SPEED TOUCH Pro

CLI Reference Guide

with Firewall DSL Router
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td>5</td>
</tr>
<tr>
<td>1 ADSL Commands</td>
<td>21</td>
</tr>
<tr>
<td>2 ATMF Commands</td>
<td>23</td>
</tr>
<tr>
<td>3 Bridge Commands</td>
<td>31</td>
</tr>
<tr>
<td>4 CIP Commands</td>
<td>47</td>
</tr>
<tr>
<td>5 Config Commands</td>
<td>57</td>
</tr>
<tr>
<td>6 DHCP Commands</td>
<td>63</td>
</tr>
<tr>
<td>7 DNS Commands</td>
<td>101</td>
</tr>
<tr>
<td>8 Firewall Commands</td>
<td>123</td>
</tr>
<tr>
<td>9 IP Commands</td>
<td>149</td>
</tr>
<tr>
<td>10 MER Commands</td>
<td>171</td>
</tr>
<tr>
<td>11 NAT Commands</td>
<td>181</td>
</tr>
<tr>
<td>12 Phonebook Commands</td>
<td>197</td>
</tr>
<tr>
<td>13 PPP Commands</td>
<td>207</td>
</tr>
<tr>
<td>14 PPTP Commands</td>
<td>225</td>
</tr>
<tr>
<td>15 Software Commands</td>
<td>233</td>
</tr>
<tr>
<td>16 SHDSL Commands</td>
<td>239</td>
</tr>
<tr>
<td>17 System Commands</td>
<td>245</td>
</tr>
<tr>
<td>18 TD Commands</td>
<td>253</td>
</tr>
<tr>
<td>Index</td>
<td>257</td>
</tr>
</tbody>
</table>
Preface

Welcome to the Alcatel SpeedTouch™ Pro with Firewall Series Command Line Interface Reference Guide!

This Reference Guide aims to give the fastidious user a concise, practical and easy to use document for configuring the SpeedTouch™ Pro with Firewall via its character based Command Line Interface.

Although the SpeedTouch™ Pro with Firewall Web interface is adequate enough for most users, access via the CLI may be still important for advanced and detailed configuration and troubleshooting.

This CLI Reference Guide covers the CLI commands of the following Alcatel DSL SpeedTouch products:

- Alcatel SpeedTouch™ Pro with Firewall
- Alcatel SpeedTouch™ Pro with Firewall ISDN
- Alcatel SpeedTouch™ Pro with Firewall SHDSL

The Reference Guide consists of three main parts:

- **Part 1 : CLI Navigation**
  This part is meant to make the user familiar with the use and operation of the SpeedTouch™ Pro with Firewall CLI. Next to describing the various access methods to the CLI, this part will describe in brief some general manipulations to navigate through and to perform some operations on the CLI.

- **Part 2 : CLI Command Description**
  This part forms the main part of this Reference Guide. Here all available CLI commands of the SpeedTouch™ Pro with Firewall products are alphabetically described per group selection.
  Each command is described in a systematic manner:
  - The full name of the CLI command (including the group selection)
  - A short description of the CLI command, if needed completed by a description of the possible impact on the user and/or the SpeedTouch™ Pro with Firewall
  - The syntax of the command with a description of each parameter
  - An example to demonstrate the use of the CLI command
  - A list of related CLI commands.

- **Part 3 : CLI Command Index**
  This part allows the user to look up a command alphabetically in its incomplete form.
Preface

► Trademarks

The following trademarks are used in this document:

- Speed Touch™ is a trademark of the Alcatel Company
- Netscape® and Netscape Navigator® are registered trademarks of Netscape Communications Corporation
- Windows™ and Internet Explorer™ are trademarks of Microsoft Corporation
- Apple® and Mac®OS are registered trademarks of Apple Computer Inc.
- UNIX® is a registered trademark of UNIX System Laboratories, Inc.
- Ethernet™ is a trademark of Xerox Corporation.

Other products may be trademarks or registered trademarks of their respective manufacturers.

► Updates

Due to the continuous evolution of the Alcatel DSL technology, existing products are regularly upgraded. Alcatel documentation changes accordingly.

For more information on the newest technological changes and documents, please consult the Alcatel web site at following URL:

  http://www.alcatel.com
  http://www.alcateldsl.com

► Directive

Unless expressly and unambiguously approved by Alcatel, you may not:

- disassemble, de-compile, reverse engineer, trace or otherwise analyze the equipment, its content, operation, or functionality, or otherwise attempt to derive source code (or the underlying ideas, algorithms, structure or organization) from the equipment or from any other information provided by Alcatel, except to the extent that this restriction is expressly prohibited by local law;
- copy, rent, loan, re-sell, sublicense, or otherwise transfer or distribute the equipment to others;
- modify, adapt or create a derivative work of the equipment;
- remove from any copies of the equipment any product identification, copyright or other notices;
- disseminate performance information or analysis (including, without limitation, benchmarks) from any source relating to the equipment.

Such acts not expressly approved by Alcatel will result in the loss of product warranty and will invalidate the user’s authority to operate this equipment.
Alcatel SpeedTouch™ Pro with Firewall

CLI Navigation
Accessing the Command Line Interface

Users can access the Command Line Interface via:

- The *SpeedTouch™ Pro with Firewall* CLI web pages
  This requires that TCP/IP connectivity exists between the host from which the web browser is opened and the *SpeedTouch™ Pro with Firewall*
- A Telnet session
  This requires that TCP/IP connectivity exists between the host from which the Telnet session is opened and the *SpeedTouch™ Pro with Firewall*
- The serial ‘Console’ interface.

Access via the Web Pages

The *SpeedTouch™ Pro with Firewall* CLI is accessible via its web interface. Browse to the *SpeedTouch™ Pro with Firewall* web pages and click in the left frame. As a result the CLI web menu is opened in a new browser window:

You can open the CLI web pages directly by pointing the browser to the following URL: http://10.0.0.138/cli.htm (in which the 10.0.0.138 IP address should be replaced by the actual *SpeedTouch™ Pro with Firewall* IP address if needed).

All CLI groups and commands are placed in a menu. You can open a group by clicking the mark next to a group name, or clicking the group name.

Clicking on a command name will execute it. Commands without parameters are indicated with and are executed immediately.

Commands which require additional parameters are indicated with . After you configured all parameters you must click to execute the command.
Access via a Telnet Session or Serial Console

As soon a session to the CLI is opened, the **SpeedTouch™ Pro with Firewall** banner pops up, followed by the CLI prompt.

In case of a Telnet session authentication via the System password might be required before access is granted.

The following figure shows an example of the **SpeedTouch™ Pro with Firewall** banner after opening a Telnet session and authentication.

**EXAMPLE:**

```bash
/home/doejohn(1)$ telnet 10.0.0.138
Trying 10.0.0.138...
Connected to 10.0.0.138.
Escape character is '^]'.
User: SpeedTouch (00–90–D0–00–01–23–45)
Password: ######
```

---

Alcatel Speed Touch Pro
with Firewall DSL Router
Version R3.4

Copyright (c) 1999–2001 Alcatel

=>
```
Navigation and Manipulation

Manipulation commands are commands that manipulate operations on the command line, for example changing the command group, go to the beginning of the command line, go to the end of the command line, etc.

Command group Navigation

From top level, you can change to a command group by executing the name of the desired command group.

To obtain a list of all available command groups, execute help from the top level.

EXAMPLE:

```
=>help
Following commands are available:
help : Displays this help information
?    : Displays this help information
exit : Exits this shell.
..   : Exits group selection.

Following command groups are available:
dhcp  dns  td  atmf  mer
bridge pptp  ppp  cip  nat
adsl/shdsl qosbook  phonebook  ip  software
system  config  firewall
```

To return to top level, or to descend one level (in case of nested command groups) execute .. .

EXAMPLE:

```
=>phonebook
[phonebook]=>
[phonebook]=>..
=>
```
The Help Command

Execute help from top level to list all available command groups for the SpeedTouch™ Pro with Firewall.

**EXAMPLE:**

```plaintext
=>help
Following commands are available:
help : Displays this help information
? : Displays this help information
exit : Exits this shell.
.. : Exits group selection.

Following command groups are available:
dhcp  dns  td  atm  mer
bridge  pptp  ppp  cip  nat
adsl/shdsl  gosbook  phonebook  ip  software
system  config  firewall
=>
```

You can execute the help command from each command group selection. This results in a list of the available commands (and nested command groups, if available) in this particular command group.

**EXAMPLE:**

```plaintext
=>firewall
[firewall]=>
[firewall]=>help
Following commands are available:
tron : Enables verbose console messaging.
troff : Disables verbose console messaging.
match : Defines an ip packet match.
assign : Assign a chain to an entry point.
list : Shows a list of all the hooks with the chain attached.
flush : Clears all hooks. If a hook is provided, that hook is cleared.

Following command groups are available:
chain  rule
[firewall]=>
```
Executing e.g. **help firewall** from top level gives the same result as executing **help** from the firewall command group selection.

**EXAMPLE:**

```
=>firewall help
Following commands are available :
tron : Enables verbose console messaging.
troff : Disables verbose console messaging.
match : Defines an ip packet match.
assign : Assign a chain to an entry point.
list : Shows a list of all the hooks with the chain attached.
flush : Clears all hooks. If a hook is provided, that hook is cleared.
```

Following command groups are available :

```
chain rule
=>
```

Entering **help** followed by a specific command, e.g. **help firewall assign** (starting from top level) or **help assign** (e.g. on the firewall command group selection) results in a description of the syntax for the command.

**EXAMPLE:**

```
=>help firewall assign
Assign a chain to an entry point.
Syntax : assign hook = <{input|sink|forward|source|output}> chain = <string>
```

```
parameters :
    hook = <{input|sink|forward|source|output}>
     Name of hook to assign chain to.
    chain = <string>
     Name of chain to use.
=>
```
Command Completion

The CLI features command completion, which means that when starting to enter a command it can be completed by pressing the “Tab” key.
For example, entering a at the firewall command group selection, followed by a “Tab” stroke results in the full assign command being completed. Entering firewall a from top level gives the same result.
For the completion to be successful, the part to be added must be unique. Completion works for the command groups, for the commands, for the options, but not for values.

EXAMPLE:

```plaintext
=>firewall
[firewall]=>a “Tab”
[firewall]=>assign
```

Going to the beginning or end of the Command Line

Go to the beginning of the Command Line by pressing “Ctrl+A”; to go to the end of the Command Line press “Ctrl+E”.
In the following example, the first || indicates the position of the cursor after pressing “Ctrl+A”, the second || the position of the cursor after pressing “Ctrl+E”.

EXAMPLE:

```plaintext
=>||list||
```

Breaking off Commands

You can break off a command by pressing “Ctrl+G”. This can be useful in a situation where a user is prompted to enter a value which it does not know and wants to abort the command. Instead of being prompted over and over again for the same value, this allows to break of the command.
In the example below “Ctrl+G” is pressed after the third prompt ‘vpi =’. The command is broken of and the user returns to the command line.

EXAMPLE:

```plaintext
[firewall]=>match
chain =
chain =
chain = “Ctrl+G”
[firewall]=>
```
History of Commands

To retake previous commands press the up arrow "↑" and come back to more recent commands with the down arrow "↓". Press "Enter (.)" to select and execute the retaken command.

EXAMPLE:

```
->firewall
[firewall]=>list
assign hook=input chain=input
assign hook=sink chain=sink
assign hook=forward chain=forward
assign hook=source chain=source
[firewall]=> "↑"
[firewall]=>:firewall list
```
Command Line Interface Top Level Structure

The following command groups are available:

- **adsl** (only applicable for the SpeedTouch™ Pro with Firewall ADSL/POTS variant)
- **atmf** (only applicable for models equipped with an ATMF-25.6Mbpsport)
- **bridge**
- **cip**
- **config**
- **dhcp**
- **dns**
- **firewall**
- **ip**
- **mer**
- **nat**
- **phonebook**
- **ppp**
- **pptp**
- **shdsl** (only applicable for the SpeedTouch™ Pro with Firewall SHDSL variant)
- **software**
- **system**
- **td**
Command Line Interface Commands

All CLI commands are commands that operate on, or configure, the SpeedTouch™ Pro with Firewall.

You can execute these commands from top level, preceded by the name of the command group from which the command should be executed (e.g. firewall list).

You can also execute the commands from the command group itself, using the reduced form of the command (e.g. list at the firewall command group selection).

‘!’ in a command means ‘NOT’, e.g. the ‘[!]syn’ parameter in the firewall rule create command.

EXAMPLE:

```plaintext
=>firewall list
assign hook=input chain=input
assign hook=sink chain=sink
assign hook=forward chain=forward
assign hook=source chain=source
=>firewall
[firewall]=>list
assign hook=input chain=input
assign hook=sink chain=sink
assign hook=forward chain=forward
assign hook=source chain=source
[firewall]=>
```

Instead of entering a completely built-up command with all its parameters, you can also enter just the command itself, without its parameters. After this you are prompted to complete the command with the required and the optional parameters. For the optional parameters you can simply press enter without giving a value.

The example below is the equivalent of ‘firewall assign hook= input chain= input’. To break of such incomplete command press “Ctrl+G”.

EXAMPLE:

```plaintext
=>firewall assign
hook = input
chain = input
=>
```
Alcatel SpeedTouch™ Pro with Firewall

CLI Command Description
1 ADSL Commands

The adsl command group is only applicable to the SpeedTouch™ Pro with Firewall ADSL/POTS variant, NOT to the SpeedTouch™ Pro with Firewall ADSL/ISDN and SpeedTouch™ Pro with Firewall SHDSL variant.

adsl (to access the ADSL level)
adsl info
**adsl info**

Show ADSL/POTS statistics and information about the **SpeedTouch™ Pro with Firewall** status.

**SYNTAX:**

```
adsl info
```

**EXAMPLE:**

```
=> adsl info
Modemstate : up
Operation Mode : G.DMT Annex A  [ POTS Overlay Mode ]
Channel Mode : fast
Number of resets : 1

Vendor (ITU) : Local Remote
Country : 0f 0f
Vendor : ALCB ALCB
VendorSpecific : 0000 0000
StandardRevisionNr: 01 01

<table>
<thead>
<tr>
<th></th>
<th>Downstream</th>
<th>Upstream</th>
</tr>
</thead>
<tbody>
<tr>
<td>Margin [dB]</td>
<td>31</td>
<td>31</td>
</tr>
<tr>
<td>Attenuation [dB]</td>
<td>26</td>
<td>13</td>
</tr>
</tbody>
</table>

Available Bandwidth

<table>
<thead>
<tr>
<th></th>
<th>Cells</th>
<th>Kbits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downstream</td>
<td>2641</td>
<td>1014</td>
</tr>
<tr>
<td>Upstream</td>
<td>301</td>
<td>115</td>
</tr>
</tbody>
</table>

Transfer statistics

<table>
<thead>
<tr>
<th></th>
<th>Cells</th>
<th>Kbits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downstream</td>
<td>185670</td>
<td>71297</td>
</tr>
<tr>
<td>Upstream</td>
<td>10254</td>
<td>3937</td>
</tr>
</tbody>
</table>

Current Connection

<table>
<thead>
<tr>
<th></th>
<th>Cells</th>
<th>Kbits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downstream</td>
<td>185668</td>
<td>71296</td>
</tr>
<tr>
<td>Upstream</td>
<td>N/Avail</td>
<td>N/Avail</td>
</tr>
</tbody>
</table>

Errors

- Received FEC : 0
- Received CRC : 0
- Received HEC : 0

=>
```
2 ATMF Commands

The atmf command group is only applicable to the SpeedTouch™ Pro with Firewall ADSL/POTS variant equipped with an ATMF-25.6Mbps port.

- `atmf` (to access the ATMF level)
- `atmf add`
- `atmf delete`
- `atmf flush`
- `atmf list`
- `atmf load`
- `atmf save`
**atmf add**
Add a cross-connection between the ATMF-25 interface and the WAN interface.

**SYNTAX:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>atmf add</td>
<td>vpi = &lt;number {0-7}&gt;</td>
</tr>
<tr>
<td></td>
<td>vci = &lt;number {0-511}&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>vpi</td>
<td>A number between 0 and 7. Represents the Virtual Path identifier.</td>
<td>REQUIRED</td>
</tr>
<tr>
<td>vci</td>
<td>A number between 0 and 511. Represents the Virtual Channel identifier.</td>
<td>REQUIRED</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

```
=>atmf list
VPI = 1 VCI = 0
VPI = 2 VCI = 0
VPI = 3 VCI = 0
VPI = 4 VCI = 0
VPI = 5 VCI = 0
VPI = 6 VCI = 0
VPI = 7 VCI = 0
=>atmf add vpi=0 vci=35
=>atmf list
VPI = 0 VCI = 35
VPI = 1 VCI = 0
VPI = 2 VCI = 0
VPI = 3 VCI = 0
VPI = 4 VCI = 0
VPI = 5 VCI = 0
VPI = 6 VCI = 0
VPI = 7 VCI = 0
=>
```

**RELATED COMMANDS:**

- **atmf delete**
  Delete a cross-connection on the ATMF-25 interface.
- **atmf list**
  Show current ATMF-25 interface configuration.
**atmf delete**
Delete a cross-connection on the ATMF-25 interface.

**SYNTAX:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
</table>
| atmf delete | vpi = <number {0-7}>  
              vci = <number {0-511}> |          |
| vpi     | A number between 0 and 7.       | REQUIRED |
|         | Represents the Virtual Path identifier. |          |
| vci     | A number between 0 and 511.     | REQUIRED |
|         | Represents the Virtual Channel identifier. |          |
|         | Use VCI=0 for a VP cross-connection. |          |

**EXAMPLE:**

```plaintext
->atmf list
VPI = 0  VCI = 35
VPI = 1  VCI = 0
VPI = 2  VCI = 0
VPI = 3  VCI = 0
VPI = 4  VCI = 0
VPI = 5  VCI = 0
VPI = 6  VCI = 0
VPI = 7  VCI = 0

=>atmf delete vpi=0 vci=35

->atmf list
VPI = 1  VCI = 0
VPI = 2  VCI = 0
VPI = 3  VCI = 0
VPI = 4  VCI = 0
VPI = 5  VCI = 0
VPI = 6  VCI = 0
VPI = 7  VCI = 0

=>
```

**RELATED COMMANDS:**

- **atmf add** Add a cross-connection on the ATMF-25 interface.
- **atmf list** Show current ATMF-25 interface configuration.
**atmf flush**

Flush complete ATMF-25 interface configuration.
The flush command does not impact previously saved configurations.

**SYNTAX:**

```
 atmf flush
```

**EXAMPLE:**

```
=>atmf list
VPI = 1 VCI = 0
VPI = 2 VCI = 0
VPI = 3 VCI = 0
VPI = 4 VCI = 0
VPI = 5 VCI = 0
VPI = 6 VCI = 0
VPI = 7 VCI = 0
=>atmf flush
=>atmf list
=>
```

**RELATED COMMANDS:**

- **atmf load**
  - Load saved or default ATMF-25 interface configuration.

- **atmf save**
  - Save current ATMF-25 interface configuration.
**atmf list**
Show all current ATM-25 interface cross-connections.

**SYNTAX:**
```
atmf list
```

**EXAMPLE OUTPUT:**
```
=> atmf list
VPI =  0  VCI =  35
VPI =  1  VCI =  0
VPI =  2  VCI =  0
VPI =  3  VCI =  0
VPI =  4  VCI =  0
VPI =  5  VCI =  0
VPI =  6  VCI =  0
VPI =  7  VCI =  0
=>
```

**RELATED COMMANDS:**
- **atmf add** Add an ATM-25 interface cross-connection.
- **atmf delete** Delete a cross-connection on the ATM-25 interface.
**atmf load**
Load saved (or default) ATMF-25 interface configuration.
Execute **atmf flush** prior to **atmf load**.

**SYNTAX:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Parameter</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>atmf load</td>
<td>[defaults = &lt;yes</td>
<td>no&gt;]</td>
<td>OPTIONAL</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

```
=>atmf list
VPI = 0  VCI = 35
=>atmf save
=>atmf flush
=>atmf load defaults=yes
=>atmf flush
=>atmf load defaults=no
=>atmf list
VPI = 0  VCI = 35
=>
```

**RELATED COMMANDS:**

- **atmf flush**  
  Flush complete ATMF-25 interface configuration.

- **atmf save**  
  Save current ATMF-25 interface configuration.
**atmf save**

Save current ATMF-25 interface configuration.

**SYNTAX:**

```atmf save```

**EXAMPLE:**

```
=>atmf list
VPI = 0 VCI = 35
VPI = 1 VCI = 0
VPI = 2 VCI = 0
VPI = 3 VCI = 0
VPI = 4 VCI = 0
VPI = 5 VCI = 0
VPI = 6 VCI = 0
VPI = 7 VCI = 0
=>atmf save
=>atmf flush
=>atmf list
=>atmf load
=>atmf list
VPI = 0 VCI = 35
VPI = 1 VCI = 0
VPI = 2 VCI = 0
VPI = 3 VCI = 0
VPI = 4 VCI = 0
VPI = 5 VCI = 0
VPI = 6 VCI = 0
VPI = 7 VCI = 0
=>
```

**RELATED COMMANDS:**

- **atmf flush**  
  Flush complete ATMF-25 interface configuration.

- **atmf load**  
  Load saved or default ATMF-25 interface configuration.
3 Bridge Commands

- bridge (to access the Bridge level)
- bridge config
- bridge flush
- bridge ifadd
- bridge ifattach
- bridge ifconfig
- bridge ifdelete
- bridge ifdetach
- bridge iflist
- bridge load
- bridge macadd
- bridge macdelete
- bridge maclist
- bridge save
**bridge config**
Show/set bridge aging policy.

**SYNTAX:**

```
bridge config [age = <number {10 - 100000}>]
```

- **[age]**
  - A number between 10 and 100000 (seconds).
  - Represents the lifetime of a dynamically learned MAC address.
  - By default the aging timer is 300 seconds.

**EXAMPLE:**

```
=>bridge config
  Aging : 300
=>bridge config age=600
=>bridge config
  Aging : 600
=>
```
**bridge flush**
Flush complete bridging configuration.
The flush command does not impact previously saved configurations.

**SYNTAX:**
```
bridge flush
```

**EXAMPLE:**
```
=>bridge iflist
OBC : Internal
    Connection State: connected Port:OBC PortState:forwarding
    RX bytes: 75783 frames: 572
    TX bytes: 82768372 frames: 341221 dropframes: 0
eth0 : Internal
    Connection State: connected Port:eth0 PortState:forwarding
    RX bytes: 156344216 frames: 5899238
    TX bytes: 75689 frames: 425 dropframes: 5558017
Br1 : dest : Br1
    Retry: 10 QoS: default Encaps: llc/snap Fcs: off
    Connection State: connected Port:wan0 PortState:forwarding
    RX bytes: 75 frames: 12
    TX bytes: 30246 frames: 91 dropframes: 0
Br2 : dest : Br2
    Retry: 10 QoS: default Encaps: llc/snap Fcs: off
    Connection State: connected Port:wan1 PortState:forwarding
    RX bytes: 167356345 frames: 7453312
    TX bytes: 64234246 frames: 2846491 dropframes: 0
=>bridge flush
=>bridge iflist
OBC : Internal
    Connection State: connected Port:OBC PortState:forwarding
    RX bytes: 75783 frames: 572
    TX bytes: 82908667 frames: 341735 dropframes: 0
eth0 : Internal
    Connection State: connected Port:eth0 PortState:forwarding
    RX bytes: 156553257 frames: 5904070
    TX bytes: 75689 frames: 425 dropframes: 5562335
=>
```

**RELATED COMMANDS:**
- **bridge load**
  Load saved or default bridge configuration.
- **bridge save**
  Save current bridge configuration.
**bridge ifadd**
Create a bridge interface.

**SYNTAX:**

| bridge ifadd | **inf** = <string>  
|             | **dest** = <phonebook entry> |

**inf**
The bridge interface name.
If not specified, the destination parameter must be specified. In this case the name of the destination will double as interface name.

**dest**
The destination address for the new interface.
Typically a phonebook entry.

**EXAMPLE:**

```plaintext
=>bridge iflist
OBC : Internal
   Connection State: connected Port:OBC PortState: forwarding
   RX bytes: 75783 frames: 572
   TX bytes: 82768372 frames: 341221 dropframes: 0

eth0 : Internal
   Connection State: connected Port:eth0 PortState: forwarding
   RX bytes: 156344216 frames: 5899238
   TX bytes: 75689 frames: 425 dropframes: 5558017

=>phonebook list
Name  Type  Use  Address
Br1   bridge 0  8.35
Br2   bridge 0  8.36
CIPPVC3 cip 1  8.82
CIPPVC4 cip 1  8.83

=>bridge ifadd inf=NewBridge dest=Br1  
=>bridge iflist
OBC : Internal
   Connection State: connected Port:OBC PortState: forwarding
   RX bytes: 75783 frames: 572
   TX bytes: 82843610 frames: 341554 dropframes: 0

eth0 : Internal
   Connection State: connected Port:eth0 PortState: forwarding
   RX bytes: 156472129 frames: 5903256
   TX bytes: 75689 frames: 425 dropframes: 5561702
NewBridge : dest : Br1
   Retry: 10 QoS: default Encaps: llc/snap Fcs: off
   Connection State: not-connectedPort: (Unassigned) PortState: forwarding
=>
```

**RELATED COMMANDS:**

- **bridge ifattach**: Attach a bridge interface.
- **bridge ifdetach**: Detach a bridge interface.
- **bridge ifdelete**: Delete a bridge interface.
- **bridge ifconfig**: Configure a bridge interface.
- **bridge iflist**: Show current bridge configuration.
Bridge Commands

bridge ifattach
Attach (i.e. connect) a bridge interface.

