file.hex > file.bin`

This example shows how to reverse the process: converting a hex dump (`file.hex`) back into its original binary form (`file.bin`). The
inlinecode-r option is used for this reverse operation. It's particularly useful when you've made changes to the hex representation of a file and need to revert it to its binary format.

2.2.37 zcat

This command is a utility in Unix-like operating systems that allows users to view the contents of gzip-compressed files directly, without the need to explicitly decompress them first. It is equivalent to running the `gunzip -c` command and is useful for quickly inspecting the contents of compressed files or for piping the output of compressed files into other commands or programs for further processing.

Synopsis:

```
zcat [options] [file ...]
```

Description:

Zcat concatenates the uncompressed contents of compressed files to standard output. When no files are specified, or when the file name `-` is used, *zcat* reads from standard input. *Zcat* will decompress all specified files in order, making it convenient for viewing multiple compressed files.

Options:

- `-h, -help`: Display a help message and exit.
- `-V, -version`: Display version information and exit.
- Additional options available in *zcat* are generally passed to the *gzip* command, as *zcat* is often a symlink to *gzip*.

Examples:

View the contents of a compressed file:

```
zcat file.gz
```

Concatenate multiple compressed files:

```
zcat file1.gz file2.gz file3.gz
```

Pipe the contents of a compressed file into `grep` :

```
zcat file.gz | grep 'search_pattern'
```

The `zcat` command's ability to directly read compressed files makes it an invaluable tool for quickly accessing or processing data within gzip-compressed files without the overhead of decompression.

2.3 System Commands

System administration commands provide the necessary tools for managing users, system services, and hardware settings. They are crucial for maintaining the system's integrity, security, and performance.

2.3.1 adduser

Incorrect usage of this command may lead to system malfunction.

The `adduser` command is used to create a new user or add an existing user to a specified group.

Synopsis:

```
adduser [OPTIONS] USER [GROUP]
```

Options:

Option	Description
`-s SHELL`	Specify the login shell for the new user.
`-G GRP`	Add the user to the specified group.
	Undocumented options are not recommended for use if you are not sure what you are doing, as their incorrect usage may lead to system malfunction.

Table 50: adduser options

Examples:

Create a new user `john` with `user` role:

```
adduser -s /bin/false -G users john
```

Create a new user `george` with `admin` role:

```
adduser -s /bin/sh -G root george
```

2.3.2 backup

This program is used to create a backup of the router configuration. The backup data is written to the standard output (`stdout`) and can be redirected to a file using shell redirection, as shown in the examples below. A previously saved configuration can be restored using the *restore* command; see Chapter *2.3.26 restore* for details.

The backup output is generated in a plain-text, key-value format that is suitable for storage, transfer, and later restoration. Sensitive data can optionally be filtered out or encrypted.

Synopsis:

```
backup [<options>] [<filename>]
```

Note: The optional `<filename>` is not passed as a command argument. The backup file is specified using shell output redirection (>), as shown in the examples.

Options:

Option	Description
`-c`	Back up the configuration excluding user account data (default if no option is specified).
`-u`	Back up only user account data.
`-a`	Back up the complete configuration, including user accounts, scripts, and Router Apps settings.
`-f`	Filter sensitive information (passwords, keys, secrets, and passphrases are replaced by `###REMOVED###`).
`-p <password>`	Encrypt the backup using AES-256; the output is Base64 encoded.

Table 51: backup options

Examples:

Back up the whole configuration to a file

```
backup -a > /tmp/my.cfg
```

Back up configuration only (without user accounts)

```
backup -c > /tmp/config-only.cfg
```

Back up only user accounts

```
backup -u > /tmp/users.cfg
```

Back up configuration while hiding sensitive data

```
backup -a -f > /tmp/support-safe.cfg
```

Create an encrypted backup

```
backup -a -p MySecretPass > /tmp/backup.enc
```

View backup directly on the console

```
backup -c
```

Create a filtered, encrypted backup in one step

```
backup -a -f -p MySecretPass > /tmp/secure-backup.enc
```

2.3.3 chmod

This command is used to change file and directory permissions (mode bits) in a Unix-like system. It allows you to control who can read, write, or execute a file or directory.

For a detailed explanation of permission modes and their numeric or symbolic notation, see: chmod(1) — Linux manual page

Synopsis:

```
chmod [-R] <mode> <filename>
```

Options:

Option	Description
`-R`	Change permissions of files and directories recursively.

Table 52: chmod options

Examples:

Make a script executable by its owner and readable by others:

```
chmod 755 /tmp/script.sh
```

Allow full access for the owner and read-only access for others:

```
chmod 744 /tmp/config.cfg
```

Change permissions for all files in a directory recursively:

```
chmod -R 755 /opt/custom
```

2.3.4 chown

This command changes the user and/or group ownership of files and directories. It is commonly used by administrators to manage access rights and file control in Unix-like systems.

For detailed usage and syntax, see: chown(1) — Linux manual page

Synopsis:

```
chown [-R] <user>[:<group>] <filename>
```

Options:

Option	Description
`-R`	Change ownership of files and directories recursively.

Table 53: chown options

Examples:

Change the owner of a file to user `root`:

```
chown root /tmp/config.cfg
```

Change both owner and group of a directory recursively:

```
chown -R admin:admin /opt/custom
```

Change only the group of a file:

```
chown :network /tmp/log.txt
```

2.3.5 chpasswd

This utility can be used to update user passwords, particularly for bulk updates of multiple users' passwords and for updating without user interaction, such as in scripts.

Synopsis:

```
chpasswd [options]
```

Options:

Option	Description
`-c, -crypt-method METHOD`	Specify the crypt method (one of NONE, DES, MD5, SHA256, SHA512, BCRYPT, YESCRYPT)
`-e, -encrypted`	Supplied passwords are encrypted
`-h, -help`	Display this help message and exit
`-m, -md5`	Encrypt the clear text password using the MD5 algorithm
`-R, -root CHROOT_DIR`	Directory to chroot into
`-P, -prefix PREFIX_DIR`	Directory prefix
`-s, -sha-rounds`	Number of rounds for the SHA, BCRYPT, or YESCRYPT crypt algorithms

Table 54: chpasswd options

Examples:

Update the password for a user, encrypting the password with MD5:

```
chpasswd -m <<EOF
username:newpassword
EOF
```

Update the password for a user, providing an already encrypted password:

```
chpasswd -e <<EOF
username:encryptedpassword
EOF
```

Update the password for two users, specifying the crypt method to SHA512:

```
chpasswd -c SHA512 <<EOF
username:newpassword
username2:newpassword
EOF
```

2.3.6 clog

This command can be used to print the connection logs.

2.3.7 date

This command can be used to display the current time in the given FORMAT, or set the system date (and time).

Synopsis:

```
date [-R] [-d <string>] [-s] [-r <file>] [-u] [MMDDhhmm[[CC]YY][.ss]]
```

Options:

Option	Description
`-R`	Output date and time in RFC 2822 format
`-d <string>`	Display time described by STRING, not 'now'
`-s`	Set time described by STRING
`-r <file>`	Display the last modification time of FILE
`-u`	Print or set Coordinated Universal Time

Table 55: date options

Examples:

Display the current date and time.

```
date
```

Setting the date and time on December 24, 2011 20:00.

```
date 122420002011
```

2.3.8 deluser

Incorrect usage of this command may lead to system malfunction.

The `deluser` command is used to delete a user from the system.

Synopsis:

```
deluser [--remove-home] USER
```

Description:

The `deluser` command removes the specified user from the system. If the `-remove-home` option is used, the user's home directory and mail spool will also be deleted.

Options:

Option	Description
`-remove-home`	Remove the user's home directory and mail spool along with the user account

Table 56: deluser options

Examples:

Delete a user named `testuser` from the system:

```
deluser testuser
```

Delete a user named `testuser` and remove their home directory and mail spool:

```
deluser --remove-home testuser
```

2.3.9 df

This command can be used to view report file system disk space usage.

Synopsis:

```
df [-PkT] [-t TYPE] [FILESYSTEM...]
```

Options:

Option	Description
`-P`	POSIX output format
`-k`	Print sizes in kilobytes
`-T`	Print filesystem type
`-t TYPE`	Print only mounts of this type

Table 57: df options

2.3.10 dmesg

This command can be used to display the Kernel log messages.

Synopsis:

```
dmesg [-R] [-T] [-c]
```

Options:

Option	Description
`-R`	Display relative time since the router booted in *sec.nanosec* format.
`-T`	Display human-readable timestamp in *YYYY-MM-DD hh:mm:ss* format.
`-c`	Read and clear all messages.

Table 58: dmesg options

Examples:

Display latest Kernel log messages and subsequent deletion of the Kernel ring buffer.

```
dmesg -c
```

Display latest Kernel log messages including the human-readable timestamp.

```
dmesg -T
```

2.3.11 doas

This command can be used to execute commands as another user. The command argument is mandatory unless -C, -L, or -s is specified. The user will be required to authenticate by entering their password, unless configured otherwise.

By default, a new environment is created. The variables `HOME`, `LOGNAME`, `PATH`, `SHELL`, and `USER` and the `umask` are set to values appropriate for the target user. `DOAS_USER` is set to the name of the user executing `doas`. The variables `DISPLAY` and `TERM` are inherited from the current environment. This behavior may be modified by the config file. The working directory is not changed.

```
doas [-Lns] [-C config] [-u user] command [args]
```

Options:

Option	Description
`-L`	Clear any persisted authentications from previous invocations, then immediately exit. No command is executed.
`-n`	Non interactive mode, fail if the matching rule doesn't have the nopass option.
`-s`	Execute the shell from `SHELL` or */etc/passwd*.
`-C config`	Parse and check the configuration file config, then exit. If command is supplied, `doas` will also perform command matching. In the latter case either 'permit', 'permit nopass' or 'deny' will be printed on standard output, depending on command matching results. No command is executed.
`-u user`	Execute the command as *user*. The default is *root*.

Table 59: doas options

Exit Status:

The `doas` utility exits 0 on success, and > 0 if an error occurs. It may fail for one of the following reasons:

- The config file /etc/doas.conf could not be parsed.
- The user attempted to run a command which is not permitted.
- The password was incorrect.
- The specified command was not found or is not executable.

2.3.12 faillock

This command is not supported by routers of *v1* production line.

This program is used to handle user login failures. It allows listing failures for each user, resetting failures, or eventually resetting locked users. Note that this feature is associated with account locking after exceeding the allowed number of unsuccessful login attempts and is not related to the administrative account locking function in the GUI.

Synopsis:

`faillock [--user username] [--reset] [--legacy-output]`

Options:

Option	Description
`-user username`	Apply the command to the specified user.
`-reset`	Reset the failure records. This can be used to manually unlock accounts or reset the failure count.
`-legacy-output`	Display output in the legacy format.

Table 60: faillock options

Examples:

View the authentication failure records for all users.

`faillock`

View the authentication failure records for the user `john`.

`faillock --user john`

```
john:
When                Type    Source          Valid
2024-07-25 12:31:22 RHOST   10.64.0.1           V
2024-08-22 11:30:30 RHOST   10.64.0.25          V
```

View the authentication failure records using the legacy output format.

`faillock --legacy-output`

```
Login           Failures    Latest failure          From
john            2           2024-08-22 11:30:30     10.64.0.25
root            0
faillock: Error no such user: Test_admin
faillock: Error no such user: pam_client
faillock: Error no such user: test_admin
```

Reset the failure records for the user `john`.

`faillock --user john --reset`

2.3.13 free

The `free` command displays information about free and used memory, reported in kB (kilobytes).

Synopsis:

`free`

Options:

This command does not support any options.

Example:

```
~ # free
              total        used        free      shared  buff/cache   available
Mem:        1008588      141496      834472         188       32620      854040
Swap:             0           0           0
```

Columns in this examle represents the following:

- **total**: The total physical memory in the system (in kilobytes). In this example, the system has 1,008,588 KB (985 MB) of RAM.
- **used**: The amount of memory currently used by running processes.
- **free**: The memory that is completely unallocated.
- **shared**: Memory used by temporary shared memory segments.
- **buff/cache**: Memory used by the kernel for buffers and caches. Although this memory is marked as “used,” it is available to be reclaimed quickly when needed.
- **available**: An estimate of how much memory is available for new applications without swapping. This value includes both free memory and memory that can be quickly freed from the buffer/cache.

Difference Between Free and Available Memory

- **Free Memory**: This refers to the portion of RAM that is not being used at all. It is completely idle and unallocated. However, relying solely on this metric can be misleading because modern Linux systems deliberately use much of the free memory for caching to speed up system performance.
- **Available Memory**: This is a more meaningful metric for determining how much memory is truly available for applications. It not only accounts for the free memory but also includes the memory used for caching and buffers that can be quickly reclaimed. Therefore, even if the "free" memory appears low, a high "available" memory indicates that the system can still allocate memory for new processes without resorting to swap space.