SYNTAX:

bridge ifattach   intf = <ifname>

intf           The name of the bridge interface to attach. REQUIRED

EXAMPLE:

=> bridge iflist
OBC : Internal
    Connection State: connected Port:OBC PortState:forwarding
    RX bytes: 75783 frames: 572
    TX bytes: 82843610 frames: 341554 dropframes: 0
eth0 : Internal
    Connection State: connected Port:eth0 PortState:forwarding
    RX bytes: 156472129 frames: 5903256
    TX bytes: 75689 frames: 425 dropframes: 5561702
NewBridge : dest : Br1
    Retry: 10 QoS: default Encaps: llc/snap Fcs: off
    Connection State: not-connectedPort: (Unassigned) PortState: forwarding

=> bridge ifattach intf=NewBridge
=> bridge iflist
OBC : Internal
    Connection State: connected Port:OBC PortState:forwarding
    RX bytes: 75783 frames: 572
    TX bytes: 82843610 frames: 341554 dropframes: 0
eth0 : Internal
    Connection State: connected Port:eth0 PortState:forwarding
    RX bytes: 156472129 frames: 5903256
    TX bytes: 75689 frames: 425 dropframes: 5561702
NewBridge : dest : Br1
    Retry: 10 QoS: default Encaps: llc/snap Fcs: off
    Connection State: connected Port:wan0 PortState:forwarding
    RX bytes: 75 frames: 12
    TX bytes: 30246 frames: 91 dropframes: 0

RELATED COMMANDS:

bridge ifadd          Create a bridge interface.
bridge ifdetach       Detach a bridge interface.
bridge ifdelete       Delete a bridge interface.
bridge ifconfig       Configure a bridge interface.
bridge iflist         Show current bridge configuration.
**bridge ifconfig**

Configure a bridge interface.

**SYNTAX:**

<table>
<thead>
<tr>
<th>bridge ifconfig</th>
<th>intf = &lt;ifname&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>[dest = &lt;ifname&gt;]</td>
<td>[qos = &lt;string&gt;]</td>
</tr>
<tr>
<td>[encaps = {llc/snap</td>
<td>vcmux}]</td>
</tr>
<tr>
<td>[portstate = {disabled</td>
<td>learning</td>
</tr>
</tbody>
</table>

- **intf**: The name of the bridge interface to configure. **REQUIRED**
- **[dest]**: The destination for this interface. Typically a phonebook entry. This parameter needs only to be specified in case of an interface created without specified destination. **OPTIONAL**
- **[qos]**: The name of a configured Quality Of Service book entry. This parameter never needs to be specified. **OPTIONAL**
- **[encaps]**: The type of encapsulation to be used for this bridge interface. Choose between: *llc/snap* | *vcmux* **OPTIONAL**
- **[fcs]**: Whether or not to include the Ethernet FCS in the packet header on the WAN side. Choose between: *off* | *on* **OPTIONAL**
- **[portstate]**: The bridge portstate for this interface. Choose between: *disabled* | *learning* | *forwarding* **OPTIONAL**
- **[retry]**: A number between 0 and 65535. Represents the number of WAN connection setup retries before giving up. By default the retry value is 10. **OPTIONAL**
EXAMPLE:

```bash
=>bridge iflist intf=NewBridge
NewBridge : dest : Br1
  Retry: 10  QoS: default  Encaps: llc/snap  Fcs: off
  Connection State: connected  Port: wan0  PortState: forwarding
  RX bytes: 75  frames: 12
  TX bytes: 30246  frames: 91  dropframes: 0

=>bridge ifconfig intf=NewBridge encap=vcmux retry=15
=>bridge iflist intf=NewBridge
NewBridge : dest : Br1
  Retry: 15  QoS: default  Encaps: vcmux  Fcs: off
  Connection State: connected  Port: wan0  PortState: forwarding
  RX bytes: 83  frames: 13
  TX bytes: 30740  frames: 102  dropframes: 0
```

RELATED COMMANDS:

- **bridge ifadd**
  - Create a bridge interface.

- **bridge ifattach**
  - Attach a bridge interface.

- **bridge ifdetach**
  - Detach a bridge interface.

- **bridge ifdelete**
  - Delete a bridge interface.

- **bridge iflist**
  - Show current bridge configuration.
**bridge ifdelete**
Delete a bridge interface.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bridge ifdelete</td>
<td>The name of the interface name to delete.</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

```bash
=>bridge iflist
OBC : Internal
    Connection State: connected  Port:OBC  PortState: forwarding
    RX bytes: 75783   frames: 572
    TX bytes: 82768372 frames: 341221 dropframes: 0
eth0 : Internal
    Connection State: connected  Port:eth0  PortState: forwarding
    RX bytes: 156344216 frames: 5899238
    TX bytes: 75689 frames: 425 dropframes: 5558017
NewBridge : dest : Br1
    Retry: 10  QoS: default  Encaps: llc/snap  Fcs: off
    Connection State: not-connectedPort: (Unassigned)  PortState: forwarding
=>bridge ifdelete intf=NewBridge
=>bridge iflist
OBC : Internal
    Connection State: connected  Port:OBC  PortState: forwarding
    RX bytes: 75783   frames: 572
    TX bytes: 82843610 frames: 341554 dropframes: 0
eth0 : Internal
    Connection State: connected  Port:eth0  PortState: forwarding
    RX bytes: 156472129 frames: 5903256
    TX bytes: 75689 frames: 425 dropframes: 5561702
=>
```

**RELATED COMMANDS:**

- **bridge ifadd**  Create a bridge interface.
- **bridge ifattach**  Attach a bridge interface.
- **bridge ifconfig**  Configure a bridge interface.
- **bridge ifdetach**  Detach a bridge interface.
- **bridge iflist**  Show current bridge configuration.
**bridge ifdetach**

Detach (i.e. disconnect) a bridge interface.

**SYNTAX:**

```
bridge ifdetach  intf = <ifname>
```

- `intf`  The name of the bridge interface to detach.  REQUIRED

**EXAMPLE:**

```bash
=>bridge iflist intf=NewBridge
NewBridge : dest : Br1
   Retry: 10  QoS: default  Encaps: llc/snap  Fcs: off
   Connection State: connected  Port:wan0  PortState: forwarding
   RX bytes: 75  frames: 12
   TX bytes: 30246  frames: 91  dropframes: 0
=>bridge ifattach intf=NewBridge
=>bridge iflist intf=NewBridge
NewBridge : dest : Br1
   Retry: 10  QoS: default  Encaps: llc/snap  Fcs: off
   Connection State: not-connected  Port:(Unassigned)  PortState: forwarding
=>
```

**RELATED COMMANDS:**

- `bridge ifadd`  Create a bridge interface.
- `bridge ifattach`  Attach a bridge interface.
- `bridge ifconfig`  Configure a bridge interface.
- `bridge ifdelete`  Delete a bridge interface.
- `bridge iflist`  Show current bridge configuration.
**bridge iflist**
Show the current state of all or the selected bridge interfaces.

**SYNTAX:**

```
bridge iflist  [intf = <ifname>]
```

- **[intf]** The name of the bridge interface to show the configuration of. OPTIONAL
  If not specified all bridge interfaces are shown.

**EXAMPLE OUTPUT:**

```
=>bridge iflist
  
  OBC: Internal
  Connection State: connected  Port: OBC  PortState: forwarding
  RX bytes: 75783  frames: 572
  TX bytes: 82768372  frames: 341221  dropframes: 0

  eth0: Internal
  Connection State: connected  Port: eth0  PortState: forwarding
  RX bytes: 156344216  frames: 5899238
  TX bytes: 75689  frames: 425  dropframes: 5558017

  NewBridge: dest: Br1
  Retry: 15  QoS: default  Encaps: vcmux  Fcs: off
  Connection State: connected  Port: wan0  PortState: forwarding
  RX bytes: 83  frames: 13
  TX bytes: 30740  frames: 102  dropframes: 0

=>
```

**DESCRIPTION:**

‘RX bytes’ indicates the number of Received bytes, ‘TX bytes’ the number of Transmitted bytes. OBC is short for On Board Controller and indicates the physical bridge port.

**RELATED COMMANDS:**

- **bridge ifadd** Create a bridge interface.
- **bridge ifattach** Attach a created bridge interface.
- **bridge ifconfig** Configure a bridge interface.
- **bridge ifdelete** Delete a bridge interface.
- **bridge ifdetach** Detach a bridge interface.
bridge load

Load saved (or default) bridge configuration.
Execute bridge flush prior to bridge load.

SYNTAX:

```
bridge load [defaults = <yes|no>]
```

**EXAMPLE:**

```
=>bridge iflist
OBC : Internal
    Connection State: connected Port: OBC PortState: forwarding
    RX bytes: 75783 frames: 572
    TX bytes: 82768372 frames: 341221 dropframes: 0
eth0 : Internal
    Connection State: connected Port: eth0 PortState: forwarding
    RX bytes: 156344216 frames: 5899238
    TX bytes: 75689 frames: 425 dropframes: 5558017

=>bridge ifadd intf=Br1 dest=Br1
=>bridge save
=>bridge flush
=>bridge iflist
OBC : Internal
    Connection State: connected Port: OBC PortState: forwarding
    RX bytes: 75783 frames: 572
    TX bytes: 82908667 frames: 341735 dropframes: 0
eth0 : Internal
    Connection State: connected Port: eth0 PortState: forwarding
    RX bytes: 156553257 frames: 5904070
    TX bytes: 75689 frames: 425 dropframes: 5562335

=>bridge load defaults=no
=>bridge iflist
OBC : Internal
    Connection State: connected Port: OBC PortState: forwarding
    RX bytes: 75783 frames: 572
    TX bytes: 82768372 frames: 341221 dropframes: 0
eth0 : Internal
    Connection State: connected Port: eth0 PortState: forwarding
    RX bytes: 156344216 frames: 5899238
    TX bytes: 75689 frames: 425 dropframes: 5558017
Br1 : dest : Br1
    Retry: 10 QoS: default Encaps: llc/snap Fcs: off
    Connection State: connected Port: wan0 PortState: forwarding
    RX bytes: 75 frames: 12
    TX bytes: 30246 frames: 91 dropframes: 0
```

**RELATED COMMANDS:**

- bridge flush: Flush complete bridge configuration.
- bridge save: Save current bridge configuration.
bridge macadd

Add a static MAC address to the filtering database. Allows to manually add static addresses, which should normally be dynamically discovered by the bridge itself.

**SYNTAX:**

<table>
<thead>
<tr>
<th>bridge macadd</th>
<th>intf = &lt;ifname&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>hwaddr = &lt;hardware-address&gt;</td>
</tr>
</tbody>
</table>

- **intf**: The name of the bridge interface to add the MAC address for. **REQUIRED**
- **hwaddr**: The MAC address of the new entry. **REQUIRED**

**EXAMPLE:**

```
=>bridge maclist
00:90:d0:01:02:03 -- static, OBC
ff:ff:ff:ff:ff:ff -- static, OBC
01:80:c2:00:00:00 -- static, OBC
01:80:c2:00:00:01 -- static, OBC
...
01:80:c2:00:00:10 -- static, OBC
00:01:42:5f:7d:81 -- dynamic, eth0, 597 seconds
00:50:8b:31:cc:aa -- dynamic, eth0, 513 seconds
08:00:20:c1:9a:12 -- dynamic, eth0, 600 seconds
...
=>bridge macadd intf=eth0 hwaddr=00:80:9f:01:23:45

=>bridge maclist
00:90:d0:01:02:03 -- static, OBC
ff:ff:ff:ff:ff:ff -- static, OBC
01:80:c2:00:00:00 -- static, OBC
01:80:c2:00:00:01 -- static, OBC
...
01:80:c2:00:00:10 -- static, OBC
00:80:9f:01:23:45 -- permanent, OBC
00:01:42:5f:7d:81 -- dynamic, eth0, 598 seconds
00:50:8b:31:cc:aa -- dynamic, eth0, 379 seconds
08:00:20:c1:9a:12 -- dynamic, eth0, 600 seconds
00:08:c7:c3:5f:fc -- dynamic, eth0, 215 seconds
...
=>
```

**RELATED COMMANDS:**

- **bridge macdelete**: Delete a MAC address entry.
- **bridge maclist**: Show current filtering database.
**bridge macdelete**
Remove a MAC address from the filtering database.

**SYNTAX:**

```
bridge macdelete  hwaddr = <hardware-address>
```

- **hwaddr**  The MAC address of the entry to delete.  REQUIRED

**EXAMPLE:**

```
=>bridge maclist
00:90:d0:01:02:03 -- static, OBC
ff:ff:ff:ff:ff:ff -- static, OBC
01:80:c2:00:00:00 -- static, OBC
01:80:c2:00:00:01 -- static, OBC
...  01:80:c2:00:00:10 -- static, OBC
00:01:42:9f:7d:81 -- permanent, OBC
00:50:8b:31:cc:aa -- dynamic, eth0, 513 seconds
08:00:20:c1:9a:12 -- dynamic, eth0, 600 seconds
...  =>bridge macdelete hwaddr=00:80:9f:01:23:45
=>bridge maclist
00:90:d0:01:02:03 -- static, OBC
ff:ff:ff:ff:ff:ff -- static, OBC
01:80:c2:00:00:00 -- static, OBC
01:80:c2:00:00:01 -- static, OBC
...  01:80:c2:00:00:10 -- static, OBC
00:01:42:9f:7d:81 -- dynamic, eth0, 597 seconds
00:50:8b:31:cc:aa -- dynamic, eth0, 513 seconds
08:00:20:c1:9a:12 -- dynamic, eth0, 600 seconds
...  =>
```

**RELATED COMMANDS:**

- **bridge macadd**  Add a static MAC address entry.
- **bridge maclist**  Show current filtering database.
**bridge maclist**

Show current MAC address filtering database.

**SYNTAX:**

```
bridge maclist
```

**EXAMPLE:**

```
=>bridge maclist
=>bridge maclist
00:90:d0:01:02:03 -- static, OBC
ff:ff:ff:ff:ff:ff -- static, OBC
01:80:c2:00:00:00 -- static, OBC
01:80:c2:00:00:01 -- static, OBC
...
01:80:c2:00:00:10 -- static, OBC
00:80:9f:24:ab:cf -- static, OBC
00:01:42:5f:7d:81 -- dynamic, eth0, 598 seconds
00:50:8b:31:cc:aa -- dynamic, eth0, 379 seconds
08:00:20:c1:9a:12 -- dynamic, eth0, 600 seconds
00:08:c7:c3:5f:fc -- dynamic, eth0, 215 seconds
08:00:20:a8:f4:34 -- dynamic, eth0, 600 seconds
08:00:20:83:b7:26 -- dynamic, eth0, 600 seconds
00:10:83:1b:13:18 -- dynamic, eth0, 599 seconds
...
=>
```

**RELATED COMMANDS:**

- **bridge macadd**
  Add a static MAC address entry.
- **bridge macdelete**
  Delete a MAC address entry.
bridge save
Save current bridge configuration.

SYNTAX:

bridge save

EXAMPLE:

=>bridge iflist
OBC: Internal
   Connection State: connected Port: OBC PortState: forwarding
   RX bytes: 75783 frames: 572
   TX bytes: 82768372 frames: 341221 dropframes: 0
eth0: Internal
   Connection State: connected Port: eth0 PortState: forwarding
   RX bytes: 156344216 frames: 5899238
   TX bytes: 75689 frames: 425 dropframes: 5558017
=>bridge ifadd intf=Br1 dest=Br1
=>bridge save
=>bridge flush
=>bridge iflist
OBC: Internal
   Connection State: connected Port: OBC PortState: forwarding
   RX bytes: 75783 frames: 572
   TX bytes: 82908667 frames: 341735 dropframes: 0
eth0: Internal
   Connection State: connected Port: eth0 PortState: forwarding
   RX bytes: 156553257 frames: 5904070
   TX bytes: 75689 frames: 425 dropframes: 5562335
=>bridge load defaults=no
=>bridge iflist
OBC: Internal
   Connection State: connected Port: OBC PortState: forwarding
   RX bytes: 75783 frames: 572
   TX bytes: 82768372 frames: 341221 dropframes: 0
eth0: Internal
   Connection State: connected Port: eth0 PortState: forwarding
   RX bytes: 156344216 frames: 5899238
   TX bytes: 75689 frames: 425 dropframes: 5558017
Br1: dest: Br1
   Retry: 10 QoS: default Encaps: llc/snap Fcs: off
   Connection State: connected Port: wan0 PortState: forwarding
   RX bytes: 75 frames: 12
   TX bytes: 30246 frames: 91 dropframes: 0
=>

RELATED COMMANDS:

bridge flush  Flush complete bridge configuration.
bridge load   Load saved or default bridge configuration.
4 CIP Commands

cip (to access the CIP level)
cip flush
cip ifadd
cip ifdelete
cip iflist
cip load
cip pvca dd
cip pvcd elete
cip pvclist
cip save
cip flush
Flush complete CIP configuration.
The flush command does not impact previously saved configurations.

SYNTAX:

```
cip flush
```

EXAMPLE:

```
=> cip iflist
cip0
  addr = 172.16.0.5  mask = 255.255.255.0
  UNI address = A0:*.04
  inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
  inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out = 0
=> cip flush
=> cip iflist
=>
```

RELATED COMMANDS:

- **cip load**
  Load saved or default CIP configuration.
- **cip save**
  Save current CIP configuration.


**cip ifadd**  
Create a CIP interface at the local side of the Logical IP Subnet (LIS).

**SYNTAX:**

```
cip ifadd
    addr = <ip-address>
    [netmask = <ip-mask (dotted or cidr)>]
    [uniaddr = <portspec:address[.selector]>]
```

- **addr**: The CIP interface’s local IP address in the LIS. **REQUIRED**
- **netmask**: The LIS’s subnetmask. **OPTIONAL**
- **uniaddr**: The UNI-address/port specification for incoming connections, e.g. ‘A0::.04’: ADSL port, any address, selector 3. **OPTIONAL**

**EXAMPLE:**

```bash
=> cip iflist

<table>
<thead>
<tr>
<th>ip</th>
<th>mask</th>
<th>UNI address</th>
<th>inarp_reqs_in</th>
<th>inarp_repl_in</th>
<th>inarp_inv_in</th>
<th>inarp_reqs_out</th>
<th>inarp_repl_out</th>
<th>inarp_inv_out</th>
</tr>
</thead>
<tbody>
<tr>
<td>cipl</td>
<td>172.16.0.5</td>
<td>A0:*:0.04</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>255.255.255.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

=> cip ifadd addr=172.16.1.1 netmask=255.255.255.0

=> cip iflist

<table>
<thead>
<tr>
<th>ip</th>
<th>mask</th>
<th>UNI address</th>
<th>inarp_reqs_in</th>
<th>inarp_repl_in</th>
<th>inarp_inv_in</th>
<th>inarp_reqs_out</th>
<th>inarp_repl_out</th>
<th>inarp_inv_out</th>
</tr>
</thead>
<tbody>
<tr>
<td>cipl</td>
<td>172.16.0.5</td>
<td>A0:*:0.04</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>255.255.255.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cip0</td>
<td>172.16.1.1</td>
<td>A0:*:0.03</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>255.255.255.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

**RELATED COMMANDS:**

- **cip ifdelete**: Delete a CIP interface.
- **cip ifadd**: Show current CIP configuration.
**cip ifdelete**
Delete a CIP interface at the local side of the Logical IP Subnet (LIS).

**SYNTAX:**
```
cip ifdelete addr = <ip-address>
```

<table>
<thead>
<tr>
<th>addr</th>
<th>The CIP interface's local IP address in the LIS.</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQUIRED</td>
<td></td>
</tr>
</tbody>
</table>

**EXAMPLE:**
```
=> cip iflist
cip0  addr = 172.16.1.1  mask = 255.255.255.0
    UNI address = A0:*.03
    inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
    inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out= 0

cip1  addr = 172.16.0.5  mask = 255.255.255.0
    UNI address = A0:*.04
    inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
    inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out= 0

=> cip ifdelete addr=172.16.1.1
=> cip iflist

cip1  addr = 172.16.0.5  mask = 255.255.255.0
    UNI address = A0:*.04
    inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
    inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out= 0

=>
```

**RELATED COMMANDS:**
- **cip ifadd**  Create a CIP interface.
- **cip iflist**  Show current CIP configuration.
**cip iflist**

Show current CIP configuration.

**SYNTAX:**

cip iflist

**EXAMPLE OUTPUT:**

```
=>cip iflist

cip0  addr = 172.16.1.1  mask = 255.255.255.0
    UNI address = A0:*.03
    inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
    inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out= 0

cip1  addr = 172.16.0.5  mask = 255.255.255.0
    UNI address = A0:*.04
    inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
    inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out= 0

=>
```

**DESCRIPTION:**
inarp_reqs_in/inarp_reqs_out : Incoming/outgoing inverse ARP requests
inarp_repl_in/inarp_repl_out : Incoming/outgoing inverse ARP replies
inarp_inv_in/inarp_inv_out : Incoming/outgoing invalid inverse ARP messages

**EXAMPLE INPUT/OUTPUT: EVOLUTION OF ARP REQUESTS IN A NETWORKED ENVIRONMENT:**

```
=>cip iflist

cip0  addr = 200.200.200.138  mask = 255.255.255.0
    UNI address = A0:*.03
    inarp_reqs_in = 18  inarp_repl_in = 75  inarp_inv_in = 0
    inarp_reqs_out = 18  inarp_repl_out = 75  inarp_inv_out= 0

=>cip iflist

cip0  addr = 200.200.200.138  mask = 255.255.255.0
    UNI address = A0:*.03
    inarp_reqs_in = 22  inarp_repl_in = 75  inarp_inv_in = 0
    inarp_reqs_out = 22  inarp_repl_out = 75  inarp_inv_out= 0

=>cip iflist

cip0  addr = 200.200.200.138  mask = 255.255.255.0
    UNI address = A0:*.03
    inarp_reqs_in = 22  inarp_repl_in = 76  inarp_inv_in = 0
    inarp_reqs_out = 22  inarp_repl_out = 76  inarp_inv_out= 0

=>
```

**RELATED COMMANDS:**
cip ifadd Create a CIP interface.
cip ifdelete Delete a CIP interface.
**cip load**

Load saved (or default) CIP configuration.
Execute **cip flush** prior to **cip load**.

SYNTAX:

```
| cip load | [defaults = <yes/no>] |
```

**defaults**
Load factory defaults (yes) or saved configuration (no).
Not specifying this parameter loads the saved configuration

OPTIONAL

EXAMPLE:

```
=>cip iflist
| cip0 addr = 172.16.1.1  mask = 255.255.255.0
| UNI address = A0:*0.03
| inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
| inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out = 0
| cip1 addr = 172.16.0.5  mask = 255.255.255.0
| UNI address = A0:*0.04
| inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
| inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out = 0

=>cip save
=>cip flush
=>cip iflist
=>**cip load defaults=yes**
=>cip iflist
| cip0 addr = 172.16.1.1  mask = 255.255.255.0
| UNI address = A0:*0.03
| inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
| inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out = 0

=>cip flush
=>cip iflist
=>**cip load defaults=no**
=>cip iflist
| cip0 addr = 172.16.1.1  mask = 255.255.255.0
| UNI address = A0:*0.03
| inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
| inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out = 0
| cip1 addr = 172.16.0.5  mask = 255.255.255.0
| UNI address = A0:*0.04
| inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
| inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out = 0

=>
```

RELATED COMMANDS:

- **cip flush**
  Flush complete CIP configuration.
- **cip save**
  Save current CIP configuration.
**cip pvca**
Create a PVC ARP entry for destinations which are not RFC 1577/RFC2225 compliant.

**SYNTAX:**

```
cip pvca
dest = <phonebookname>
[destaddr = <ip-address>]
[mtu = <number (273–20000)>]
```

- **dest**: The ATM address (hardware address) of the destination host. Typically a phonebook name. **REQUIRED**
- **[destaddr]**: The IP address of the destination host. **OPTIONAL**
- **[mtu]**: A number between 273 and 20000 (bytes). Represents the maximum AAL5 packet size for this connection. By default the mtu is 9180 bytes. **OPTIONAL**

**EXAMPLE:**

```bash
=> phonebook list
Name     Type Use Address
Br1      bridge 1 8.35
Br2      bridge 1 8.36
Br3      bridge 1 8.37
Br4      bridge 0 8.38
RELAY_PPP1 ppp 0 8.48
RELAY_PPP2 ppp 0 8.49
RELAY_PPP3 ppp 0 8.50
RELAY_PPP4 ppp 0 8.51
PPP1     ppp 1 8.64
PPP2     ppp 1 8.65
PPP3     ppp 1 8.66
DHCP_SPOOF ppp 1 8.67
CIPPVC1  cip 0 8.80
CIPPVC2  cip 0 8.81
CIPPVC3  cip 0 8.82
CIPPVC4  cip 0 8.83

=> cip pvca dest CIPPVC1 destaddr 172.16.1.2 mtu 546

=> cip pvclist
CIPPVC1
atmport = 0 vpi = 8 vci = 80 dest_ip = 172.16.1.2
encaps = l1c mtu = 546

=>
```

**RELATED COMMANDS:**

- **cip pvca**: Delete a PVC ARP entry.
- **cip pvclist**: Show current PVC ARP entries.
**cip pvcdelete**
Delete a PVC ARP entry.

**SYNTAX:**

```
cip pvcdelete dest = <phonebookname>
```

- **dest**: Typically a phonebook entry name. REQUIRED
  Represents the ATM address (hardware address) or name of the entry to delete.

**EXAMPLE:**

```
=>cip pvclist
CIPPVC1 atmport = 0 vpi = 8 vci = 80 dest_ip = 172.16.1.2
encaps = llc mtu = 546
=>cip pvcdelete dest=CIPPVC1
=>cip pvclist
=>
```

**RELATED COMMANDS:**

- **cip pvcadd**: Create a PVC ARP entry.
- **cip pvclist**: Show current PVC ARP entries.
**cip pvclist**
Show current PVC ARP entries.

**SYNTAX:**

```
cip pvclist
```

**EXAMPLE OUTPUT:**

```
=>cip pvclist
CIPPVC1 atmport = 0  vpi = 8  vci = 80  dest_ip = 172.16.1.2
   encaps = llc  mtu = 546
=>
```

**EXAMPLE INPUT/OUTPUT IN A NETWORKED ENVIRONMENT:**