2.3.14 fwupdate

This program can be used for router's ICR-OS firmware update.

Synopsis:

```
fwupdate [-i <filename> [-h] [-n]] [-f] [-c]
```

Options:

Option	Description
-i	File of the new ICR-OS, filename has to be specified
-h	HTML output (used when called from web configuration)
-n	Do not reboot after ICR-OS update
-f	Finish update procedures (restore configuration, complete firmware switch)
-c	Check firmware update status

Table 61: fwupdate options

2.3.15 id

This command is utilized to display user and group information for a specified user, or for the current user if no user is specified. It provides details such as user ID (UID), group ID (GID), and supplementary groups associated with the user.

Synopsis:

`id [OPTIONS] [USER]`

Description:

By default, without options, the command prints the UID, GID, and supplementary groups of the specified user or the current user. The output includes both the IDs and the names of the user and groups.

Options:

Option	Description
`-u`	Prints the user ID of the specified user or the current user
`-g`	Prints the primary group ID of the specified user or the current user
`-G`	Prints all the supplementary group IDs of the specified user or the current user
`-n`	When used with `-u`, `-g`, or `-G`, prints the name(s) of the user or group(s) instead of the numeric ID(s)
`-r`	Prints the real, rather than effective, user or group ID

Table 62: id options

Examples:

Print the current user's UID, GID, and supplementary groups:
`id`

Print the UID of the current user:
`id -u`

Print the primary group name of the user:
`id -gn`

Print all supplementary group names of the current user:
`id -Gn`

For a comprehensive guide on the `id` command and its options, consult the man pages or official documentation available on your system or online.

2.3.16 kill

This command can be used to terminate process.

Synopsis:

```
kill [ -<signal> ] <process-id> [ <process-id> ... ] \[1mm] kill -l
```

Options:

Option	Description
-l	Print a list of signal names. These are found in /usr/include/linux/signal.h

Table 63: kill options

Examples:

End the process with PID 1234 by sending signal SIGTERM.

```
kill 1234
```

End the process with PID 1234 by sending signal SIGKILL.

```
kill -9 1234
```

2.3.17 killall

This command can be used to kill all process with process name.

Synopsis:

```
killall [ -q] [ -<signal> ] <process-name> [<process-name> ...]
```

Options:

Option	Description
-l	Print a list of signal names. These are found in /usr/include/linux/signal.h
-q	Do not complain if no processes were killed

Table 64: killall options

Examples:

End the all processes with name pppd by sending signal SIGTERM.

```
killall pppd
```

End the all processes with name pppd by sending signal SIGKILL.

```
killall -9 pppd
```

2.3.18 klog

This command can be used to print the kernel logs.

2.3.19 logger

This program makes entries in the system log. It provides a shell command interface to the system log module.

Synopsis:

```
logger [ option ] [ message ... ]
```

Options:

Option	Description
`-i`	Log the process id of the logger process with each line
`-s`	Log the message to standard error, as well as the system log
`-f <file>`	Log the specified file
`-p <priority>`	Enter the message with the specified priority. The priority may be specified numerically or as a facility.level pair.
`-t <tag>`	Mark every line in the log with the specified tag
`-u <socket>`	Write to socket as specified with socket instead of builtin syslog routines
`-d`	Use a datagram instead of a stream connection to this socket

Table 65: logger options

Examples:

Send the message System rebooted to the syslogd daemon.

```
logger System rebooted
```

Send the message System going down immediately!!! to the syslog daemon, at the emerg level and user facility.

```
logger -p user.emerg "System going down immediately!!!"
```

2.3.20 losetup

This program can be used to to set up and control loop devices.

Synopsis:

`losetup [-rP] [-o OFS] -f|LOOPDEV FILE` : associate loop devices
`losetup -c LOOPDEV` : reread file size
`losetup -d LOOPDEV` : disassociate
`losetup -a` : show status
`losetup -f` : show next free loop device

Options:

Option	Description
`-o OFS`	Start OFS bytes into FILE
`-P`	Scan for partitions
`-r`	Read-only
`-f`	Show/use next free loop device

Table 66: losetup options

2.3.21 mount

This program can be used to mount a file system.

Synopsis:

```
mount [-a] [-o] [-r] [-t] [-w] <DEVICE> <NODE> [ -o <option>, ...]
```

Options:

Flag	Description
-a	Mount all filesystems in fstab
-o	One of many filesystem options, listed below
-r	Mount the filesystem read-only
-t	Specify the filesystem type
-w	Mount for reading and writing (default)

Table 67: mount flags

Option	Description
async/sync	Writes are asynchronous / synchronous
atime/noatime	Enable / disable updates to inode access times
dev/nodev	Allow use of special device files / disallow them
exec/noexec	Allow use of executable files / disallow them
suid/nosuid	Allow set-user-id-root programs / disallow them
remount	Re-mount a mounted filesystem, changing its flags
ro/rw	Mount for read-only / read-write
bind	Bind a directory to an additional location
move	Relocate an existing mount point

Table 68: mount options

For detail description this command, visit Linux manual pages.

Examples:

Connect a contents of USB flash drive to the directory /mnt.

```
mount -t vfat /dev/sda1 /mnt
```

2.3.22 openssl

The openssl program is a command line tool for using the various cryptography functions of OpenSSL's crypto library from the shell. It can be used for:

- Creation of RSA, DH and DSA key parameters
- Creation of X.509 certificates, CSRs and CRLs
- Calculation of Message Digests
- Encryption and Decryption with Ciphers
- SSL/TLS Client and Server Tests
- Handling of S/MIME signed or encrypted mail

Synopsis:

`openssl [<option> ...]`

Options:

For detail description this command, visit Linux manual pages.

Examples:

Generate a new key for the SSH server.

```
openssl genrsa -out /etc/certs/ssh\_rsa\_key 512
```

Generate a new certificate for the HTTPS server.

```
openssl req -new -out /tmp/csr -newkey rsa:1024 -nodes -keyout /etc/certs/https\_key
openssl x509 -req -setstart 700101000000Z -setend 400101000000Z -in /tmp/csr -signkey
/etc/certs/https\_key -out /etc/certs/https\_cert
```

2.3.23 passwd

This program can be used to change user passwords. A user with the *User* role can change only their own password. A user with the *root* role can change passwords for all users.

Synopsis:

`passwd [options] [LOGIN]`

Options:

Option	Description
`-a, -all`	Report password status on all accounts
`-d, -delete`	Delete the password for the named account
`-e, -expire`	Force expire the password for the named account
`-h, -help`	Display this help message and exit
`-k, -keep-tokens`	Change password only if expired
`-i, -inactive INACTIVE`	Set password inactive after expiration to INACTIVE
`-l, -lock`	Lock the password of the named account
`-n, -mindays MIN_DAYS`	Set minimum number of days before password change to MIN_DAYS
`-q, -quiet`	Quiet mode
`-r, -repository REPOSITORY`	Change password in REPOSITORY repository
`-R, -root CHROOT_DIR`	Directory to chroot into
`-P, -prefix PREFIX_DIR`	Directory prefix
`-S, -status`	Report password status on the named account
`-u, -unlock`	Unlock the password of the named account
`-w, -warndays WARN_DAYS`	Set expiration warning days to WARN_DAYS
`-x, -maxdays MAX_DAYS`	Set maximum number of days before password change to MAX_DAYS
`-s, -stdin`	Read new token from stdin

Table 69: passwd options

Examples:

Change the password for the logged-in user `john`.

`$ passwd`

```
Changing password for john.
Current password:
New password:
Retype new password:
passwd: password updated successfully
```

Administrator is changing the password for the `john` user. This operation is not permitted to a user with the *User* role.

`# passwd john`

```
New password:
Retype new password:
passwd: password updated successfully
```

2.3.24 pidof

This program lists the PIDs of all processes with names that match the names on the command line.

Synopsis:

```
pidof <process-name> [<option>] [<process-name> ...]
```

Options:

Option	Description
-s	Display only a single PID.

Table 70: pidof options

2.3.25 ps

This program can be used to view information related with the processes on a system.

Synopsis:

```
ps [options]
```

2.3.26 restore

This program restores a previously created router configuration from a backup file generated by the *backup* command; see Chapter *2.3.2 backup*. The backup file may be either plain text or encrypted.

During restoration, the configuration is validated, optionally decrypted, and then applied to the device.

Synopsis:

```
restore [-p <password>] <filename>
```

Options:

Option	Description
`-p <password>`	Decrypt the backup file before restoration (required if the backup was created with `backup -p`).

Table 71: restore options

Notes:

- The `<filename>` must be provided as a normal command argument (unlike `backup`, which uses output redirection).
- If the backup is encrypted, the same password that was used for backup must be supplied with `-p` option.
- When the configuration is changed, the system generates an internal *configuration changed* event.

Examples:

Restore a plain-text backup

```
restore /tmp/my.cfg
```

Restore an encrypted backup

```
restore -p MySecretPass /tmp/backup.enc
```

Typical output messages returned by restore

- `Configuration remains unchanged.`
- `File not found.`
- `Decryption of configuration failed.`
- `Configuration successfully updated.`

2.3.27 rlog

This command can be used to print the emergency logs.

2.3.28 slog

This command can be used to print system log (file `/var/log/message`).

Synopsis:

```
slog [-n <number>] [-f]
```

Options:

Option	Description
`-n`	Print last N lines instead of last 10
`-f`	Output data as the file grows

Table 72: slog options

Examples:

Continuous listing the system log. Listing stops when reaching the maximum number of lines of log.

```
slog -f
```

2.3.29 service

This program can be used to start, stop or restart specified service. You can list all services by `ls /etc/init.d/` command.

Synopsis:

```
service <service_name> <start | stop | restart>
```

Examples:

Start service cron.

```
service cron start
```

Restart service ppp.

```
service ppp restart
```

Restart service syslog.

```
service syslog restart
```

2.3.30 sudo

This command is not supported by the ICR-OS from version 6.2.8 anymore. For compatibility reasons, the `sudo` command is just a symlink to the `doas` command, see Chapter 2.3.11.

2.3.31 sysctl

This program can be used to list and modify kernel parameters at runtime. The parameters available are those listed under */proc/sys/*.

Synopsis:

```
sysctl -p [-enq] [FILE...] / [-enqaw] [KEY[=VALUE]]...
```

Options:

Option	Description
`-p`	Set values from FILEs (default /etc/sysctl.conf).
`-e`	Ignore errors about unknown keys.
`-n`	Disable printing of the key name when printing values.
`-q`	Quiet mode, don't display the values set to stdout.
`-a`	Display all values currently available.
`-w`	Use this option when all arguments prescribe a key to be set.

Table 73: sysctl options

Examples:

Return the value of `kernel.hostname` kernel parameter, don't print the key name.

```
sysctl -n kernel.hostname
Router
```

Set the value of `kernel.hostname` kernel parameter to "example.com".

```
sysctl -w kernel.domainname="example.com"
kernel.domainname = example.com
```

2.3.32 times

This command is a shell builtin that provides information on the accumulated user and system times for the current shell and all of its child processes. It is an essential tool for evaluating the performance of scripts, helping users and administrators gauge the resource usage and efficiency of their commands or scripts executed within the shell.

2.3.33 top

This command displays only free memory, not available memory. For more information, refer to Chapter *2.3.13 free*.

This program provides a dynamic real-time view of a running system. It can display system summary information, as well as a list of processes or threads currently being managed by the kernel.

Synopsis:

```
top [-b] [-nCOUNT] [-dSECONDS]
```

Options:

Option	Description
`-b`	Batch mode - could be useful for sending output from *top* to other programs or to a file. In this mode, *top* will not accept input and runs until the iterations limit you've set with the '-n' command-line option, or until killed.
`-nCOUNT`	Exit after N iterations.
`-dSECONDS`	Delay between updates in *secs* format.

Table 74: touch options

Keys:

Key	Description
`n/m/p/t`	Sort by pid/mem/cpu/time
`r`	Reverse sort
`q, ^C`	Exit

Table 75: touch keys

Examples:

Run *top* program in batch mode and exit after 5 iterations.

```
top -b -n5
```

Run *top* program and update the output every 10 seconds. Exit by *q* key.

```
top -d10
```

2.3.34 umask

This command is a built-in shell command used to set the default permission or file mode creation mask. The setting of the umask determines the permissions that are not set on newly created files and directories. This command is crucial for controlling the default file permissions and ensuring security by restricting the default set of permissions when new files or directories are created.

Synopsis:

`umask [OPTION]... [MODE]`

Description:

`umask` is used without any arguments to display the current umask value in a shell session. The umask can be set by providing an octal or symbolic value representing the set of permissions to mask out (i.e., permissions that won't be set on new files and directories). For example, a umask of 022 prevents new files from being created with write permissions for group and others.