```
=>cip iflist
cip0    addr = 200.200.200.138  mask = 255.255.255.0
      UNI address = A0:*.*.03
      inarp_reqs_in = 0  inarp_repl_in = 75  inarp_inv_in = 0
      inarp_reqs_out = 0  inarp_repl_out = 75  inarp_inv_out= 0
=>
cip pvclist
699 atmport = 0  vpi = 6  vci = 99  dest_ip = 172.16.1.3
   encaps = llc  mtu = 9180
8.50 atmport = 0  vpi = 8  vci = 50  dest_ip = 200.200.200.14
   encaps = llc  mtu = 9180
=>
```

**RELATED COMMANDS:**

- **cip pvcdecl**e
  - Delete a PVC ARP entry.
- **cip pvccadd**
  - Create a PVC ARP entry.
**cip save**

Save current CIP configuration.

**SYNTAX:**

```
cip save
```

**EXAMPLE:**

```
=>cip iflist

cip0          addr = 172.16.1.1  mask = 255.255.255.0
               UNI address = A0:*.03
               inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
               inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out= 0

cip1          addr = 172.16.0.5  mask = 255.255.255.0
               UNI address = A0:*.04
               inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
               inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out= 0

=>>cip save

=>>cip flush

=>>cip iflist

=>>cip load defaults=no

=>>cip iflist


cip0          addr = 172.16.1.1  mask = 255.255.255.0
               UNI address = A0:*.03
               inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
               inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out= 0


cip1          addr = 172.16.0.5  mask = 255.255.255.0
               UNI address = A0:*.04
               inarp_reqs_in = 0  inarp_repl_in = 0  inarp_inv_in = 0
               inarp_reqs_out = 0  inarp_repl_out = 0  inarp_inv_out= 0

=>>
```

**RELATED COMMANDS:**

- **cip flush**  
  Flush complete CIP configuration.

- **cip load**  
  Load saved or default CIP configuration.
5 Config Commands

    config (to access the Config level)
    config erase
    config flush
    config load
    config reset
    config save
**config erase**
Physically remove all saved configurations.

**SYNTAX:**
```
  config erase
```

**EXAMPLE:**
```
  => config erase
```

**RELATED COMMANDS:**
- `config flush`  Flush complete runtime configuration.
- `config load`   Load complete saved or default configuration.
- `config reset`  Flush current and optionally restore default configuration.
- `config save`   Save complete runtime configuration.
**config flush**

Flush complete current configuration without affecting saved configurations.

This combines all flush commands: `atmf flush`, `bridge flush`, `cip flush`, `config flush`, `dhcp client flush`, `dhcp server flush`, `dns flush`, `firewall flush`, `firewall rule flush`, `mer flush`, `nat flush`, `phonebook flush`, `ppp flush`, `pptp flush`, `qosbook flush`, `system flush` and optionally `ip flush`.

**SYNTAX:**

```
config flush [flush_ip = <no|yes>]
```

- **[flush_ip]** Keep current IP configuration (yes) or not (no).
  
  Not keeping the IP settings could cause lost IP connectivity in the LAN. By default IP settings are preserved.

**EXAMPLE:**

```
-> ip rtlist
    Destination Source Gateway Intf Mtrc
    10.0.0.0/24 10.0.0.0/24 10.0.0.140 eth0 0
    172.16.0.5/32 0.0.0.0/0 172.16.0.5 cip1 0
    0.0.0.140/32 0.0.0.0/0 10.0.0.140 eth0 0
    127.0.0.1/32 0.0.0.0/0 127.0.0.1 loop 0
    10.0.0.0/24 0.0.0.0/0 10.0.0.140 eth0 0
    172.16.0.0/24 0.0.0.0/0 172.16.0.5 cip1 0

=> config flush flush_ip=no

-> ip rtlist
    Destination Source Gateway Intf Mtrc
    10.0.0.0/24 10.0.0.0/24 10.0.0.140 eth0 0
    10.0.0.140/32 0.0.0.0/0 10.0.0.140 eth0 0
    127.0.0.1/32 0.0.0.0/0 127.0.0.1 loop 0
    10.0.0.0/24 0.0.0.0/0 10.0.0.140 eth0 0

=> config flush flush_ip=yes

######### ALL TCP/IP CONNECTIVITY IS LOST #########
```

**RELATED COMMANDS:**

- **config erase** Physically remove all saved configurations.
- **config load** Load complete saved or default configuration.
- **config reset** Flush current and optionally restore default configuration.
- **config save** Save current runtime configuration.
### config load

Load complete saved or default configuration. Execute config flush prior to config load.
In case the saved configuration is loaded (defaults=no) this combines all load commands: atmf
load, bridge load, cip load, dhcp client load, dhcp server load, dns load, firewall load,
firewall rule load, mer load, nat load, phonebook load, ppp load, pptp load, qosbook
load, system load and optionally ip load.

**SYNTAX:**

| config load | [load_ip = <{no|yes}>] |
|-------------|------------------------|
|             | [defaults = {yes|no}]  |

**[load_ip]**
Load IP settings (yes) or not (no).
Not keeping the IP settings could cause lost IP connectivity in the LAN.

**[defaults]**
Load default configuration (yes) or saved configuration (no).
Not specifying this parameter loads the saved configuration

**EXAMPLE:**

```plaintext
=>ip rtlst
Destination   Source     Gateway    Intf  Mtrc
10.0.0.0/24    10.0.0.0/24 10.0.0.140 eth0 0
172.16.0.5/32  0.0.0.0/0   172.16.0.5 cip1 0
0.0.0.140/32   0.0.0.0/0   10.0.0.140 eth0 0
127.0.0.1/32   0.0.0.0/0   127.0.0.1 loop 0
10.0.0.0/24    0.0.0.0/0   10.0.0.140 eth0 0
172.16.0.24    0.0.0.0/0   172.16.0.5 cip1 1
=>config flush flush_ip=no
=>ip rtlst
Destination   Source     Gateway    Intf  Mtrc
10.0.0.0/24    10.0.0.0/24 10.0.0.140 eth0 0
10.0.0.140/32  0.0.0.0/0   10.0.0.140 eth0 0
127.0.0.1/32   0.0.0.0/0   127.0.0.1 loop 0
10.0.0.0/24    0.0.0.0/0   10.0.0.140 eth0 0
=>config load load_ip=yes
=>ip rtlst
Destination   Source     Gateway    Intf  Mtrc
10.0.0.0/24    10.0.0.0/24 10.0.0.140 eth0 0
172.16.0.5/32  0.0.0.0/0   172.16.0.5 cip1 0
127.0.0.1/32   0.0.0.0/0   127.0.0.1 loop 0
10.0.0.0/24    0.0.0.0/0   10.0.0.140 eth0 0
172.16.0.24    0.0.0.0/0   172.16.0.5 cip1 1
=>
```

**RELATED COMMANDS:**

- **config erase**  
  Physically remove all saved configurations.
- **config flush**  
  Flush complete runtime configuration.
- **config reset**  
  Flush current and optionally restore default configuration.
- **config save**  
  Save current runtime configuration.
**config reset**
Flush current runtime configuration and restore factory default configuration. Optionally the runtime, saved IP configuration can be preserved.

**SYNTAX:**
```
config reset  [keep_ip = <{no|yes}>]
```

- **[keep_ip]** Keep IP settings (yes) or not (no).
  Not keeping the IP settings could cause lost IP connectivity in the LAN.
  OPTIONAL

**EXAMPLE:**
```bash
-> dns list
Domain: business.lan
Nr. Hostname IP Address
0 SpeedTouch *.*.*.*
1 TestHost 10.0.0.140
2 HTTP_Server 10.0.0.8
3 FTP_Server 10.0.0.7
Total Table Size: 73 entries
Amount used: 4 (5%)
-> dns save
=> config reset
-> dns list
Domain: lan
Nr. Hostname IP Address
0 SpeedTouch *.*.*.*
Total Table Size: 73 entries
Amount used: 1 (1%)
=> config flush
=> config load
=> dns list
Domain: business.lan
Nr. Hostname IP Address
0 SpeedTouch *.*.*.*
1 TestHost 10.0.0.140
2 HTTP_Server 10.0.0.8
3 FTP_Server 10.0.0.7
Total Table Size: 73 entries
Amount used: 4 (5%)
->
```

**RELATED COMMANDS:**
- **config erase** Physically remove all saved configurations.
- **config flush** Flush complete current configuration.
- **config load** Load complete saved or default configuration.
- **config save** Save current runtime configuration.
**config save**

Save all existing configurations and modifications entered by the user. This combines all save commands: `atmf save`, `bridge save`, `cip save`, `config save`, `dhcp client save`, `dhcp server save`, `dns save`, `firewall chain save`, `firewall save`, `ip save`, `mer save`, `nat save`, `phonebook save`, `ppp save`, `pptp save`, `qosbook save`, `system save`.

**SYNTAX:**

```
config save
```

**EXAMPLE:**

```
=>config save
=>
```

**RELATED COMMANDS:**

- **config erase**
  - Physically remove all saved configurations.
- **config flush**
  - Flush complete current configuration.
- **config load**
  - Load complete saved or default configuration.
- **config reset**
  - Flush current and optionally restore default configuration.
6 DHCP Commands

dhcp (to access the DHCP level)
dhcp client (to access the DHCP Client level)
dhcp client clear
dhcp client config
dhcp client flush
dhcp client ifadd
dhcp client ifattach
dhcp client ifconfig
dhcp client ifdelete
dhcp client iflist
dhcp client ifrelease
dhcp client ifrenew
dhcp client load
dhcp client save
dhcp client stats
dhcp server (to access the DHCP Server level)
dhcp server add
dhcp server client
dhcp server clrstats
dhcp server config
dhcp server delete
dhcp server flush
dhcp server list
dhcp server load
dhcp server policy
dhcp server save
dhcp server spoof
dhcp server start
dhcp server stats
dhcp server status
dhcp server stop
dhcp server troff
dhcp server tron
**dhcp client clear**
Clear DHCP client statistics.

**SYNTAX:**

```
dhcp client clear
```

**EXAMPLE:**

```bash
=> dhcp client stats
DHCP client statistics:
Corrupted packet recv : 0
OFFERS recv : 0
ACKs recv : 0
NAKs recv : 0
Pure BOOTP REPLIES : 0
Other message types : 0
DISCOVERs sent : 253
REQUESTs sent : 9
DECLINEs sent : 0
RELEASEs sent : 0
INFORMs sent : 0
Number of dynamic interfaces: 1
Memory usage:
Table size of dyn leases: 19, in use: 1, free: 94 %
```

```bash
=> dhcp client clear
```

```bash
=> dhcp client stats
DHCP client statistics:
Corrupted packet recv : 0
OFFERS recv : 0
ACKs recv : 0
NAKs recv : 0
Pure BOOTP REPLIES : 0
Other message types : 0
DISCOVERs sent : 0
REQUESTs sent : 0
DECLINEs sent : 0
RELEASEs sent : 0
INFORMs sent : 0
Number of dynamic interfaces: 1
Memory usage:
Table size of dyn leases: 19, in use: 1, free: 94 %
```
**dhcp client config**

Show/set DHCP client configuration.

**SYNTAX:**

```
dhcp client config [trace = <{off|on}>]
```

<table>
<thead>
<tr>
<th>[trace]</th>
<th>Enable tracing (on) or not (off).</th>
<th>OPTIONAL</th>
</tr>
</thead>
</table>

**EXAMPLE:**

```
=> dhcp client config
tracing:off
=> dhcp client config trace=on
=> dhcp client config
tracing:on
=>
```

**RELATED COMMANDS:**

**dhcp client ifconfig**  Configure a DHCP lease created for a specific interface.
**dhcp client flush**
Flush complete DHCP client configuration and dynamic interfaces.
The flush command does not impact previously saved configurations.

**SYNTAX:**
```
dhcp client flush
```

**EXAMPLE:**
```
=>dhcp client iflist
NewMer : [SELECTING]
    flags = uc
    IP address : 10.0.0.10
    HW address : 0:90:d0:01:47:de
    DHCP server : 255.255.255.255
    hostname : NewLease
    req.leasetime= 10800 s
    trying to get a lease for 8 min, 32 sec
    transmission of DISCOVER in 57 sec
    retransmission timeout: 64
    nbr of retransmissions: 14
Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %
=>dhcp client flush
=>dhcp client iflist
No dynamic interfaces defined.
=>
```

**RELATED COMMANDS:**
- **dhcp client load**  
  Load saved or default DHCP client configuration and dynamic interfaces.
- **dhcp client save**  
  Save current DHCP client configuration and dynamic interfaces.
**dhcp client ifadd**

Create a DHCP lease for a specific interface.

**SYNTAX:**

```
dhcp client ifadd      intf = <interface name>
```

* `intf`  
The name of an existing interface, e.g. created via `mer ifadd`.  
REQUIRED

**EXAMPLE:**

```
=> dhcp client iflist
No dynamic interfaces defined.