Options:

Option	Description
`-S`	Display the current umask in a symbolic format, which can be more understandable than the default octal representation.
`-p`	Display the output in a format that can be reused as input, facilitating the replication of umask settings in scripts or session initializations.

Table 76: umask options

Examples:

Display the current umask value:

`umask`

Set a new umask value using octal notation:

`umask 077`

This command configures the shell so that new files and directories are created with permissions allowing only the owner to read, write, and execute them, while preventing group and others from any access.

Display the current umask in a symbolic format:

`umask -S`

Setting the umask is an essential part of system administration and security, as it helps ensure that files and directories are not inadvertently created with overly permissive or restrictive permissions.

This section aims to provide a clear understanding of how the `umask` command functions, its syntax, options, and usage examples to guide users in managing default permissions effectively.

2.3.35 umount

This program can be used to umount file systems.

Synopsis:

```
umount [-a] [-r] [-l] [-f] <file system> | <directory>
```

Options:

Option	Description
`-a`	Unmount all file systems
`-r`	Try to remount devices as read-only if mount is busy
`-l`	Lazy umount (detach filesystem)
`-f`	Force umount (i.e. unreachable NFS server)

Table 77: umount options

Examples:

Disconnecting the disc connected to the directory /mnt.

```
umount /mnt
```

2.3.36 umupdate

This program can be used for adding or deleting of a router app from the command line.

Synopsis:

```
umupdate [-a <filename>] [-d <name>]
```

Options:

Option	Description
`-a`	Add new or update installed router app. Enter path to the installation file.
`-d`	Delete an installed router app with the specified name. List of installed router apps can be obtained by `service module list` command.

Table 78: umupdate options

2.4 Network Commands

Networking commands facilitate the configuration and troubleshooting of network settings. These tools are essential for managing connections, analyzing traffic, and ensuring secure communication over the network.

2.4.1 arp

This program displays and modifies the Internet-to-Ethernet address translation tables used by the address resolution protocol.

Synopsis:

```
arp [-a <hostname>] [-s <hostname> <hw_addr>] [-d <hostname>] [-v] [-n] [-i <if>]
[-D <hostname>] [-A ] [-f <filename>]
```

Options:

Option	Description
-a	The entries will be displayed in alternate (BSD) style.
-s	Manually create an ARP address mapping entry for host hostname with hardware address set to hw_addr.
-d	Remove any entry for the specified host.
-v	Tell the user what is going on by being verbose.
-n	Shows numerical addresses instead of trying to determine symbolic host, port or user names.
-i	Select an interface.
-D	Use the interface ifa's hardware address.
-f	Similar to the *-s* option, only this time the address info is taken from file filename set up. The name of the data file is very often */etc/ethers*, but this is not official. If no filename is specified */etc/ethers* is used as default. The format of the file is simple; it only contains ASCII text lines with a hardware address and a hostname separated by whitespace. Additionally the pub, temp and netmask flags can be used.

Table 79: arp options

With no flags, the program displays the current ARP entry for hostname. The host may be specified by name or by number, using Internet dot notation. For detail description of this command, visit Linux manual pages.

Examples:

View arp table without translating IP addresses to domain names

```
arp -n
```

2.4.2 brctl

The `brctl` command is a legacy utility for managing Ethernet bridges. While retained for backward compatibility, it is considered deprecated. For new configurations, especially those involving VLANs or Distributed Switch Architecture (DSA), it is strongly recommended to use the modern `ip` and `bridge` commands. `brctl` does not support advanced features like VLAN filtering.

This command can be used to set up, maintain, and inspect the Ethernet bridge configuration in the Linux kernel. An Ethernet bridge is a device commonly used to connect different Ethernet networks, making them appear as a single, unified network to all connected devices.

Each connected network segment corresponds to a physical interface added to the bridge. These individual interfaces are bundled into one larger logical network, which corresponds to the bridge network interface itself.

Synopsis:

`brctl [<command>]`

Commands:

Command	Parameters	Description
addbr	<bridge>	Creates a new bridge instance.
delbr	<bridge>	Deletes an existing bridge.
addif	<bridge> <device>	Adds a network interface (port) to a bridge.
delif	<bridge> <device>	Removes a network interface from a bridge.
setageing	<bridge> <time>	Sets the Ethernet address ageing time (in seconds).
setbridgepri	<bridge> <priority>	Sets the bridge's priority for STP (Spanning Tree Protocol).
setfd	<bridge> <time>	Sets the bridge's forward delay (in seconds).
sethello	<bridge> <time>	Sets the time between hello packets for STP.
setmaxage	<bridge> <time>	Sets the maximum message age for STP.
setpathcost	<bridge> <port> <cost>	Sets the STP cost of a path via a specific port.
setportprio	<bridge> <port> <priority>	Sets the STP priority for a specific port.
show		Displays a list of all current bridge instances.
showmacs	<bridge>	Displays a list of MAC addresses learned by the bridge.
showstp	<bridge>	Displays the STP status for a bridge.
stp	<bridge> {on | off}	Enables or disables the Spanning Tree Protocol on a bridge.

Table 80: brctl commands

Examples:

Create a bridge named `br0` and add interfaces `eth0` and `eth1` to it. This effectively connects the two network segments.

```
brctl addbr br0
brctl addif br0 eth0
brctl addif br0 eth1
```

Display the status of all configured bridges.

```
brctl show
```

2.4.3 bridge

The `bridge` command is the modern, feature-rich utility for managing Ethernet bridges and their advanced features, such as VLAN filtering. It is part of the `iproute2` suite and serves as the replacement for the legacy `brctl` command. For all new configurations, the use of `ip` and `bridge` is strongly recommended.

The `bridge` command allows for the configuration and inspection of bridge devices, ports, and VLAN settings. It works in conjunction with the `ip` command, which is used for creating the bridge device itself and for assigning interfaces to it.

Synopsis:

```
bridge [ OPTIONS ] OBJECT { COMMAND | help }
```

Global Options:

Option	Description
`-V, -Version`	Displays the version of the `bridge` utility.
`-s, -stats`	Displays more detailed statistics.
`-d, -details`	Provides more detailed information in the output.
`-j, -json`	Displays output in JSON format.
`-p, -pretty`	Makes JSON output more human-readable.
`-o, -oneline`	Formats output on a single line, which is useful for scripting.

Table 81: Global bridge options

The `bridge` utility operates on several objects. The most common objects are `link` , `mdb` , and `vlan` .

Object: link

The `link` object is used to inspect and configure bridge ports.

Command	Description
`show`	Displays the status and configuration of all bridge ports.
`set dev <port> <args>`	Sets various parameters for a specific port. Common arguments include: • `cost <value>` : Sets the STP path cost. • `priority <value>` : Sets the STP port priority. • `state <state>` : Sets the port state (e.g., `forwarding`). • `vlan_filtering <0|1>` : Disables or enables VLAN filtering on the bridge device. Must be set on the bridge interface (e.g., `br0`), not a port.

Table 82: bridge link commands

Object: vlan

The `vlan` object is used to configure VLAN filtering on bridge ports, which is a key feature not available in `brctl` . VLAN filtering must first be enabled on the bridge device itself using `bridge link set dev <bridge> vlan_filtering 1` .

Command	Description
`show [dev <port>]`	Displays the VLAN configuration for all ports or for a specific port.
`add vid <ID> dev <port> [pvid] [untagged]`	Adds a VLAN to a port. • `vid <ID>` : The VLAN ID to add. • `pvid` : Marks this VLAN as the Port VLAN ID (native VLAN). Ingress untagged traffic is assigned to this VLAN. • `untagged` : Egress traffic for this VLAN will be sent untagged.
`delete vid <ID> dev <port>`	Removes a VLAN from a port.

Table 83: bridge vlan commands

Examples:

Create a bridge, add ports, and bring it up. This uses the `ip` command, which is standard practice.
`ip link add name br0 type bridge` Create a bridge device named br0
`ip link set dev eth0 master br0` Add eth0 to the bridge
`ip link set dev eth1 master br0` Add eth1 to the bridge
`ip link set dev br0 up` Bring the bridge interface up

Enable VLAN filtering on the bridge and configure VLANs on its ports.
`bridge link set dev br0 vlan_filtering 1` Enable VLAN awareness
`bridge vlan add vid 100 dev eth1` Allow tagged VLAN 100 on eth1
`bridge vlan add vid 200 dev eth2 pvid untagged` Allow untagged VLAN 200 on eth2 (native VLAN)

Display the VLAN configuration for the bridge.
`bridge vlan show`

2.4.4 conntrack

This program is the user interface to the netfilter connection tracking system.

Synopsis:

```
conntrack [commands] [option]
```

Options:

Command	Description
`-L [table] [option]`	List conntrack or expectation table
`-G [table]`	Get conntrack or expectation
`-D [table]`	Delete conntrack or expectation
`-I [table]`	Create a conntrack or expectation
`-U [table]`	Update a conntrack
`-E [table]`	Show events
`-F [table]`	Flush table

Table 84: conntrack comands

Table	Description
`conntrack`	This is the default table. It contains a list of all currently tracked connections through the system.
`expect`	This is the table of expectations. Connection tracking expectations are the mechanism used to "expect" RELATED connections to existing ones.

Table 85: conntrack tables

Option	Description
`-n <ip>`	Source NAT ip
`-g <ip>`	Destination NAT ip
`-m <mark>`	Set mark
`-e <eventmask>`	Event mask, eg. NEW,DESTROY
`-z`	Zero counters while listing
`-o <type[...]>`	Output format, eg. xml

Table 86: conntrack options

Option	Description
`--tuple-src <ip>`	Source address in expect tuple
`--tuple-dst <ip>`	Destination address in expect tuple
`--mask-src <ip>`	Source mask address
`--mask-dst <ip>`	Destination mask address

Table 87: expectation options

Option	Description
`-s <ip>`	Source address from original direction
`-d <ip>`	Destination address from original direction
`-r <ip>`	Source addres from reply direction
`-q <ip>`	Destination address from reply direction
`-p <proto>`	Layer 4 Protocol, eg. 'tcp'
`-f <proto>`	Layer 3 Protocol, eg. 'ipv6'
`-t <timeout>`	Set timeout
`-u <status>`	Set status, eg. ASSURED

Table 88: conntrack and expectation options

Examples:

Display content of conntrack table.
`conntrack -L`

Delete content of contrack table.
`conntrack -F`

2.4.5 curl

Curl (transfer a URL) is a tool to transfer data from or to a server, using one of the supported protocols (DICT, FILE, FTP, FTPS, GOPHER, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, POP3, POP3S, RTMP, RTSP, SCP, SFTP, SMTP, SMTPS, TELNET and TFTP). It is an alternative to `wget` (see Chapter 2.4.37).

Synopsis:

`curl [options...] <url>`

Options:

Type `curl ---help` for options to show in the command line or visit online manual page at *curl.1 the man page*.

2.4.6 dhcrelay

The Dynamic Host Configuration Protocol (DHCP) Relay Agent, dhcrelay, provides a means for relaying DHCP and BOOTP requests from a subnet to which no DHCP server is directly connected to one or more DHCP servers on other subnets. It supports both DHCPv4/BOOTP and DHCPv6 protocols (v3 routers only).

Synopsis:

```
dhcrelay [-4] [-d] [-q] [-a] [-D] [-A <length>] [-c <hops>] [-p <port>]
[-pf <pid-file>] [--no-pid] [-m append|replace|forward|discard]
[-i interface0 [ ...  -i interfaceN] server0 [ ...  serverN]

dhcrelay -6 [-d] [-q] [-I] [-c <hops>] [-p <port>] [-pf <pid-file>] [--no-pid]
-l lower0 [ ...  -l lowerN] -u upper0 [ ...  -u upperN]
```

Options:

Option	Description
-a	Append an agent option field to each request before forwarding it to the server. Agent option fields in responses sent from servers to clients will be stripped before forwarding such responses back to the client.
-A <length>	Specify the maximum packet size to send to a DHCPv4/BOOTP server. This might be done to allow sufficient space for addition of relay agent options while still fitting into the Ethernet MTU size.
-D	Drop packets from upstream servers if they contain Relay Agent Information options that indicate they were generated in response to a query that came via a different relay agent.
-i <ifname>	Listen for DHCPv4/BOOTP queries on interface ifname. Multiple interfaces may be specified by using more than one -i option. If no interfaces are specified on the command line, dhcrelay will identify all network interfaces, eliminating non-broadcast interfaces if possible, and attempt to listen on all of them.
-m <option>	Control the handling of incoming DHCPv4 packets which already contain relay agent options.

Table 89: dhcrelay options available in DHCPv4 mode only

Option	Description
-c <hops>	Maximum hop count. When forwarding packets, dhcrelay discards packets which have reached a hop count of <hops>. Default is 10. Maximum is 255.
-d	Force dhcrelay to run as a foreground process.
-p <port>	Listen and transmit on port <port>. Default is port 67 for DHCPv4, or port 547 for DHCPv6.
-q	Quiet mode.

Table 90: dhcrelay options available for both DHCPv4 and DHCPv6

Option	Description
`-I`	Force use of the DHCPv6 Interface-ID option. This option is automatically sent when there are two or more downstream interfaces in use, to disambiguate between them.
`-l`	Specifies the ”lower” network interface for DHCPv6 relay mode: the interface on which queries will be received from clients or from other relay agents.
`-u`	Specifies the ”upper” network interface for DHCPv6 relay mode: the interface to which queries from clients and other relay agents should be forwarded.

Table 91: dhcrelay options available in DHCPv6 mode only:

For help in command line type `dhcrelay -h` . For more information on this command, look up the man page on the Internet.

It is necessary to set at least **two interfaces** for sending/listening – see an Example below.

Examples:

The DHCP Relay Agent listens for DHCPv4 or DHCPv6 queries from clients on interfaces eth0 and eth1, passing them along to ”upstream” server <server ip>. When a reply is received from upstream, it is multicast or unicast back downstream to the source of the original request.

```
dhcrelay -i eth1 -i eth0 <server ip>
```

2.4.7 ebtables

This program can be used as an administration tool for firewall IP packets filtering. It enables transparent filtering of network traffic passing through a Linux bridge. The filtering possibilities are limited to link layer filtering and some basic filtering on higher network layers. Advanced logging, MAC DNAT/SNAT and brouter facilities are also included.

The `ebtables` tool can be combined with the other filtering tools (`iptables` and `ip6tables`) to make a bridging firewall that is also capable of filtering these higher network layers.

Synopsis:

```
ebtables -[ADI] chain rule-specification [options] ebtables -P chain target ebtables -[LFZ] [chain] ebtables -[NX] [chain] ebtables -E old-chain-name new-chain-name
```

Commands:

Option	Description
`-append -A chain`	append to chain
`-delete -D chain`	delete matching rule from chain
`-delete -D chain rulenum`	delete rule at position rulenum from chain
`-change-counters -C chain [rulenum] pcnt bcnt`	change counters of existing rule
`-insert -I chain rulenum`	insert rule at position rulenum in chain
`-list -L [chain]`	list the rules in a chain or in all chains
`-flush -F [chain]`	delete all rules in chain or in all chains
`-init-table`	replace the kernel table with the initial table
`-zero -Z [chain]`	put counters on zero in chain or in all chains
`-policy -P chain target`	change policy on chain to target
`-new-chain -N chain`	create a user defined chain
`-rename-chain -E old new`	rename a chain
`-delete-chain -X [chain]`	delete a user defined chain
`-atomic-commit`	update the kernel w/t table contained in <FILE>
`-atomic-init`	put the initial kernel table into <FILE>
`-atomic-save`	put the current kernel table into <FILE>
`-atomic-file file`	set <FILE> to file

Table 92: ebtables commands

Options:

Option	Description
`-proto -p [!] proto`	protocol hexadecimal, by name or LENGTH
`-src -s [!] address[/mask]`	source mac address
`-dst -d [!] address[/mask]`	destination mac address
`-in-if -i [!] name[+]`	network input interface name
`-out-if -o [!] name[+]`	network output interface name
`-logical-in [!] name[+]`	logical bridge input interface name
`-logical-out [!] name[+]`	logical bridge output interface name
`-set-counters -c chain pcnt bcnt`	set the counters of the to be added rule
`-modprobe -M program`	try to insert modules using this program
`-concurrent`	use a file lock to support concurrent scripts
`-version -V`	print package version

Table 93: ebtables options

Environment variable:
EBTABLES_ATOMIC_FILE – if set <FILE> (see above) will equal its value

Standard targets: DROP, ACCEPT, RETURN or CONTINUE;
The target can also be a user defined chain.

Supported chains for the filter table:
INPUT FORWARD OUTPUT

2.4.8 email

The program can be used for sending email.

To work properly, this command requires the SMTP service to be configured correctly. Refer to the *Configuration Manual* of your router, chapter *Configuration* → *Services* → *SMTP*.

Synopsis:

```
email -t <to> [-s <subject>] [-m <message>] [-a <attachment>] [-r <retries>]
```

Options:

Option	Description
-t	E-mail address of the recipient
-s	Subject, enter the subject in quotation marks
-m	Message, enter the message in quotation marks
-a	Attachment file of the email
-r	Number of attempts to send the e-mail (default value: 2)

Table 94: email options

Examples:

Send system logs to the address john.doe@email.com.

```
email -t john.doe@email.com -s "System Log" -a /var/log/messages
```

2.4.9 ether-wake

This command sends a Wake-on-LAN (WoL) Magic Packet to a network interface, which can wake up sleeping machines that support this feature. The target machine is identified by its MAC address.

Synopsis:

```
ether-wake [-b] [-i IFACE] [-p aa:bb:cc:dd[:ee:ff]] MAC
```

Options:

Option	Description
`MAC`	The MAC address of the network card of the machine to be woken up. The address must be specified in the format `00:11:22:33:44:55` .
`-b`	Broadcasts the Magic Packet to all devices on the network segment.
`-i IFACE`	Specifies the network interface through which the Magic Packet will be sent. If not specified, the default interface is *eth0*.
`-p PASSWORD`	Appends a four or six-byte password to the Magic Packet for systems that require a SecureON password. The password should be specified in a MAC-like hex format, separated by colons.

Table 95: ether-wake options

Examples:

Wake up a machine with a specific MAC address using the default *eth0* interface.

```
ether-wake 00:11:22:33:44:55
```

Wake up a machine, sending the packet through the *eth1* interface.

```
ether-wake -i eth1 00:11:22:33:44:55
```

Broadcast a Magic Packet to wake up a machine with a specific MAC address.

```
ether-wake -b 00:11:22:33:44:55
```

Wake up a machine that requires a six-byte SecureON password.

```
ether-wake -p 11:22:33:44:55:66 00:11:22:33:44:55
```

2.4.10 ethtool

This command can be used to display or change Ethernet card settings.

Synopsis:

```
ethtool [<option> ...] <devname> [<commands>]
```

Options:

For detail description this command, visit Linux manual pages.

Examples:

View the status of the interface eth0.

```
ethtool eth0
```

Switch interface eth0 to mode 10 Mbit/s, half duplex.

```
ethtool -s eth0 speed 10 duplex half autoneg off
```

Turn on autonegacion on the interface eth0.

```
ethtool -s eth0 autoneg on
```

2.4.11 ftpput

This command facilitates uploading files to an FTP server. It is designed for simplicity and embedded environments, offering essential functionality for transferring files over FTP.

Usage:

`ftpput [OPTIONS] HOST [REMOTE_FILE] LOCAL_FILE`

Description:

This command uploads `LOCAL_FILE` from the local system to the specified `HOST` . The file on the host can be specified with `[REMOTE_FILE]` ; if not provided, the name of the `LOCAL_FILE` is used on the remote server.

Options: Examples:

Option	Description
`-v`	Enables verbose output, providing additional details during the file transfer
process. `-u USER`	Specifies the username for authentication with the FTP server.
`-p PASS`	Specifies the password for authentication with the FTP server.
`-P PORT`	Specifies the port on which to connect to the FTP server. If not provided, the standard FTP port (21) is used.

Table 96: ftpput options

Upload a file to the FTP server without verbose output:

```
ftpput -u username -p password ftp.example.com /remote/path/file.txt /local/path/file.txt
```

Upload a file with verbose output:

```
ftpput -v -u username -p password -P 21 ftp.example.com /remote/file.txt /local/file.txt
```

For more detailed information about using *ftpput* refer to the official BusyBox documentation or the built-in help by executing `ftpput --help` in the terminal.

2.4.12 ifconfig

This command can be used to configure a network interface.

Synopsis:

```
ifconfig [-a] <interface> [<option> ...]
```

Options:

Option	Description
`broadcast <addr.>`	If the address argument is given, set the protocol broadcast address for this interface.
`pointtopoint <ad.>`	This keyword enables the point-to-point mode of an interface, meaning that it is a direct link between two machines with nobody else listening on it.
`netmask <address>`	Set the IP network mask for this interface.
`dstaddr <address>`	Set the remote IP address for a point-to-point link (such as PPP).
`metric <NN>`	This parameter sets the interface metric.
`mtu <NN>`	This parameter sets the Maximum Transfer Unit of an interface.
`trailers`	This flag used to cause a non-standard encapsulation of inet packets on certain link levels.
`arp`	Enable or disable the use of the ARP protocol on this interface.
`allmulti`	Enable or disable all-multicast mode. If selected, all multicast packets on the network will be received by the interface.
`multicast`	Set the multicast flag on the interface. This should not normally be needed as the drivers set the flag correctly them-selves.
`promisc`	Enable or disable the promiscuous mode of the interface. If selected, all packets on the network will be received by the interface.
`txqueuelen <NN>`	Set the length of the transmit queue of the device.
`up | down`	This flag causes the interface to be activated. | This flag causes the driver for this interface to be shut down.

Table 97: ifconfig options

Examples:

View the status of all interfaces.

```
ifconfig
```

Activation of loopback with IP address 127.0.0.1/8.

```
ifconfig lo up
```

Activation of virtual interface eth0:0 with IP address 192.168.2.1/24.

```
ifconfig eth0:0 192.168.2.1 netmask 255.255.255.0 up
```

2.4.13 ip

This command can be used to configure a network interface or show the current configuration. Type `ip --help` for help in the terminal.

The SPECTRE v3 routers support more ip options and commands (options: `-d[etails]` , `-t[imestamp]` , `-b[atch],<filename>` , `-rc[vbuf]` ; objects: `addrlabel` , `ntable` , `tuntap` , `mrule` , `netns` , `l2tp` , `tcp_metrics` , `token`). For information how to use, type `ip <object> help` , for detailed description of all options, visit Linux manual pages or look up them online.

Synopsis:

```
ip [ <options> ] <object> { <command> | help }
```

Options:

Option	Description
`-V[ersion]`	Print the version of the ip utility and exit
`-s[tatistics]`	Output more information. If the option appears twice or more, the amount of information increases.
`-r[esolve]`	use the system's name resolver to print DNS names instead of host addresses
`-f[amily] <family>`	Specifies the protocol family to use. The protocol family identifier can be one of `inet`, `inet6`, `bridge` , `ipx`, `dnet` or `link`.
`-o[neline]`	output each record on a single line, replacing line feeds with the '\' character

Table 98: ip options

Object	Description
`link`	network device
`addr`	protocol (IP or IPv6) address on a device
`route`	routing table entry
`rule`	rule in routing policy database
`neigh`	manage ARP or NDISC cache entries
`tunnel`	tunnel over IP
`maddr`	multicast address
`mroute`	multicast routing cache entry
`monitor`	watch for netlink messages
`xfrm`	manage IPSec policies

Table 99: ip objects

Examples:

View the status of all interfaces.

```
ip link show
```

View the route table.

```
ip route list
```

Add routing networks 192.168.3.0/24 through interface eth0.

```
ip route add 192.168.3.0/24 dev eth0
```

Add routing IP address 192.168.3.1 trough gateway 192.168.1.2.

```
ip route add 192.168.3.1 via 192.168.1.2
```

Add default gateway 192.168.1.2.

```
ip route add default via 192.168.1.2
```

2.4.14 ipcalc

This utility is designed to calculate and display various network settings based on an IP address and optionally a netmask or prefix length. It is a valuable tool for network administrators and anyone needing to quickly derive network configuration details.

Usage:

```
ipcalc [-bnmphs] ADDRESS[/PREFIX] [NETMASK]
```

Description:

`ipcalc` takes an IP address, and optionally a slash notation prefix or a netmask, and calculates the resulting broadcast, network, netmask, and IP prefix. It simplifies network configuration and planning tasks.

Options:

Option	Description
`-b`	Displays the broadcast address for the given IP network.
`-n`	Calculates and displays the network address.
`-m`	Shows the default netmask for the provided IP address.
`-p`	Displays the prefix length for the given IP address/netmask.
`-h`	Resolves and shows the hostname associated with the IP address.
`-s`	Suppresses error messages, making the output cleaner in scripts or batch operations.

Table 100: ipcalc options

Examples:

Calculate the network settings for an IP address with a specific prefix:
`ipcalc -bnmp 192.168.1.100/24`

Determine the default netmask for an IP address:
`ipcalc -m 192.168.1.100`

For further details on using *ipcalc* refer to the BusyBox documentation or use the built-in help with `ipcalc --help`
.