=> dhcp client ifadd intf=NewMer

NewMer : [INIT]
  flags= uc
  IP address : 0.0.0.0
  HW address : 00:90:d0:01:47:de
  DHCP server : 255.255.255.255
Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %

=>
```

**RELATED COMMANDS:**

- **dhcp client ifattach**  
  Attach a DHCP lease to an interface.

- **dhcp client ifconfig**  
  Configure a DHCP lease created for a specific interface.

- **dhcp client ifdelete**  
  Delete a dynamic interface.

- **dhcp client iflist**  
  Show all dynamic interfaces.
**dhcp client ifattach**

Attach a DHCP lease to a dynamic interface. Firstly create the interface with the **dhcp client ifadd** command.

**SYNTAX:**

```
dhcp client ifattach intf = <interface name>
```

<table>
<thead>
<tr>
<th>intf</th>
<th>The name of the dynamic interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQUIRED</td>
<td></td>
</tr>
</tbody>
</table>

**EXAMPLE:**

```bash
=>dhcp client iflist
NewMer : [INIT]
    flags = uc
    IP address : 0.0.0.0
    HW address : 00:90:d0:01:47:de
    DHCP server : 255.255.255.255
Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %
=>dhcp client ifattach intf=NewMer
=>dhcp client iflist
NewMer : [SELECTING]
    flags = uc
    IP address : 10.0.0.10
    HW address : 0:90:d0:01:47:de
    DHCP server : 255.255.255.255
    hostname : NewLease
    req.leasetime= 10800 s
    trying to get a lease for 8 min, 32 sec
    transmission of DISCOVER in 57 sec
    retransmission.timeout: 64
    nbr of retransmissions: 14
Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %
=>
```

**RELATED COMMANDS:**

- **dhcp client ifadd** Create a DHCP lease for a specific interface.
- **dhcp client ifconfig** Configure a DHCP lease created for a specific interface.
- **dhcp client ifrelease** Release a lease attached to a dynamic interface.
**dhcp client ifconfig**

Show/set the configuration of DHCP lease created for a specific interface. Execute the **dhcp client ifrelease** command prior to configuring it.

**SYNTAX:**

<table>
<thead>
<tr>
<th>dhcp client ifconfig</th>
<th>intf = &lt;interface name&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[clientid = &lt;client-id&gt;]</td>
</tr>
<tr>
<td></td>
<td>[hostname = &lt;hostname&gt;]</td>
</tr>
<tr>
<td></td>
<td>[addr = &lt;ip-address&gt;]</td>
</tr>
<tr>
<td></td>
<td>[leasetime = &lt;number&gt;]</td>
</tr>
<tr>
<td></td>
<td>[addrtrans = &lt;none</td>
</tr>
<tr>
<td></td>
<td>[dns = &lt;off</td>
</tr>
<tr>
<td></td>
<td>[gateway = &lt;off</td>
</tr>
</tbody>
</table>

- **intf** The name of the dynamic interface to be configured. **REQUIRED**
- **[clientid]** The client identity to be associated with the lease. **OPTIONAL**
- **[hostname]** The host name of the client to be associated with the lease. **OPTIONAL**
- **[addr]** The preferred dynamic IP address. **OPTIONAL**
- **[leasetime]** A number between 0 and 1814400 (seconds). Represents the preferred time the client wants to use an address. By default the leasetime is 7200 seconds (2 hours). Specifying -1 makes the lease permanent. **OPTIONAL**
- **[addrtrans]** Automatically enable address translation for this dynamic interface (pat) or not (none). **OPTIONAL**
- **[dns]** Request (and accept) DNS server IP addresses (on) or not (off). **OPTIONAL**
- **[gateway]** Request (and accept) gateway IP addresses (on) or not (off). **OPTIONAL**
EXAMPLE:

```
=> dhcp client iflist
NewMer   : [INIT]
    flags = uc
    IP address : 0.0.0.0
    HW address : 00:90:d0:01:47:de
    DHCP server : 255.255.255.255
Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %

=> dhcp client ifconfig intf=NewMer hostname=NewLease addr=10.0.0.10 leasetime=10800
=> dhcp client iflist
NewMer   : [INIT]
    flags = uc
    IP address : 10.0.0.10
    HW address : 00:90:d0:01:47:de
    DHCP server : 255.255.255.255
    hostname : NewLease
              req.leasetime= 10800 s
Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %
```

RELATED COMMANDS:

- `dhcp client ifadd` Create a DHCP lease for a specific interface.
- `dhcp client ifdelete` Delete a dynamic interface.
- `dhcp client iflist` Show all dynamic interfaces.
- `dhcp client ifrelease` Release a lease attached to a dynamic interface.
dhcp client ifdelete
Delete a dynamic interface.

SYNTAX:

```
dhcp client ifdelete  inf = <interface name>
```

*inf*  The name of the dynamic interface.  REQUIRED

EXAMPLE:

```bash
=> dhcp client iflist
NewMer : [SELECTING]
    flags = uc
    IP address : 10.0.0.10
    HW address : 0:90:d0:01:47:de
    DHCP server : 255.255.255.255
    hostname : NewLease
    req.leasetime= 10800 s
    trying to get a lease for 8 min, 32 sec
    transmission of DISCOVER in 57 sec
    retransmission timeout: 64
    nbr of retransmissions: 14
Number of leases: 1
Total size of table: 19,  in use: 1, free: 94 %
=> dhcp client ifdelete inf NewMer
=> dhcp client iflist
No dynamic interfaces defined.
=>
```

RELATED COMMANDS:

- **dhcp client ifadd**  Create a DHCP lease for a specific interface.
- **dhcp client ifattach**  Attach a DHCP lease to an interface.
- **dhcp client ifconfig**  Configure a DHCP lease created for a specific interface.
- **dhcp client iflist**  Show all dynamic interfaces.
- **dhcp client ifrelease**  Release a lease attached to a dynamic interface.
**dhcp client iflist**

Show all dynamic interfaces.

**SYNTAX:**

```
dhcp client iflist
```

**EXAMPLE:**

```
=>dhcp client iflist
NewMer : [INIT]
    flags= uc  
    IP address : 0.0.0.0  
    HW address : 0:90:d0:01:47:de  
    DHCP server : 255.255.255.255
Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %
=>
```

**EXAMPLE INPUT/OUTPUT IN A NETWORKED ENVIRONMENT:**

The SpeedTouch™ Pro with Firewall is configured as DHCP client on its Ethernet interface eth0.

```
=>dhcp client iflist
eth0 : [BOUND]
    flags= uc  
    IP address : 10.0.0.3  
    HW address : 00:90:d0:01:47:f1  
    DHCP server : 10.10.1.1  
    lease renewal in  5 days, 1 h, 26 min, 45 sec  
    lease rebind in  8 days, 20 h, 34 min, 15 sec  
    lease expires in 10 days, 2 h, 56 min, 45 sec
Number of leases: 1
Total size of table: 18, in use: 1, free: 94 %
=>dhcp client iflist
eth0 : [BOUND]
    flags= uc  
    IP address : 10.0.0.3  
    HW address : 00:90:d0:01:47:f1  
    DHCP server : 10.10.1.1  
    lease renewal in  5 days, 1 h, 25 min, 27 sec  
    lease rebind in  8 days, 20 h, 32 min, 57 sec  
    lease expires in 10 days, 2 h, 55 min, 27 sec
Number of leases: 1
Total size of table: 18, in use: 1, free: 94 %
=>
```

**RELATED COMMANDS:**

- **dhcp client ifadd**  
  Create a DHCP lease for a specific interface.
- **dhcp client ifdelete**  
  Delete a dynamic interface.
**dhcpp client ifrelease**

Release a lease attached to a dynamic interface.

**SYNTAX:**

```
dhcpp client ifrelease  intf = <interface name>
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>intf</td>
<td>The name of the dynamic interface.</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

```
=>dhcpp client iflist
NewMer : [SELECTING]
  flags = uc
  IP address : 10.0.0.10
  HW address : 0:90:d0:01:47:de
  DHCP server : 255.255.255.255
  hostname : NewLease
  req.leasetime : 10800 s
  trying to get a lease for 8 min, 32 sec
  transmission of DISCOVER in 57 sec
  retransmission timeout: 64
  nbr of retransmissions: 14

Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %

=>dhcpp client ifattach intf=NewMer

=>dhcpp client iflist
NewMer : [INIT]
  flags = uc
  IP address : 0.0.0.0
  HW address : 00:90:d0:01:47:de
  DHCP server : 255.255.255.255
  hostname : NewLease
  req.leasetime : 10800 s

Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %
=>
```
EXAMPLE INPUT/OUTPUT IN A NETWORKED ENVIRONMENT:
The SpeedTouch™ Pro with Firewall is configured as DHCP client on its Ethernet interface eth0.

```
=> dhcp client iflist
eth0: [BOUND]
   flags = uc
   IP address: 10.0.0.3
   HW address: 00:90:d0:01:47:f1
   DHCP server: 10.10.1.1
   lease renewal in: 5 days, 58 min, 48 sec
   lease rebinding in: 8 days, 20 h, 6 min, 18 sec
   lease expires in: 10 days, 2 h, 28 min, 48 sec

Number of leases: 1
Total size of table: 18, in use: 1, free: 94%
```

```
=> dhcp client stats
DHCP client statistics:
   Corrupted packet recv: 0
   DECLINEs sent: 0
   RELEASEs sent: 0
   INFORMs sent: 0
Number of dynamic interfaces: 1
Memory usage:
   Table size of dyn leases: 19, in use: 1, free: 94%
```

```
=> dhcp client ifrelease intf=eth0
=> (CTRL + Q)
=> STATE ACTIVATE!
STATE IDLE!
STATE ACTIVATE!
dhcc: intf 1 releases 10.0.0.3 to server 10.10.1.1.
dhcc: 10.0.0.3 deleted: ok.
STATE IDLE!
STATE ACTIVATE!
.............
dhcc: intf 1 in init state.
n_send() broadcast triggered; To be verified
dhcc: broadcast discover on intf 1.
=> (CTRL + S)
=> dhcp client stats
DHCP client statistics:
   Corrupted packet recv: 0
   DECLINEs sent: 0
   RELEASEs sent: 1
   INFORMs sent: 0
Number of dynamic interfaces: 1
Memory usage:
   Table size of dyn leases: 18, in use: 1, free: 94%
```

RELATED COMMANDS:
```
dhcp client ifattach
   Attach a DHCP lease to an interface.
dhcp client ifconfig
   Configure a DHCP lease created for a specific interface.
dhcp client ifdelete
   Delete a dynamic interface.
```
**dhcp client ifrenew**
Renew the lease of a dynamic interface.

**SYNTAX:**

```
dhcp client ifrenew  intf = <interface name>
```

- **intf**: The name of the dynamic interface.
  - **REQUIRED**

**EXAMPLE:**

```
=> dhcp client iflist
NewMer  : [BOUND]
  flags = uc
  IP address  : 10.0.0.10
  HW address  : 00:90:d0:01:47:de
  DHCP server : 255.255.255.255
  hostname    : NewLease
  req.leasetime= 10800 s
  lease renewal in  5 days, 58 min, 48 sec
  lease rebinding in 8 days, 20 h, 6 min, 18 sec
  lease expires in 10 days, 2 h, 28 min, 48 sec
Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %
=> dhcp client ifrenew intf=NewMer
=> dhcp client iflist
NewMer  : [RENEWING]
  flags = uc
  IP address  : 10.0.0.10
  HW address  : 00:90:d0:01:47:de
  DHCP server : 255.255.255.255
  hostname    : NewLease
  req.leasetime= 10800 s
  trying to get a lease for 12 sec
  transmission of DISCOVER in 24 sec
  retransmission timeout: 64
  nbr of retransmissions: 11
Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %
=>
```
EXAMPLE INPUT/OUTPUT IN A NETWORKED ENVIRONMENT:
The SpeedTouch™ Pro with Firewall is configured as DHCP client on its Ethernet interface eth0.

```plaintext
=> dhcp client stats
DHCP client statistics:
Corrupted packet recv : 0
OFFERS recv : 0
ACKs recv : 0
NAKs recv : 0
Pure BOOTP REPLIES : 0
Other message types : 0
DISCOVERs sent : 0
REQUESTs sent : 0
DECLINEs sent : 0
RELEASEs sent : 1
INFORMs sent : 0
Number of dynamic interfaces: 1
Memory usage:
Table size of dyn leases: 18, in use: 1, free: 94 %
=> dhcp client ifrenew intf=eth0
=> dhcp client stats
DHCP client statistics:
Corrupted packet recv : 0
OFFERS recv : 1
ACKs recv : 1
NAKs recv : 0
Pure BOOTP REPLIES : 0
Other message types : 0
DISCOVERs sent : 1
REQUESTs sent : 1
DECLINEs sent : 0
RELEASEs sent : 1
INFORMs sent : 0
Number of dynamic interfaces: 1
Memory usage:
Table size of dyn leases: 19, in use: 1, free: 94 %
=> (CTRL + Q)
 ................
STATE IDLE !
STATE ACTIVATE !
dhcc: intf 1 renews lease 10.0.0.3.
dhcc: intf 1 requests 10.0.0.3 from 10.10.1.1
dhcc: 10.10.1.1 acks 10.0.0.3 to intf 1.
dhcc: lease 10.0.0.3 bound to intf 1.
STATE IDLE !
STATE ACTIVATE !
...........
=> (CTRL + S)
```

RELATED COMMANDS:

- `dhcp client ifadd` Create a DHCP lease for a specific interface.
- `dhcp client ifattach` Attach a DHCP lease to an interface.
### dhcp client load

Load saved DHCP client configuration and dynamic interfaces.

Execute **dhcp client flush** prior to **dhcp client load**.