2.4.15 iptables

This program can be used as an administration tool for IP packets filtering and NAT.

Synopsis:

`iptables [<options>]`

Options:

- For detailed description of this command type `iptables -h` or visit iptables manual pages.
- "DSCP" target and "dscp" match extension is supported – it is possible to configure and use QoS based on marked packets.
- "CONNMARK" target and "connmark" match extension is supported – it sets the netfilter mark value associated with a connection unlike MARK target which is used to set the netfilter mark value associated with the packet. With the CONNMARK target you can mark all the packets of a connection or related to a connection with the same mark. Another usefull use of CONNMARK is that you can mark packets using the criteria that only matches with the first packet.
- `string` match extension is supported. This modules matches a given string by using some pattern matching strategy. See man pages for iptables for more information.
- `u32` match extension is supported. This modules This module allows the matching of arbitrary bytes in a packet. See the u32 tutorial page for more information.
- `statistic` module is supported. This module matches packets based on some statistic condition. It supports the "nth" and "random" distinct modes settable with the `---mode` option. Example: `iptables -I INPUT -p icmp -m statistic --mode nth --every 2 -packet 0 -j DROP`. See more in iptables man pages.
- Supported iptables modules can be found in `/proc/net/ip_tables_matches` proc filesystem entry.

Examples:

Redirect incoming TCP connections to port 8080 on IP address 192.168.1.2 and port 80.

```
/sbin/iptables -t nat -A pre_nat -p tcp --dport 8080 -j DNAT --to-destination
               192.168.1.11:80
/sbin/iptables -t mangle -A pre_nat -p tcp --dport 8080 -j ACCEPT
```

Example of using DSCP with iptables:

```
iptables -t mangle -I POSTROUTING -p tcp --dport 81 -j DSCP --set-dscp 0x0a
iptables -t mangle -I POSTROUTING -m dscp --dscp 0x0a -j MARK --set-mark 81
```

2.4.16 iw

This program is used for displaying and manipulating wireless devices and their configuration.

Synopsis:

`iw [options] command`

Options:

For more details see iw manual page.

Command	Description
`-debug`	enable netlink debugging
`-version`	show version

Table 101: iw options

Commands:

For more details see iw manual page.

Command	Description
`dev`	List all devices or specify a device to manipulate.
`link`	Display the status of the current link.
`scan`	Trigger a scan and dump the results.
`station dump`	Show information about all stations.
`interface add`	Add a new virtual interface.
`phy`	Show information about physical devices.

Table 102: iw commands

Examples:

`iw dev wlan0 scan`
This command initiates a scan for wireless networks on the device `wlan0`. It is useful for finding available wireless networks and their details like SSID, signal strength, and security protocols.

`iw dev wlan0 link`
Displays the status of the current wireless link on `wlan0`, including the SSID, signal quality, and other connection details. This is helpful for diagnosing connection issues or verifying the connected network.

`iw dev wlan0 station dump`
Provides detailed information about all stations (devices) currently connected to `wlan0`. This includes data rates, signal strength, and other relevant statistics, useful for network analysis and troubleshooting.

`iw phy phy0 info`
Shows detailed information about the physical device `phy0`, including supported wireless standards, frequencies, and capabilities. This is valuable for understanding the hardware's wireless capabilities.

`iw dev wlan0 interface add wlan1 type monitor`
Adds a new virtual interface `wlan1` of type `monitor` to the device `wlan0`. Monitor mode allows the capture of packets without being associated with a network, useful for network analysis and wireless testing.

2.4.17 nc

The `nc` (`netcat`) program can be used to open a pipe to IP:port.

Synopsis:

`nc [OPTIONS] HOST PORT` : connect
`nc [OPTIONS] -l -p PORT [HOST] [PORT]` : listen

Options:

Option	Description
`-e`	PROG Run PROG after connect (must be last)
`-l`	Listen mode, for inbound connects
`-lk`	With -e, provides persistent server
`-p PORT`	Local port number
`-s ADDR`	Local address
`-w SEC`	Timeout for connects and final net reads
`-i SEC`	Delay interval for lines sent
`-n`	Don't do DNS resolution
`-u`	UDP mode
`-b`	Allow broadcasts
`-v`	Verbose
`-o FILE`	Hex dump traffic
`-z`	Zero-I/O mode (scanning)

Table 103: nc options

Example:

Open a TCP connection to port 42 of 192.168.3.1, using port 31337 as the source port, with a timeout of 5 seconds:

```
nc -p 31337 -w 5 192.168.3.1 42
```

2.4.18 net-snmpinform

This command is designed to send SNMP INFORM messages to a management entity. It supports SNMP versions 1, 2c, and 3, offering a reliable way to notify management systems of significant events.

Synopsis:

```
net-snmpinform [OPTIONS] AGENT TRAP-PARAMETERS
```

Options:

For more details, see the *snmpinform(1) - Linux man page*.

Option	Description
`-v 1|2c|3`	Specifies the SNMP version to use.
`-c COMMUNITY`	Sets the community string (for SNMP v1 and v2c).
`-a PROTOCOL`	Sets the authentication protocol (for SNMP v3).
`-A PASSPHRASE`	Sets the authentication protocol passphrase (for SNMP v3).
`-l LEVEL`	Sets the security level (for SNMP v3).
`-u USER-NAME`	Sets the security name (for SNMP v3).

Table 104: net-snmpinform options

Examples:

`net-snmpinform -v 2c -c public AGENT " 0 .1.3.6.1.6.3.1.1.5.2`
Sends an INFORM message to the SNMP manager specified by AGENT using SNMP version 2c and the community string "public".

`net-snmpinform -v 3 -u myUser -l authPriv -a SHA -A myAuthPass -x DES -X myPrivPass AGENT " 0 .1.3.6.1.6.3.1.1.5.3`
Sends an SNMPv3 INFORM message with authentication (SHA) and encryption (DES), using the specified usernames and passphrases.

`net-snmpinform -v 1 -c public AGENT .1.3.6.1.4.1.8072.2.3.2.1 0 .1.3.6.1.4.1.8072.2.3.2.2 6`
Uses SNMP version 1 to send an INFORM message with specified enterprise OID and trap parameters.

2.4.19 net-snmptrap

This command is used to send SNMP trap messages to a management entity. It supports SNMP versions 1, 2c, and 3.

Synopsis:

```
net-snmptrap [OPTIONS] AGENT TRAP-PARAMETERS
```

Options:

For more details see the *snmptrap(1) - Linux man page*.

Option	Description
`-v 1|2c|3`	Specifies SNMP version to use.
`-c COMMUNITY`	Set the community string (for SNMP v1 and v2c).
`-a PROTOCOL`	Set authentication protocol (for SNMP v3).
`-A PASSPHRASE`	Set authentication protocol pass phrase (for SNMP v3).
`-l LEVEL`	Set security level (for SNMP v3).
`-u USER-NAME`	Set security name (for SNMP v3).
`-C i`	Send an INFORM instead of a TRAP.

Table 105: net-snmptrap options

Examples:

`net-snmptrap -v 2c -c public AGENT " 0 .1.3.6.1.4.1.8072.2.3.0.1 0 0 "`
Sends a test trap to the SNMP manager specified by AGENT using SNMP version 2c and the community string "public".

`net-snmptrap -v 3 -u myUser -l authPriv -a SHA -A myAuthPass -x DES -X myPrivPass AGENT " 0 .1.3.6.1.6.3.1.1.5.1`
Sends an SNMPv3 trap with authentication (SHA) and encryption (DES), using the specified usernames and passphrases.

`net-snmptrap -v 2c -c public -C i AGENT " 0 .1.3.6.1.4.1.8072.2.3.0.2 0 0 "`
Uses the `-C i` option to send an INFORM message instead of a TRAP using SNMP version 2c.

2.4.20 netstat

This program can be used to display the networking information.

Synopsis:

```
netstat [<options>]
```

Options:

Option	Description
-l	display listening server sockets
-a	display all sockets (default: connected)
-e	display other/more information
-n	don't resolve names
-r	display routing table
-t	tcp sockets
-u	udp sockets
-w	raw sockets
-x	unix sockets

Table 106: netstat options

2.4.21 ntpdate

This program can be used to set the system time from NTP server.

Synopsis:

```
ntpdate [-p <probes>] [-t <timeout>] <server>
```

Options:

Option	Description
-p	Specify the number of samples to be acquired from each server as the integer samples, with values from 1 to 8 inclusive.
-t	Specify the maximum time waiting for a server response as the value timeout, in seconds and fraction.

Table 107: ntpdate options

Examples:

Set the system time according to the NTP server time.windows.com.

```
ntpdate time.windows.com
```

2.4.22 ping

This program can be used to send ICMP echo request to network host.

Synopsis:

```
ping [OPTIONS] HOST
```

Options:

Option	Description
-4,-6	Force IP or IPv6 name resolution.
-c CNT	Send only CNT pings.
-s SIZE	Send SIZE data bytes in packets (default = 56).
-i SECS	Interval.
-A	Ping as soon as reply is recevied.
-t TTL	Set TTL.
-I IFACE/IP	Source interface or IP address.
-W SEC	Seconds to wait for the first response (default 10). After all -c CNT packets are sent.
-w SEC	Seconds until ping exits (default:infinite). Can exit earlier with -c CNT.
-q	Quiet mode, only displays output at start and when finished.
-p HEXBYTE	Payload pattern.

Table 108: ping options

Examples:

Send one ICMP packet Echo Request with size 500 B on IP address 10.0.0.1.

```
ping -c 1 -s 500 10.0.0.1
```

2.4.23 ping6

This command is designed for diagnosing IPv6 network connections by sending ICMP ECHO_REQUEST packets to a specified host. It is a useful tool for testing the reachability of hosts on an IPv6 network and measuring the round-trip time for messages sent from the originating host to a destination computer.

Usage:

```
ping6 [OPTIONS] HOST
```

Description:

Sends ICMP ECHO_REQUEST packets to the HOST specified as an argument. By default, *ping6* sends packets until interrupted. If the host is reachable and responding, *ping6* displays the time taken for the round-trip.

Options:

Option	Description
`-c CNT`	Stop after sending `CNT` pings.
`-s SIZE`	Specifies the number of data bytes to be sent.
`-i SECS`	Wait `SECS` seconds between sending each packet.
`-A`	Ping the host as soon as the reply is received.
`-I IFACE/IP`	Use the specified interface or IP address as the source.
`-W SEC`	Time to wait for the first response before timing out.
`-w SEC`	Timeout before *ping6* exits, regardless of how many packets have been sent or received.
`-q`	Operate in quiet mode, only displaying summary lines at startup and completion.
`-p HEXBYTE`	Pattern to use for payload data.

Table 109: ping6 options

Examples:

Ping a host only a certain number of times:

```
ping6 -c 4 fe80::1
```

Ping a host with a specific interval and packet size:

```
ping6 -i 2 -s 120 fe80::1
```

The `ping6` command is essential for network administrators and users who need to verify IPv6 connectivity or diagnose IPv6 network problems.

2.4.24 route

This program can be used to show and manipulate the IP routing table.

Synopsis:

```
route [ -n ] [ -e ] [ -A ] [ add | del | delete ]
```

Options:

Option	Description
-n	Don't resolve names
-e	Display other/more information
-A	Select address family

Table 110: route options

For detail description this command, visit Linux manual pages.

Examples:

View the routing table without translating IP addresses to domain names.

```
route -n
```

Add routing networks 192.168.3.0/24 through eth0.

```
route add -net 192.168.3.0/24 dev eth0
```

Add routing IP addresses 192.168.3.1 through 192.168.1.2 gateway.

```
route add -host 192.168.3.1 gw 192.168.1.2
```

Add default gateway 192.168.1.2

```
route add default gw 192.168.1.2
```

2.4.25 scp

This program can be used for secure file transferring between hosts on a network. It uses *ssh* protocol for data transfer with the same authentication and security.

Synopsis:

```
scp [-12346BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
[-l limit] [-o ssh_option] [-P port] [-S program]
[[user@]host1:]file1 ... [[user@]host2:]file2
```

Options:

Option	Description
`-1`	Forces scp to use protocol 1.
`-2`	Forces scp to use protocol 2.
`-4`	Forces scp to use IPv4 addresses only.
`-6`	Forces scp to use IPv6 addresses only.
`-B`	Selects batch mode (prevents asking for passwords or passphrases).
`-C`	Compression enable. Passes the -C flag to ssh to enable compression.
`-c cipher`	Selects the cipher to use for encrypting the data transfer. This option is directly passed to ssh.
`-F ssh_config`	Specifies an alternative per-user configuration file for ssh. This option is directly passed to ssh.
`-i identity_file`	Selects the file from which the identity (private key) for public key authentication is read. This option is directly passed to ssh.
`-l limit`	Limits the used bandwidth, specified in Kbit/s.
`-o ssh_option`	Can be used to pass options to ssh in the format used in ssh_config.
`-P port`	Specifies the port to connect to on the remote host.
`-p`	Preserves modification times, access times, and modes from the original file.
`-q`	Quiet mode: disables the progress meter as well as warning and diagnostic messages from ssh.
`-r`	Recursively copy entire directories. Note that scp follows symbolic links encountered in the tree traversal.
`-S program`	Name of program to use for the encrypted connection. The program must understand ssh options.
`-v`	Verbose mode. Causes scp and ssh to print debugging messages about their progress.

Table 111: scp options

The scp utility exits 0 on success, and >0 if an error occurs.

Examples:

Copy the file `/etc/version` from remote host `remotehost.edu` to the local host, into subdirectory `myFolder` in user's home directory.

```
scp root@remotehost.edu:/etc/version/myFolder
```

Copy the file `/etc/version` from the local host to remote host `remotehost.edu`, into user's home directory.

```
scp /etc/version root@remotehost.edu:/
```

Copy the directory `/home/user` from the local host to a remote host's `/tmp/bar` directory.

```
scp -r /home/user root@remotehost.edu:/tmp/bar
```

2.4.26 sipcalc

This command is an advanced console-based IP subnet calculator. It is capable of handling both IPv4 and IPv6 addressing schemes. It provides detailed information about network addresses, subnet masks, and more, making it a valuable tool for network administrators and IT professionals.

Usage:

```
sipcalc [OPTIONS]... <[ADDRESS]... [INTERFACE]... | [-]>
```

Global options

Option	Description
`-a, -all`	Display all possible information.
`-d, -resolve`	Enable name resolution for addresses.
`-h, -help`	Show help message and exit.
`-I, -addr-int=INT`	Specify an interface to add.
`-n, -subnets=NUM`	Display NUM extra subnets starting from the current subnet.
`-u, -split-verbose`	Enable verbose output for subnet splitting.
`-v, -version`	Show version information and exit.
`-4, -addr-ipv4=ADDR`	Specify an IPv4 address to add.
`-6, -addr-ipv6=ADDR`	Specify an IPv6 address to add.

Table 112: sipcalc – global options

IPv4 options

Option	Description
`-b, -cidr-bitmap`	Show CIDR bitmap.
`-c, -classful-addr`	Display classful address information.
`-i, -cidr-addr`	Display CIDR address information (default).
`-s, -v4split=MASK`	Split the current network into subnets of MASK size.
`-w, -wildcard`	Display information for a wildcard (inverse mask).
`-x, -classful-bitmap`	Show classful bitmap.

Table 113: sipcalc – IPv4 options

IPv6 Options

Option	Description
`-e, -v4inv6`	Show IPv4 compatible IPv6 information.
`-r, -v6rev`	Generate IPv6 reverse DNS output.
`-S, -v6split=MASK`	Split the current IPv6 network into subnets of MASK size.
`-t, -v6-standard`	Show standard IPv6 address information (default).

Table 114: sipcalc – IPv6 options

Examples:

Calculate and display all information for an IPv4 address:
`sipcalc -a 192.168.1.1/24`

Split an IPv6 network into smaller subnets:
`sipcalc -S 64 2001:db8::/32`

Address and netmask formats are flexible, supporting dotted quad, number of bits, and hex formats. The tool also supports reading arguments from stdin if `-` is used in place of an address or interface name.

For detailed usage and options, refer to the `sipcalc` manual or the help option `-h` .

2.4.27 snmpget

This is an SNMP application that uses the SNMP GET request to query for information on a network entity. One or more object identifiers (OIDs) may be given as arguments on the command line.

Synopsis:
`snmpget [OPTIONS] [-Cf] OID [OID]...`

Options:

Option	Description
`-h, -help`	Display the help.
`-H`	Display configuration file directives understood.
`-v 1|2c|3`	Specifies SNMP version to use.
`-V, -version`	Display package version number.
`-Cf`	If -Cf is not specified, some applications (*snmpdelta*, *snmpget*, *snmpgetnext* and *snmpstatus*) will try to fix errors returned by the agent that you were talking to and resend the request. The only time this is really useful is if you specified a OID that didn't exist in your request and you're using SNMPv1 which requires "all or nothing" kinds of requests.
other	*For other options see the command help.*

Table 115: snmpget options

Examples:

Retrieve the variable system.sysDescr.0 from the host *zeus* using the community string *public*.
`snmpget -c public zeus system.sysDescr.0`

2.4.28 snmpset

This is an SNMP application that uses the SNMP SET request to set information on a network entity. One or more object identifiers (OIDs) must be given as arguments on the command line. A type and a value to be set must accompany each object identifier.

Synopsis:

```
snmpset [OPTIONS] OID TYPE VALUE [OID TYPE VALUE]...
```

Options:

Option	Description
`-h, -help`	Display the help.
`-H`	Display configuration file directives understood.
`-v 1|2c|3`	Specifies SNMP version to use.
`-V, -version`	Display package version number.
other	*For other options see the command help.*

Table 116: snmpset options

The TYPE is a single character, one of:

Character	Description
`i`	integer
`u`	unsigned
`s`	string
`x`	hex string
`d`	decimal string
`n`	nullobj
`o`	objid
`t`	timeticks
`a`	ipaddress
`b`	bits

Table 117: Type options

Examples:

set the variables sysContact.0 and ipForwarding.0:
system.sysContact.0 = STRING: "dpz@noc.rutgers.edu"
ip.ipForwarding.0 = INTEGER: not-forwarding(2)

```
snmpset -c private -v 1 test-hub system.sysContact.0 s dpz@noc.rutgers.edu
 ip.ipforwarding.0 = 2
```

2.4.29 snmptrap

See similar programs `snmpinform` (Chapter 2.4.18) and `snmptrap` (Chapter 2.4.19).

This program can be used to send a SNMP trap.

Synopsis:

```
snmptrap [-c <community>] [-g <generic>] [-s <specific>] <hostname>
[<oid> <type> <value>]
```

Options:

Option	Description
-c	Community
-g	Specifies generic trap types: 0 – coldStart 1 – warmStart 2 – linkDown 3 – linkUp 4 – authenticationFailure 5 – egpNeighborLoss 6 – enterpriseSpecific
-r	Sends MAC address of eth0 interface
-s	Specifies user definition trap types in the enterpriseSpecific

Table 118: snmptrap options

Examples:

Send TRAP with info about the status of a digital input BIN0 to the IP address 192.168.1.2.

```
snmptrap 192.168.1.2 1.3.6.1.4.1.30140.2.3.1.0 u 'io get bin0'
```

Send TRAP "warm start" to the IP address 192.168.1.2.

```
snmptrap -g 1 192.168.1.2
```

2.4.30 ssh

This is a program for logging into a remote machine and for executing commands on a remote machine.

Synopsis:

```
ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface] [-b bind_address]
  [-c cipher_spec] [-D [bind_address:]port] [-E log_file]
  [-e escape_char] [-F configfile] [-I pkcs11] [-i identity_file]
  [-J destination] [-L address] [-l login_name] [-m mac_spec]
  [-O ctl_cmd] [-o option] [-P tag] [-p port] [-Q query_option]
  [-R address] [-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]]
  destination [command [argument ...]]
```

Options:

For more details see ssh(1) — Linux manual page.

Option	Description
`-4`	Forces to use IPv4 addresses only.
`-6`	Forces to use IPv6 addresses only.
`-i identity_file`	Specifies the file from which the identity (private key) for public key authentication is read.
`-l login_name`	Specifies the user to log in as on the remote machine.
`-p port`	Specifies the port to connect to on the remote host.
`-v`	Verbose mode. Causes ssh to print debugging messages about its progress. This is helpful in diagnosing connection, authentication, and configuration problems.
`-A and -a`	These options control the use of SSH agent forwarding, a mechanism for single sign-on.
`-X and -x`	These manage X11 forwarding, enabling the secure transmission of X11 windows from the remote machine to the local machine.
`-L address`	Specifies that connections to the given TCP port or Unix socket on the local (client) host are to be forwarded to the given host and port, or Unix socket, on the remote side.
`-C`	Requests compression of all data. This can speed up transfers over slow networks.
`-F configfile`	Specifies an alternative per-user configuration file.

Table 119: ssh options

Examples:

`ssh -l username remote_host`

This command is used for a basic login to a remote server. Replace `username` with your actual username and `remote\_host` with the address of the remote server. It initiates an SSH session where you are prompted for the password of the specified user account on the remote server.

`ssh -p 2222 username@remote_host`
If the SSH server is listening on a non-standard port (other than the default port 22), the `-p` option allows you to specify the port to connect to.

`ssh -i /path/to/private_key username@remote_host`
In this scenario, the `-i` option allows you to specify a private key to use for authentication, instead of the default key. This is particularly useful when multiple keys are used for different servers or accounts.

`ssh username@remote_host 'command'`
Here, you can execute a single command on the remote server without entering into a full login shell. Replace `command` with the desired command.
For example, `ssh user@server 'ls -l /var/www'` lists the contents of the specified directory on the remote server.

`ssh -L local_port:remote_host:remote_port username@ssh_server`
This example sets up SSH port forwarding. It forwards connections to a local port to a specified port on a remote machine. It's commonly used to securely access a service on the remote server that isn't exposed to the public internet.

2.4.31 tc

The `tc` (traffic control) command in Linux is a powerful tool utilized for managing network bandwidth and handling Quality of Service (QoS). It allows administrators to control the flow of network traffic by defining policies for traffic classification, prioritization, and rate limiting, among other functionalities. This command is essential for optimizing network performance and ensuring that critical network services remain highly available and responsive.

Synopsis:

```
tc [OPTIONS] OBJECT  COMMAND | help
```

Description:

Tc operates on several objects such as qdiscs (queuing disciplines), classes, and filters, each of which plays a vital role in defining traffic control policies. By manipulating these objects, administrators can tailor the network traffic behavior to suit the specific needs of their environment. This includes creating traffic queues, setting up bandwidth limits for different types of traffic, and applying traffic filters to categorize network packets into various classes.

Options:

Option	Description
`-s`	Show detailed information. When used, *tc* displays more detailed information about the specified object.
`-d`	Show raw data. This option is useful for debugging purposes.
`-p`	Pretty print. Formats the output in a more readable form.
`-b`	Batch mode. Allows *tc* to read a series of commands from a file.

Table 120: tc options

Examples:

Show all current qdiscs:

```
tc -s qdisc show
```

Limit the outbound traffic rate on an interface:

```
tc qdisc add dev eth0 root tbf rate 1mbit burst 32kbit latency 400ms
```

For a comprehensive guide on setting up Quality of Service (QoS) with *tc*, refer to our Quality of Service (QoS) Application Note available at https://icr.advantech.com/download/application-notes#quality-service-qos.

This section aims to introduce the `tc` command, highlighting its significance in network traffic management and Quality of Service (QoS) configuration. Through practical examples, users are guided on how to employ *tc* for optimizing network performance, with additional resources available for deeper exploration of QoS setups.

2.4.32 tcpdump

This program can be used to dump traffic on a network.

Synopsis:

```
tcpdump [-AdDefIlLnNOpqRStuUvxX] [-c <count>] [-C <file size>] [-E algo:secret]
[-F <file>] [-i <interface>] [-r <file>] [-s <snaplen>] [-T type] [-w <file>]
[-y <datalinktype>] [expression]
```

Options:

For detail description this command, visit Linux manual pages.

Examples:

View traffic on interface ppp0.

```
tcpdump -n -i ppp0
```

View traffic on interface eth0 except protocol Telnet.

```
tcpdump -n not tcp port 23
```

View UDP traffic on interface eth0.

```
tcpdump -n udp
```

View HTTP traffic on interface eth0.

```
tcpdump -n tcp port 80
```

View all traffic from/to IP address 192.168.1.2.

```
tcpdump -n host 192.168.1.2
```

View traffic from/to IP address 192.168.1.2 except protocol Telnet.

```
tcpdump -n host 192.168.1.2 and not tcp port 23
```

2.4.33 telnet

This program can be used to establish interactive communication with another computer over a network using the TELNET protocol.

Synopsis:

```
telnet <host> [<port>]
```

Examples:

Connect to 192.168.1.2 by protocol Telnet.

```
telnet 192.168.1.2
```

2.4.34 traceroute

This program can be used to track the route to a network host.