**SYNTAX:**

```plaintext
dhcp client load
```

**EXAMPLE:**

```plaintext
=>dhcp client iflist
NewMer : [SELECTING]
  flags = uc
  IP address : 10.0.0.10
  HW address : 00:90:d0:01:47:de
  DHCP server : 255.255.255.255
  hostname : NewLease
  req.leasetime= 10800s
  trying to get a lease for 8 min, 32 sec
  transmission of DISCOVER in 57 sec
  retransmission timeout: 64
  nbr of retransmissions: 14

Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %

=>dhcp client save
=>dhcp client flush
=>dhcp client iflist
No dynamic interfaces defined.

=>dhcp client load
=>dhcp client iflist
NewMer : [REBOOTING]
  flags = uc
  IP address : 10.0.0.10
  HW address : 00:90:d0:01:47:de
  DHCP server : 255.255.255.255
  hostname : NewLease
  req.leasetime= 10800s
  trying to get a lease for 2 sec
  transmission of REQUEST in 2 sec
  retransmission timeout: 4
  retransmissions left before reinitializing : 2

Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %

=>
```

**RELATED COMMANDS:**

- **dhcp client flush**  
  Delete all dynamic interfaces.

- **dhcp client save**  
  Save current DHCP client configuration and dynamic interfaces.
**dhcp client save**

Save current DHCP client configuration and dynamic interfaces.

**SYNTAX:**

```
dhcp client save
```

**EXAMPLE:**

```
=>dhcp client iflist
NewMer : [SELECTING]
      flags = uc
      IP address : 10.0.0.10
      HW address : 00:90:d0:01:47:de
      DHCP server : 255.255.255.255
      hostname : NewLease
      req.leasetime= 10800s
      trying to get a lease for 8 min, 32 sec
      transmission of DISCOVER in 57 sec
      retransmission timeout: 64
      nbr of retransmissions: 14
Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %

=>dhcp client save
=>dhcp client flush
=>dhcp client iflist
No dynamic interfaces defined.

=>dhcp client load
=>dhcp client iflist
NewMer : [REBOOTING]
      flags = uc
      IP address : 10.0.0.10
      HW address : 00:90:d0:01:47:de
      DHCP server : 255.255.255.255
      hostname : NewLease
      req.leasetime= 10800s
      trying to get a lease for 2 sec
      transmission of REQUEST in 2 sec
      retransmission timeout: 4
      retransmissions left before reinitializing : 2
Number of leases: 1
Total size of table: 19, in use: 1, free: 94 %

=>
```

**RELATED COMMANDS:**

- **dhcp client flush**
  Flush complete DHCP client configuration and dynamic interfaces.

- **dhcp client load**
  Load saved or default DHCP client configuration and dynamic interfaces.
**dhcp client stats**

Show DHCP client statistics.

**SYNTAX:**

```
dhcp client stats
```

**EXAMPLE:**

```
=> dhcp client stats
DHCP client statistics:
  Corrupted packet recv : 0
  OFFERs recv : 1
  ACKs recv : 1
  NAKs recv : 0
  Pure BOOTP REPLIES : 0
  Other message types : 0
  DISCOVERs sent : 244
  REQUESTs sent : 9
  DECLINEs sent : 0
  RELEASEs sent : 0
  INFORMs sent : 0
  Number of dynamic interfaces: 1
  Memory usage:
  Table size of dyn leases: 19, in use: 1, free: 94 %
=>
```

**RELATED COMMANDS:**

- **dhcp client clear** Clear DHCP client statistics.
**dhcp server add**

Assign a static IP address to a host in the local network. This address is allocated on a permanent basis, and is excluded from the pool of addresses used by the **SpeedTouch™ Pro with Firewall** DHCP server.

**SYNTAX:**

```
dhcp server add  
  clientid = <client-id>  
  addr = <ip-address>  
  [leasetime = <number>]  
  [hostname = <hostname>]  
```

- **clientid**  The DHCP client’s MAC address.  **REQUIRED**
- **addr**  The IP address for this DHCP host.  **REQUIRED**
- **[leasetime]**  A number between 0 and 1814400 (seconds).  **OPTIONAL**
  Represents the preferred time the client wants to use an address.  
  By default the leasetime is 7200 seconds (2 hours).  
  Specifying -1 makes the lease permanent.
- **[hostname]**  The hostname to add to the local DNS table for this host.  **OPTIONAL**

**EXAMPLE:**

```
=> dhcp server list
Leases:  
Lease 0: 01:00:A0:24:AE:66:E1  
  Hostname = Default  
  ip address : 10.0.0.8  
  expires in : 1 h, 16 min, 20 sec  
  lease is being used.  
Total size of table: 36, in use: 1 free: 97 %

=> dhcp server add clientid=01:23:45:67:89:ab addr=10.0.0.1 leasetime=60 hostname=NewLease
=> dhcp server list
Leases:  
Lease 0: 01:00:A0:24:AE:66:E1  
  Hostname = Default  
  ip address : 10.0.0.8  
  expires in : 1 h, 16 min, 20 sec  
  lease is being used.  
Lease 1: 01:23:45:67:89:AB  
  Hostname = NewLease  
  ip address : 10.0.0.1  
  expires in : 23 sec  
  lease is being used.  
Total size of table: 36, in use: 2 free: 94 %
=>
```

**RELATED COMMANDS:**

- **dhcp server delete**  Delete a DHCP lease.
- **dhcp server list**  Show current DHCP leases.
**dhcp server client**

Set the AutoDHCP client time-out in startup phase. Only applicable in AutoDHCP mode (See **dhcp server policy** command).

**SYNTAX:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dhcp server client</td>
<td>timeout = &lt;number&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>A number between 0 and 1814400 (seconds). REQUIRED</td>
</tr>
</tbody>
</table>

Represents the time to look for another DHCP server. Specifying `-1` will make the timeout infinite: the **SpeedTouch™ Pro with Firewall** will remain client. By default the timeout is 20 seconds.

**EXAMPLE:**

```shell
=> dhcp server status
DHCP Server Status: Running
Current configuration:
  Address Range: 10.0.0.1 ... 10.255.255.254

........
Start-up client parameters:
  Timeout: 20 sec
Tracing: off
Memory usage:
  Leases: total: 36, in use: 7 free: 80 %

=> dhcp server client timeout=15
=> dhcp server status
DHCP Server Status: Running
Current configuration:
  Address Range: 10.0.0.1 ... 10.255.255.254

........
Start-up client parameters:
  Timeout: 15 sec
Tracing: off
Memory usage:
  Leases: total: 36, in use: 7 free: 80 %
```

**RELATED COMMANDS:**

- **dhcp server policy**
  Set DHCP server policy.
- **dhcp server start**
  Start DHCP server.
- **dhcp server status**
  Show current DHCP server configuration.
- **dhcp server stop**
  Stop DHCP server.
**dhcp server clrstats**

Clear SpeedTouch™ Pro with Firewall DHCP server statistics.

**SYNTAX:**

```
dhcp server clrstats
```

**EXAMPLE:**

```
=>dhcp server stats
DHCP server statistics:
Corrupted packet recv : 0
DISCOVER : 9575
REQUEST : 121
DECLINE : 0
RELEASE : 0
INFORM : 13
Pure BOOTP REQUESTS : 2
Other message types : 0
OFFERS sent : 9552
ACKs sent : 121
NAKs sent : 0
Lease table got full : no
Ping table got full : no
Second DHCP server seen : no

=>dhcp server clrstats
=>dhcp server stats
```
**dhcp server config**

Set **SpeedTouch™ Pro with Firewall** DHCP server configuration. Execute **dhcp server status** to see the actual status and configuration.

**SYNTAX:**

```
<table>
<thead>
<tr>
<th>dhcp server config</th>
<th>[beginrange = &lt;ip-address&gt;]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[endrange = &lt;ip-address&gt;]</td>
</tr>
<tr>
<td></td>
<td>[netmask = &lt;ip-address&gt;]</td>
</tr>
<tr>
<td></td>
<td>[leasetime = &lt;number&gt;]</td>
</tr>
<tr>
<td></td>
<td>[gateway = &lt;ip-address</td>
</tr>
<tr>
<td></td>
<td>[dnsaddr = &lt;ip-address&gt;]</td>
</tr>
</tbody>
</table>
```

- **beginrange**
  - The lowest IP address in the DHCP address range to use for leasing.
  - Default value of this parameter is 10.0.0.1.
  - **OPTIONAL**

- **endrange**
  - The highest IP address in the DHCP address range to use for leasing.
  - Default value of this parameter is 10.255.255.254.
  - **OPTIONAL**

- **netmask**
  - The applicable netmask for the DHCP leases.
  - **OPTIONAL**

- **leasetime**
  - A number between 0 and 1814400 (seconds).
  - Represents the time for which a client can use its dynamically allocated IP address.
  - By default the leasetime is 2 hours (7200 seconds).
  - Specifying −1 makes the lease permanent.
  - **OPTIONAL**

- **gateway**
  - The IP address of the gateway for DHCP clients.
  - **OPTIONAL**

- **dnsaddr**
  - The IP address of the DNS server for DHCP clients.
  - Entering ‘0’ sets the **SpeedTouch™ Pro with Firewall** as DNS server.
  - **OPTIONAL**
EXAMPLE:

=> dhcp server status
DHCP Server Status: Running
Current configuration:
  Address Range: 10.0.0.1 ... 10.255.255.254
  Netmask: 255.0.0.0
  Lease time: 10800 seconds
  Gateway (default router): 10.0.0.138
  DNS server: 10.0.0.1
  Domain name: lan
Policies:
  Verify first: no
  Trust client: yes
  Spoofing: no
  Start as client: yes
Spoofing parameters:
  Failure timeout (!DoD): 4 sec
  Failure lease time (!DoD): 60 sec
  Temp. lease time (DoD): 10 sec
Start-up client parameters:
  Timeout: 15 sec
Tracing: off
Memory usage:
  Leases: total: 36, in use: 7 free: 80 %

=> dhcp server config beginrange=172.16.0.2 endrange=172.16.0.122 netmask=255.0.0.0 leasetime=21600 gateway=172.16.0.1 dnsaddr=172.16.0.254

=> dhcp server status
DHCP Server Status: Running
Current configuration:
  Address Range: 172.16.0.2 ... 172.16.0.122
  Netmask: 255.0.0.0
  Lease time: 21600 seconds
  Gateway (default router): 172.16.0.1
  DNS server: 172.16.0.254
  Domain name: lan
Policies:
  Verify first: no
  Trust client: yes
  Spoofing: no
  Start as client: yes
Spoofing parameters:
  Failure timeout (!DoD): 4 sec
  Failure lease time (!DoD): 60 sec
  Temp. lease time (DoD): 10 sec
Start-up client parameters:
  Timeout: 15 sec
Tracing: off
Memory usage:
  Leases: total: 36, in use: 7 free: 80 %

RELATED COMMANDS:

dhcp server status Show current DHCP server configuration.
**dhcp server delete**
Delete a DHCP lease.

**SYNTAX:**

```
dhcp server delete  index = <number>
```

<table>
<thead>
<tr>
<th>index</th>
<th>The index number of the entry to be deleted. REQUIRED</th>
</tr>
</thead>
</table>

**EXAMPLE:**

```
=> dhcp server list
Leases:
Lease 0: 01:00:A0:24:AE:66:E1
   Hostname = Default
   ip address : 10.0.0.8
   expires in : 1 h, 16 min, 20 sec
   lease is being used.
Lease 1: 01:23:45:67:89:AB
   Hostname = NewLease
   ip address : 10.0.0.1
   expires in : 23 sec
   lease is being used.
Total size of table: 36, in use: 2 free: 94 %
```

```
=> dhcp server delete index=1
```

```
=> dhcp server list
Leases:
Lease 0: 01:00:A0:24:AE:66:E1
   Hostname = Default
   ip address : 10.0.0.8
   expires in : 1 h, 15 min, 32 sec
   lease is being used.
Total size of table: 36, in use: 1 free: 97 %
```

**RELATED COMMANDS:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dhcp server add</td>
<td>Add a DHCP lease manually.</td>
</tr>
<tr>
<td>dhcp server list</td>
<td>Show current DHCP leases.</td>
</tr>
</tbody>
</table>
dhcp server flush
Flush complete DHCP server configuration and dynamic leases.
The flush command does not impact previously saved configurations.

SYNTAX:
```
dhcp server flush
```

EXAMPLE:
```
=> dhcp server list
Leases:
Lease  2: 01:52:41:53:20:A0:1B:AE:AD:3C:01:01:00:00:00
  ip address: 10.0.7.79
  expires in: 11 sec
  lease is not being used.
Lease  1: 01:52:41:53:20:20:4D:0D:CB:03:40:C0:01:01:00:00:00
  ip address: 10.0.7.62
  Spoofed lease from 2: DHCP_SPOOF
  Assigned (temporary) private ip address.
  expires in: 1 min, 39 sec
  lease is not being used.
Lease   0: 01:00:A0:24:AE:66:E1
  Hostname = Default
  ip address: 10.0.0.8
  expires in: 1 h, 16 min, 20 sec
  lease is being used.
  Hostname = Tempo
  ip address: 10.0.0.1
  never expires!
  lease is not being used.
Total size of table: 36, in use: 4 free: 89 %
```
```
=> dhcp server flush
```
```
=> dhcp server list
No active leases
Total size of table: 36, in use: 0 free: 100 %
=>
```

RELATED COMMANDS:
```
dhcp server load
  Load saved or default DHCP server configuration and permanent leases.
dhcp server save
  Save current DHCP server configuration and permanent leases.
```
**dhcp server list**

List current DHCP leases, indicated by their index number.