Synopsis:

`traceroute [-FIldnrv] [-f <1st_ttl>] [-m <max_ttl>] [-p <port#>] [-q <nqueries>] [-s <src_addr>] [-t <tos>] [-w <wait>] [-g <gateway>] [-i <iface>] [-z <pausemsecs>] host [data size]`

Options:

Item	Description
`-4,-6`	Force IP or IPv6 name resolution
`-F`	Set the don't fragment bit
`-l`	Display the TTL value of the returned packet
`-n`	Print hop addresses numerically rather than symbolically
`-r`	Bypass the normal routing tables and send directly to a host
`-f N`	First number of hops (default 1)
`-m N`	Set the max time-to-live (max number of hops)
`-q N`	Set the number of probes per hop (default is 3)
`-p N`	Set the base UDP port number used in probes (default is 33434)
`-s IP`	Use the following IP address as the source address
`-i IFACE`	Source interface
`-t N`	Set the type-of-service in probe packets to the following value (default 0)
`-w SEC`	Set the time (in seconds) to wait for a response to a probe (default 3 sec)
`-z MSEC`	Wait before each send
`-I`	Use ICMP ECHO instead of UDP datagrams
`-d`	Enable socket level debugging
`-v`	Verbose output

Table 121: traceroute options

2.4.35 traceroute6

This command is a network diagnostic tool included with BusyBox that is designed to trace the path packets take to reach an IPv6 host. By sending packets with incrementally increasing hop limits (TTL, Time-To-Live), *traceroute6* determines the route packets follow to reach the target host. This command is essential for identifying network bottlenecks and routing issues in IPv6 networks.

Synopsis:

```
traceroute6 [-nrv] [-f 1ST_TTL] [-m MAXTTL] [-q PROBES] [-p PORT]
[-t TOS] [-w WAIT_SEC] [-s SRC_IP] [-i IFACE] [-z PAUSE_MSEC] HOST [BYTES]
```

Description:

`traceroute6` utilizes IPv6 packets to trace the network route from the source to the specified HOST. It provides various options to customize the trace, including setting the first and maximum number of hops, the number of probes per hop, and the source IP address or interface.

Options:

Option	Description
`-n`	Do not resolve IP addresses to their domain names, display numeric addresses.
`-r`	Bypass the normal routing tables and send directly to the host.
`-f N`	Set the initial time-to-live (hop limit) to N.
`-m N`	Specify the maximum number of hops (TTL) traceroute6 will probe.
`-q N`	Set the number of probe packets per hop.
`-p N`	Use N as the base UDP port number for probes.
`-s IP`	Use IP as the source address for the outgoing probe packets.
`-i IFACE`	Specify the interface through which traceroute should send packets.
`-t N`	Set the type-of-service in probe packets to N.
`-w SEC`	Set the time to wait for a response to a probe.
`-z MSEC`	Wait MSEC milliseconds between sending each packet.

Table 122: traceroute6 options

Examples:

Trace the route to an IPv6 host without resolving names:

```
traceroute6 -n HOST
```

Trace the route with a specific number of probes per hop:

```
traceroute6 -q 1 HOST
```

Specify a source address and maximum number of hops:

```
traceroute6 -s SRC_IP -m 15 HOST
```

`traceroute6` offers valuable insights into the network path and performance characteristics between the source and an IPv6 destination, aiding in network troubleshooting and analysis.

2.4.36 vconfig

This program can be used to create and remove virtual ethernet devices.

Synopsis:

```
vconfig command [OPTIONS]
```

Options:

Command [OPTIONS]	Description
`add IFACE VLAN_ID`	Creates a vlan-device on IFACE. The resulting vlan-device will be called according to the nameing convention set.
`rem VLAN_NAME`	Removes the named VLAN_NAME.
`set_flag IFACE 0|1 VLAN_QOS`	When 1, ethernet header reorders are turned on. Dumping the device will appear as a common ethernet device without vlans. When 0(default) however, ethernet headers are not reordered, which results in vlan tagged packets when dumping the device. Usually the default gives no problems, but some packet filtering programs might have problems with it.
`set_egress_map VLAN_NAME SKB_PRIO VLAN_QOS`	This flags that outbound packets with a particular skb-priority should be tagged with the particular vlan priority vlan-qos. The default vlan priority is 0.
`set_ingress_map VLAN_NAME SKB_PRIO VLAN_QOS`	RThis flags that inbound packets with the particular vlan priority vlan-qos should be queued with a particular skb-priority. The default skb-priority is 0.
`set_name_type NAME_TYPE`	Sets the way vlan-device names are created. Use vconfig without arguments to see the different formats.

Table 123: vconfig commands and options

Vlan ID 4091 and 4092 are reserved for the system

Examples:

Create VLAN ID 1 on eth0 Ethernet interface.

```
vconfig add eth0 1
```

2.4.37 wget

This program can be used to retrieve files via HTTP or FTP.

Synopsis:

```
wget [-c] [-q] [-O <document file>] [--header 'header:  value'] [-Y on/off] [-P <DIR>] <url>
```

Options:

Option	Description
`--spider`	Only check URL existence: $? is 0 if exists
`-c`	Continue retrieval of aborted transfers
`-q`	Quiet mode – do not print
`-P DIR`	Save to DIR (default .)
`-S`	Show server response
`-T SEC`	Network read timeout is SEC seconds
`-O FILE`	Save to filename ('-' for stdout)
`-o FILE`	Log messages to FILE
`-U STR`	Use STR for User-Agent header
`-Y on/off`	Use proxy ('on' or 'off')

Table 124: wget options

Examples:

Download a file my.cfg from HTTP server with IP address 10.0.0.1.

```
wget http://10.0.0.1/my.cfg
```

2.5 Scripting/Shell Commands

This section covers commands integral to shell scripting and command-line manipulation. These commands are foundational for automating tasks, managing files, and configuring system behavior through scripts.

2.5.1 awk

This program scans each input file for lines that match any of a set of patterns specified literally in program-text or in one or more files specified as -f progfile.

Synopsis:

```
awk [-v] [-F] [-f] ... [<program-text>] [<file> ...]
```

Options:

Option	Description
`-v`	Assign the value *val* to the variable *var*, before execution of the program begins. Such variable values are available to the BEGIN block of an AWK program.
`-F`	Use for the input field separator (the value of the FS predefined variable).
`-f`	Read the AWK program source from the file program-file, instead of from the first command line argument. Multiple -f (or --file) options may be used.

Table 125: awk options

Examples:

Show IP address of Gateway

```
route -n | awk '/^-0\.0\.0\.0/ { print $2 }'
```

2.5.2 break

This command is used within looping constructs in Bash/Shell scripting to exit from the loop prematurely. It breaks out of the nearest enclosing loop, skipping the remaining iterations of the loop.

2.5.3 continue

The `continue` command is used within loops in Bash/Shell scripting to skip the rest of the current loop iteration and continue with the next iteration. This command is particularly useful when a condition is met that requires prematurely proceeding to the next cycle of the loop without executing the remaining commands in the current iteration.

2.5.4 echo

This command prints the strings to standard output.

Synopsis:

```
echo [-n] [-e] [-E] [<string> ...]
```

Options:

Option	Description
`-n`	Do not output the trailing newline
`-e <level>`	Enable interpretation of backslash escapes
`-E <size>`	Disable interpretation of backslash escapes (default)

Table 126: echo options

Examples:

Switch profile to "Standard".

```
echo "PROFILE=" > /etc/settings
reboot
```

Switch profile to "Alternative 1".

```
echo "PROFILE=alt1" > /etc/settings
reboot
```

Send a sequence of bytes 0x41,0x54,0x0D,0x0A to serial line (write data in octal).

```
echo -n -e "\101\124\015\012" > /dev/ttyS0
```

2.5.5 eval

This command is a built-in utility in Bash/Shell scripting environments used to concatenate its arguments into a single command, which is then executed by the shell. This command is particularly useful for constructing commands based on variables or processing complex expressions where commands depend on other commands' outputs or file contents.

2.5.6 exec

This command is used in shell scripting to replace the current shell process with a specified program. Unlike running a program normally, *exec* does not return to the shell once the program completes. Instead, the new program takes over the current process space, maintaining the same process ID (PID).

2.5.7 exit

This command terminates the current shell session or script execution. It allows the script or shell to exit with an optional exit status, which can be used to indicate the success or failure of the script's execution to the calling environment.

2.5.8 export

This command in shell scripting is utilized to set or export environment variables and functions to the current shell and all processes started from it. By marking an environment variable or function to be exported, it becomes available to subprocesses spawned from the shell, allowing those processes to use the variable or function. This command is integral for configuring the shell environment dynamically.

Synopsis:

```
export [NAME['VALUE'] ...]
```

Description:

Without arguments, `export` displays a list of all names that are exported in the shell session. When accompanied by NAME or NAME='VALUE', the command sets the environment variable NAME to VALUE, or exports the NAME if VALUE is omitted. This enables configurations like setting the PATH, defining the home directory, and configuring terminal settings to be inherited by child processes.

Options: This command typically does not include options for its operation but may be used in conjunction with shell-specific flags to modify its behavior or output formatting.

Examples:

Set the HOME environment variable:

```
export HOME='/root'
```

Export the PATH environment variable, appending a new directory:

```
export PATH=$PATH:/usr/local/bin
```

Display exported names:

```
export
```

These examples illustrate setting environment variables that control the shell and subprocesses' behavior, like the home directory, executable search path, and other crucial runtime configurations.

2.5.9 hash

This command in shell scripting is utilized to handle the hash table of commands in memory, which tracks the full path of previously executed commands. This optimizes the shell's performance by avoiding the need to search the $PATH environment variable for command locations on subsequent executions.

Synopsis:

`hash [options] [name ...]`

Description:

When called without arguments, `hash` displays the contents of the hash table, including command names and their associated full paths. Specifying one or more names as arguments adds those commands to the hash table, assuming they can be found in the directories listed in the $PATH environment variable. This command is particularly useful in scripts and sessions where certain commands are executed repeatedly, reducing overall command lookup times.

Options:

Option	Description
`-r`	Resets the hash table, clearing all remembered locations.
`-l`	Displays output in a format that is reusable as shell input.
`-p pathname name`	Defines a pathname for a command name, bypassing $PATH search and hash table lookup.
`-d name`	Removes the specified name from the hash table.
`-t name`	Displays the remembered location of the specified command, without altering the hash table.

Table 127: hash options

Examples:

Display the hash table:

`hash`

Add a command to the hash table:

`hash myscript`

Reset the hash table:

`hash -r`

Specify a pathname for a command:

`hash -p /usr/local/bin/myscript myscript`

Remove a command from the hash table:

`hash -d myscript`

This section aims to elucidate the purpose and usage of the `hash` command, demonstrating how it manages command locations to enhance shell efficiency. Through examples, readers can learn how to view, modify, and reset the hash table according to their needs.

2.5.10 inc

This command is a specialized, proprietary tool designed to output a number that is incremented by one from the given value. This utility can be particularly useful in scripting and automation tasks where numerical adjustments and calculations are required.

Synopsis:

`inc <value>`

Description:

Accepting a numerical value as input, the `inc` command calculates and displays the value increased by one. This command simplifies operations that involve numerical incrementation, streamlining the process of generating sequences or adjusting values dynamically in scripts.

Options: The `inc` command operates directly with the numerical value provided as an argument without the need for additional options. Its simplicity ensures ease of use and integration into larger command sequences or scripts.

Examples:

Increment a given value:

`inc 5`

This example takes the number 5 as input and outputs 6, demonstrating the basic functionality of the `inc` command to increment a number.

Given its proprietary nature, users should refer to specific documentation or support resources for advanced usage, compatibility information, and integration guidance.

2.5.11 let

This command in Linux is a built-in shell command primarily used for evaluating arithmetic expressions. It provides a straightforward method for performing numerical calculations, supporting a wide range of operators similar to those found in traditional programming languages. This command allows for direct manipulation and evaluation of shell variables and expressions within scripts.

Synopsis:

`let "expression"`

Description:

`Let` evaluates each expression argument as an arithmetic expression. Variable names in expressions refer directly to shell variables without needing a dollar sign prefix. Arithmetic expansions performed by *let* allow assignment, various arithmetic operations, conditional expressions, and even bitwise operations, making it versatile for scripting applications where numerical computation is required.

Options: The *let* command does not have specific options. However, expressions need to be quoted to prevent the shell from interpreting special characters.

Examples:

Increment a variable:

```
count=5
let "count += 1"
echo $count  # Outputs: 6
```

Perform arithmetic with variables:

```
a=10
b=20
let "c = a * b"
echo $c  # Outputs: 200
```

Use conditional expressions:

```
let "d = (a > b) ? a : b"
echo $d  # Outputs: 20
```

These examples illustrate the command's capability to handle arithmetic operations, providing a powerful tool for numerical calculations within shell scripts. The versatility of `let` supports complex expressions, making it a valuable resource for script authors seeking to include mathematical logic and operations in their work.

2.5.12 local

This command is a shell built-in that is used within functions to declare variables as having a function-local scope. It prevents variable names from conflicting with variables outside the function, making shell scripts more reliable and modular.

2.5.13 nohup

`nohup` is short for “No Hang-up”. Nohup is a supplemental command that tells the system not to stop another command once it has started. That means it’ll keep running until it’s done, even if the user that started it logs out.

Synopsis:

```
nohup <program> [<arguments>]
```

Options:

Option	Description
`program`	Program to be run immune to hang-ups with output to a non-tty.
`arguments`	Arguments for the program.

Table 128: nohup options

2.5.14 printf

This command formats and prints ARGUMENT(s) according to FORMAT, where FORMAT controls the output exactly as in C printf. For detailed information please see documentation for C programming language.

Synopsis:

```
printf FORMAT [ARGUMENT...]
```

Format options:

Option	Description
d or i	Signed decimal integer
u	Unsigned decimal integer
o	Unsigned octal
x	Unsigned hexadecimal integer
X	Unsigned hexadecimal integer (uppercase)
f	Decimal floating point, lowercase
e	Scientific notation (mantissa/exponent), lowercase
E	Scientific notation (mantissa/exponent), uppercase
g	Use the shortest representation: %e or %f
G	Use the shortest representation: %E or %F
a	Hexadecimal floating point, lowercase
A	Hexadecimal floating point, uppercase
c	Character
s	String of characters
p	Pointer address
n	Nothing printed
%	A % followed by another % character will write a single % to the stream.

Table 129: printf format specifiers

Examples:

Print number 10 in unsigned hexadecimal integer (uppercase) format.

```
printf "Output: %X \n" 10
```

```
Output: A
```

Example of printing system variables.

```
printf "User '%s' in directory '%s'.\n" "$USER" "$PWD"
```

```
User 'root' in directory '/home/httpd'.
```

2.5.15 read

This command in shell scripting is utilized to read a line of input from standard input (stdin) or a file descriptor. It is frequently employed in scripts to capture user input or to process text files or streams line by line.

2.5.16 readonly

This command in shell scripting is used to mark variables and functions as immutable. Once a variable or a function is set to read-only, its value or function body cannot be changed or unset. Attempting to modify a read-only variable or function will result in an error.

2.5.17 return

This command is utilized within shell functions to terminate the function execution and optionally return an exit status to the calling environment. This command is similar in behavior to the *exit* command but is specifically designed for use within functions.

2.5.18 shlock

This command is a proprietary utility designed to lock a specified file during the execution of a script. This mechanism ensures that the file is not accessed or modified by other processes until the script completes its execution. If the file is already locked, *shlock* will wait until the lock is released before proceeding.

Synopsis:

`shlock <filename>`

Description:

Upon invocation, `shlock` attempts to create a lock on the specified file, preventing other instances or processes from modifying the file concurrently. This command is particularly useful in scripts that perform critical operations on files, requiring exclusive access to prevent data corruption or loss. If a lock cannot be immediately acquired due to an existing lock on the file, `shlock` enters a wait state until the lock becomes available.

Options: This command operates straightforwardly with the primary argument being the path to the file to be locked. There are no additional options, reflecting its focused purpose.

Examples:

Lock a file for exclusive script access:

`shlock /path/to/myfile`

This example demonstrates the basic usage of the `shlock` command to lock a file named `myfile`. The script that executed this command will have exclusive access to `myfile` until the script terminates, at which point the lock is released.

Given its proprietary nature, users are advised to consult specific documentation or support resources for detailed usage instructions, compatibility information, and advanced features.

2.5.19 set

This command is a shell built-in that sets or unsets shell options and positional parameters. It can modify the operational behavior of the shell, set positional parameters to the script, and enable or disable shell features.

2.5.20 shift

This command in shell scripting is used to shift the positional parameters to the left by a specified number of positions, which effectively decreases the number of positional parameters. This is particularly useful in scripts that process an arbitrary number of arguments in a loop.

2.5.21 sleep

This program can be used to delay for a specified amount of time.

Synopsis:

```
sleep <time>
```

Examples:

Sleep for 30 second.

```
sleep 30
```

2.5.22 source

This command in Unix-like operating systems is a shell built-in command used to read and execute commands from a specified file in the current shell environment. This command is commonly used to apply configuration changes, set environment variables, or define functions without starting a new shell session. The `source` command ensures that any variables or functions declared in the file are available in the current shell session.

Synopsis:

```
source filename [arguments]
```

Description:

The `source` command reads and executes commands from the given filename argument in the current shell context. If the specified file contains export statements, function definitions, or variable assignments, they will be applied to the running shell session, affecting its behavior. This command is particularly useful for scripts and configuration files that need to modify the environment of the current shell. Arguments can be passed to the sourced script, which then access them as positional parameters.

Options: The *source* command typically does not have options. Its behavior is straightforward: it executes commands from a file in the current shell.

Examples:

Source a shell script to set environment variables:

```
source /path/to/envvars.sh
```

Execute a script with arguments:

```
source /path/to/script.sh arg1 arg2
```

This documentation aims to clarify the purpose and usage of the `source` command, demonstrating how it is used to execute commands from a file in the current shell session, thereby altering the shell's environment or behavior based on the contents of the file.

2.5.23 test

This command in Linux is a fundamental tool utilized for evaluating conditional expressions. It allows scripts and users to check file types, compare values, and perform logical operations, serving as the backbone for conditional statements in shell scripting. This command facilitates decision-making processes in scripts by testing expressions and returning an exit status based on the evaluation result.

Synopsis:

```
test EXPRESSION
[ EXPRESSION ]
```

Description:

The *test* command evaluates the `EXPRESSION` provided as an argument. If the `EXPRESSION` evaluates to true, *test* returns an exit status of 0 (success). If the `EXPRESSION` evaluates to false, it returns a non-zero exit status (failure). This behavior integrates seamlessly with the if statements in shell scripts, enabling conditional execution of commands.

Commonly Used Tests:

Option	Description
`-f FILE`	Returns true if `FILE` exists and is a regular file.
`-d DIRECTORY`	Returns true if `DIRECTORY` exists and is a directory.
`-z STRING`	Returns true if `STRING` is empty.
`-n STRING`	Returns true if `STRING` is not empty.
`STRING1 = STRING2`	Returns true if the strings are equal.
`STRING1 != STRING2`	Returns true if the strings are not equal.
`-eq, -ne, -lt, -le, -gt, -ge`	Numerical comparisons between integers.

Table 130: Commonly used tests

Examples:

Check if a file exists:

```
test -f /path/to/file && echo "File exists." || echo "File does not exist."
```

Compare two strings:

```
test "string1" = "string2" && echo "Equal" || echo "Not equal"
```

Check if a variable is set:

```
test -n "$VARIABLE" && echo "Variable is set" || echo "Variable is not set"
```

The `test` command's versatility and simplicity make it an indispensable part of shell scripting, providing the means to perform checks and comparisons that guide the flow of execution.

2.5.24 trap

This command in Unix-like operating systems is a shell builtin that allows scripts to execute a command or a set of commands when receiving specified signals. This functionality is crucial for ensuring that scripts can handle unexpected events gracefully, perform cleanup operations, or take specific actions when interrupted.

Synopsis:

`trap [COMMANDS] [SIGNALS]`

Description:

`trap` enables the specification of commands (`COMMANDS`) to be executed automatically in response to various signals (`SIGNALS`). Signals are operating system messages that communicate to processes about events like termination requests, keyboard interrupts, or other conditions that require attention. By default, certain signals cause a script to terminate, but with *trap*, scripts can respond in user-defined ways, allowing for more robust and reliable behavior.

Common Signals:

Option	Description
SIGINT	Interrupt signal, typically sent by pressing Ctrl+C.
SIGTERM	Termination signal, requesting the program to end gracefully.
SIGHUP	Hangup signal, often sent when a terminal window is closed.
SIGQUIT	Quit signal, similar to SIGINT, but also generates a core dump.
EXIT	Not a signal, but can be used with *trap* to specify commands to run when the script exits.

Table 131: Common signals

Examples:

Trap an interrupt signal and execute a cleanup function:

```
trap cleanup_function SIGINT
```

Remove temporary files upon script exit:

```
trap 'rm -f /tmp/mytempfile' EXIT
```

Reset trap to default behavior for SIGINT:

```
trap - SIGINT
```

The `trap` command's ability to catch and respond to signals makes it an invaluable tool in creating more interactive and user-friendly shell scripts. It enhances script reliability and control over the script's execution flow, especially in long-running processes or scripts that manage critical tasks.

This documentation segment aims to elucidate the functionality and utility of the `trap` command in shell scripting, offering a comprehensive overview of its syntax, signal handling capabilities, and practical examples to guide users in implementing effective signal management and cleanup procedures in their scripts.

2.5.25 type

This command is a shell builtin that displays the kind of command the shell will execute, given a command name as an argument. This includes identifying whether the command is a shell builtin, an alias, a function, a keyword, or an external file. The `type` command is an essential tool for debugging and scripting, as it clarifies command resolution and helps script authors understand how their commands will be interpreted by the shell.

2.5.26 unset

This command in shell scripting is used to remove variables or functions from the shell environment. By unsetting a variable or function, you effectively delete it, making its value or definition no longer accessible in the current session. This command is particularly useful in scripting to ensure that the environment is clean or to prevent accidental reuse of variables and functions with stale data.

2.5.27 wait

This command in Unix-like operating systems is a shell builtin that suspends the execution of the shell script until processes identified by their Process ID (PID) have terminated or until a specified job number has completed. It is primarily used in scripts to halt script progress until background processes or jobs have finished executing, making it essential for scripts that rely on the completion of parallel processes.

2.5.28 xargs

This program executes the command on every item given by standard input.

Synopsis:

```
xargs [<commands>] [<options>] [<args> ...]
```

Options:

Option	Description
-r	Do not run command for empty readed lines
-t	Print the command line on stderr before executing it

Table 132: xargs options

Examples:

Find files named core in or below the directory /tmp and delete them. Note that this will work incorrectly if there are any filenames containing newlines or spaces.

```
find /tmp -name core -type f -print | xargs /bin/rm -f
```

3. Related Documents

You can obtain product-related documents on the **Engineering Portal** at *icr.advantech.com*.

To access your router's documents or firmware, go to the *Router Models* page, locate the required model, and select the appropriate tab below.

Documents that are common to all models and describe specific functionality areas are available on the *Application Notes* page.

The **Router Apps** installation packages and manuals are available on the *Router Apps* page.

If you are interested in further options for extending router functionality, either through scripts or custom Router Apps, please see the information available on the *Development* page.

Appendix: Command Index

A

adduser 50
arp 75
awk 116

B

backup 51
basename 25
brctl 76
break 117
bridge 77

C

cat 25
cd 26
cdmaat 4
cdmapwr 5
chdir 26
chmod 52
chown 52
chpasswd 53
clog 54
cmp 26
conntrack 79
continue 117
cp 27
curl 80
cut 28

D

date 54
dd 29
decode 29
deluser 55
df 56
dhcrelay 81
dirname 30
dmesg 56
doas 57

E

ebtables 83
echo 117
email 85
ether-wake 86
ethtool 87
eval 118
exec 118
exit 118
export 118

F

faillock 58
find 30
free 59
ftpput 88
fwupdate 60

G

grep 31
gsmat 7
gsmat2 7
gsminfo 6
gsmpwr 8
gsmpwr2 8
gsmsms 8
gunzip 32
gzip 32

H

hash 119
head 33
hwclock 9

I

id 61
ifconfig 89
inc 120
io 10

ip 90
ipcalc 92
iptables 93
iw 94

J

jq 34

K

kill 62
killall 62
klog 63

L

led 11
less 35
let 121
ln 36
local 122
logger 63
losetup 64
lpm 12
ls 37

M

mac 13
mkdir 38
mount 65
mv 38

N

nc 95
net-snmpinform 96
net-snmptrap 97
netstat 98
nohup 122
ntpdate 98

O

openssl 66

P

passwd 67
pidof 68
ping 99
ping6 100
port1 13
port2 14
portd 15
printf 123
ps 68
pse 17
pwd 39

R

read 124
readlink 39
readonly 124
realpath 40
reboot 18
report 19
restore 69
return 124
rlog 70
rm 40
rmdir 41
route 101

S

scp 102
sed 41
service 70
set 125
shift 125
shlock 124
shred 42
sipcalc 104
sleep 125
slog 70
sms 19
snmpget 105
snmpset 106
snmptrap 107
source 126
split 43
ssh 108
status 20

stty 22
sudo 70
sync 43
sysctl 71

T

tail 44
tar 45
tc 110
tcpdump 111
telnet 111
test 127
times 71
top 72
touch 46
tpm2 23
traceroute 112
traceroute6 113
trap 128
type 129

U

umask 73
umount 74
umupdate 74
unset 129

V

vconfig 114
vi 47

W

wait 129
wc 47
wget 115

X

xargs 129
xbus 24
xxd 48

Z

zcat 49