**SYNTAX:**

```
dhcp server list
```

**EXAMPLE OUTPUT:**

```
=>dhcp server list
Leases:
Lease 2: 01:52:41:53:20:50:6D:C0:40:02:32:C0:01:01:00:00:00
    ip address: 10.0.7.142
    expires in: 3 sec
    lease is not being used.
Lease 3: 01:52:41:53:20:A0:1B:A7:EB:AD:3C:C0:01:01:00:00:00
    ip address: 10.0.7.143
    expires in: 17 sec
    lease is not being used.
Lease 5: 01:52:41:53:20:FO:90:8F:09:E1:35:BE:01:01:00:00:00
    ip address: 10.0.7.144
    expires in: 55 sec
    lease is not being used.
Lease 6: 01:52:41:53:20:30:F4:89:5F:9B:44:C0:01:01:00:00:00
    ip address: 10.0.7.145
    expires in: 1 min, 6 sec
    lease is not being used.
Lease 1: 01:52:41:53:20:20:4D:0D:CB:03:40:C0:01:01:00:00:00
    ip address: 10.0.7.62
    Spoofed lease from 2: DHCP_SPOOF
    Assigned (temporary) private ip address.
    expires in: 1 min, 57 sec
    lease is not being used.
Lease 0: 01:00:A0:24:AE:66:E1
    Hostname= Default
    ip address: 10.0.0.8
    expires in: 1 h, 17 min, 21 sec
    lease is being used.
    Hostname = Tempo
    ip address: 10.0.0.1
    never expires!
    lease is not being used.
Total size of table: 36, in use: 7 free: 80 %
=>
```

**RELATED COMMANDS:**

- `dhcp server add` Add a DHCP lease manually.
- `dhcp server delete` Delete a DHCP lease.
- `dhcp server flush` Delete complete DHCP server configuration and dynamic leases.
**dhcp server load**

Load saved (or default) DHCP server configuration and permanent leases.

**SYNTAX:**

```
dhcp server load [defaults = <yes|no>]
```

**[defaults]**

Load factory defaults (yes) or saved configuration (no).

Not specifying this parameter loads the saved configuration.

**EXAMPLE:**

```
=>dhcp server list
Leases:
Lease 1: 01:52:41:53:20:20:4D:0D:CB:03:40:C0:01:01:00:00
  ip address: 10.0.7.62
  Spoofed lease from 2: DHCP_SPOOF
  Assigned (temporary) private ip address.
  expires in: 1 min, 57 sec
  lease is being used.
Lease 0: 01:00:A0:24:AE:66:E1
  Hostname= Default
  ip address: 10.0.0.8
  expires in: 1 h, 17 min, 21 sec
  lease is being used.
  Hostname = Tempo
  ip address : 10.0.0.1
  never expires!
  lease is not being used.
Total size of table: 36, in use: 3 free: 92 %
=>dhcp server save
=>dhcp server flush
=>dhcp server load
=>dhcp server list
Leases:
  Hostname = Tempo
  ip address : 10.0.0.1
  never expires!
  lease is not being used.
Total size of table: 36, in use: 1 free: 97 %
=>
```

**RELATED COMMANDS:**

- **dhcp server flush**
  - Flush current DHCP server configuration and dynamic leases.

- **dhcp server load**
  - Load saved or default DHCP server configuration and permanent leases.
**dhcp server policy**

Set SpeedTouch™ Pro with Firewall DHCP server policy.

**SYNTAX:**

```
dhcp server policy [verifyfirst = <yes|no>] [trustclient = <yes|no>] [spoofing = <yes|no>] [client = <yes|no>]
```

- **[verifyfirst]** Probe the network for conflicting IP addresses before giving a suggested IP address to the requesting DHCP client (yes) or not (no).  
  OPTIONAL

- **[trustclient]** Take the IP address suggested by a DHCP client into account (yes) or not (no).  
  OPTIONAL

- **[spoofing]** Allow a remote DHCP server to hand out IP addresses negotiated by PPP on WAN side (yes) or not (no).  
  DHCP spoofing is used to relay local DHCP requests to an external PPP connection having a specific IP address negotiation mechanism.  
  DHCP replies are in turn generated by the DHCP server based on the IP address information received by the PPP link.  
  OPTIONAL

- **[client]** Allow the SpeedTouch™ Pro with Firewall DHCP server to present itself as DHCP client (AutoDHCP mode) at boot time and probe for another DHCP server on the network for some time before starting the DHCP server (yes) or immediately start the DHCP server (no).  
  OPTIONAL

**EXAMPLE:**

```
=> dhcp server status
DHCP Server Status: Running
Current configuration:  
......
Policies:
  Verify first: no
  Trust client: yes
  Spoofing: no
  Start as client: yes
......
=> dhcp server policy verifyfirst=yes trustclient=no spoofing=yes client=no
=> dhcp server status
DHCP Server Status: Running
Current configuration:  
......
Policies:
  Verify first: yes
  Trust client: no
  Spoofing: yes
  Start as client: no
......
=>
```

**RELATED COMMANDS:**

- **dhcp server status** Show current DHCP server configuration.
**dhcp server save**

Save complete SpeedTouch™ Pro with Firewall DHCP server configuration and permanent DHCP leases.

**SYNTAX:**

```
dhcp server save
```

**EXAMPLE:**

```
=>dhcp server list
Leases:
Lease 0: 01:00:A0:24:AE:66:E1
  Hostname= Default
  ip address: 10.0.0.8
  expires in: 1 h, 57 min, 9 sec
  lease is being used.
  Hostname= Tempo
  ip address: 10.0.0.1
  never expires!
  lease is not being used.
Total size of table: 36, in use: 2 free: 94 %

=>dhcp server save
=>dhcp server flush
=>dhcp server list
No active leases
Total size of table: 36, in use: 0 free: 100 %

=>dhcp server load defaults=no
=>dhcp server list
Leases:
Lease 0: 01:00:A0:24:AE:66:E1
  Hostname= Default
  ip address: 10.0.0.8
  expires in: 1 h, 58 min, 55 sec
  lease is being used.
  Hostname= Tempo
  ip address: 10.0.0.1
  never expires!
  lease is not being used.
Total size of table: 36, in use: 2 free: 94 %
=>
```

**RELATED COMMANDS:**

- **dhcp server flush**
  Flush complete DHCP server configuration and dynamic leases

- **dhcp server load**
  Load saved or default DHCP server configuration and permanent leases.
**dhcp server spoof**

Set DHCP spoofing parameters. Only applicable in case of a PPP-to-DHCP Spoofing connection. (See `dhcp server policy` command).

**SYNTAX:**

<table>
<thead>
<tr>
<th>dhcp server spoof</th>
<th>[failtime = &lt;number&gt;]</th>
<th>[errorlt = &lt;number&gt;]</th>
<th>[dodlt = &lt;number&gt;]</th>
</tr>
</thead>
</table>

- **[failtime]**
  - A number between 0 and 1814400 (seconds).
  - Represents the time to wait for a PPP link to successfully negotiate an IP address.
  - This parameter determines how long the *SpeedTouch™ Pro with Firewall* should try to set up a PPP connection before returning to normal DHCP mode, i.e. in case the PPP connection cannot be established within the time lapse determined by failtime, the *SpeedTouch™ Pro with Firewall* DHCP server will allocate an local private IP address to the DHCP client.
  - By default the failtime is 4 seconds.

- **[errorlt]**
  - A number between 0 and 1814400 (seconds).
  - Represents the leasetime of the private address issued when a PPP link fails.
  - In case the PPP link fails after failtime has elapsed, this parameter determines how long the private DHCP lease must be maintained before retrying to set up the PPP link again.
  - By default the error lease time is 60 seconds.

- **[dodlt]**
  - A number between 0 and 1814400 (seconds).
  - Represents the leasetime of the temporary private IP address in case of a dial-on-demand PPP link.
  - In case of a dial-on-demand PPP link, this parameter determines the interval at which the the temporary DHCP lease must be maintained before checking whether a public IP address negotiated by a triggered PPP link is available.
  - By default the dial-on-demand lease time is 10 seconds.
### RELATED COMMANDS:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>dhcp server policy</code></td>
<td>Set DHCP server policy.</td>
</tr>
<tr>
<td><code>dhcp server status</code></td>
<td>Show current DHCP server configuration.</td>
</tr>
</tbody>
</table>
**dhcp server start**
Start SpeedTouch™ Pro with Firewall DHCP server.

**SYNTAX:**

```
dhcp server start
```

**EXAMPLE:**

```
=>dhcp server status
DHCP Server Status: Stopped
Current configuration:
.....
=>dhcp server start
=>dhcp server status
DHCP Server Status: Searching for server...
Current configuration:
.....
=>
=>dhcp server status
DHCP Server Status: Running
Current configuration:
.....
=>
```

**RELATED COMMANDS:**

- `dhcp server status`  
  Show current DHCP server configuration.

- `dhcp server stop`  
  Stop DHCP server.
**dhcp server stats**
Show *SpeedTouch™ Pro with Firewall* DHCP server statistics.

**SYNTAX:**

```
dhcp server stats
```

**EXAMPLE OUTPUT:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corrupted packet recvd</td>
<td>0</td>
</tr>
<tr>
<td>DISCOVER</td>
<td>9575</td>
</tr>
<tr>
<td>REQUEST</td>
<td>121</td>
</tr>
<tr>
<td>DECLINE</td>
<td>0</td>
</tr>
<tr>
<td>RELEASE</td>
<td>0</td>
</tr>
<tr>
<td>INFORM</td>
<td>13</td>
</tr>
<tr>
<td>Pure BOOTP REQUESTS</td>
<td>2</td>
</tr>
<tr>
<td>Other message types</td>
<td>0</td>
</tr>
<tr>
<td>OFFERS sent</td>
<td>9552</td>
</tr>
<tr>
<td>ACKs sent</td>
<td>121</td>
</tr>
<tr>
<td>NAKs sent</td>
<td>0</td>
</tr>
<tr>
<td>Lease table got full</td>
<td>no</td>
</tr>
<tr>
<td>Ping table got full</td>
<td>no</td>
</tr>
<tr>
<td>Second DHCP server seen</td>
<td>no</td>
</tr>
</tbody>
</table>

**DESCRIPTION:**

- **Corrupted packet recvd** Indicates the number of corrupted packets (not complaint to RFC2131) were received from the LAN.
- **DISCOVER** Indicates the number of DHCP server discovery packets were received from the LAN. These broadcasts are sent by potential DHCP clients to locate available DHCP servers.
- **REQUEST** Indicates the number of DHCP address lease requests were received from the LAN.
- **DECLINE** Indicates the number of DHCP address lease requests that were declined.
- **RELEASE** Indicates the number of DHCP address release requests that were received from DHCP clients.
- **INFORM** Indicates the number of information requests that were received from DHCP clients.
- **Pure BOOTP requests** Indicates the number of BOOTP requests that were received from the LAN.
- **OFFERS sent** Indicates the number of IP address offers were sent in reply to DHCP requests.
ACKs sent Indicates the number of ACKnowledgement replies were sent to successfully configured DHCP clients.

NAKs sent Indicates the number of Not-AcKnowledgement replies were sent to wrongly configured DHCP clients.

Lease table got full Indicates whether the maximum number of DHCP leases is reached or not.

Ping table got full Indicates whether the history list of IP address pings got full or not. These pings are sent by the SpeedTouch™ Pro with Firewall DHCP server to verify whether the IP address is already in use on the LAN or not. (dhcp server policy verifyfirst=yes)

Second DHCP server seen Indicates whether a concurrent DHCP server was found on the LAN or not.

RELATED COMMANDS:

\texttt{dhcp server clrstats} Clear DHCP server statistics.
**dhcp server status**
Show current DHCP server configuration.

**SYNTAX:**
```
dhcp server status
```

**EXAMPLE:**
```
=> dhcp server status
DHCP Server Status: Client
Current configuration:
  Address Range: 10.0.0.1 ... 10.255.255.254
  Netmask: 255.0.0.0
  Lease time: 7200 seconds
  Gateway (default router): 10.0.0.1 (auto)
  DNS server: 10.0.0.1 (auto)
  Domain name: office.lan
Policies:
  Verify first: no
  Trust client: yes
  Spoofing: no
  Start as client: yes
Spoofing parameters:
  Failure timeout (!DoD): 4 sec
  Failure lease time (!DoD): 60 sec
  Temp. lease time (DoD): 10 sec
Start-up client parameters:
  Timeout: 20 sec
Tracing: off
Memory usage:
  Leases: total: 36, in use: 7 free: 80 %
=>
```

**RELATED COMMANDS:**
- **dhcp server stop**
  Stop DHCP server.
- **dhcp server start**
  Start DHCP server.
- **dhcp server policy**
  Set DHCP server policy.
- **dhcp server spoofing**
  Set spoofing parameters.
**dhcp server stop**

Stop *SpeedTouch™ Pro with Firewall* DHCP server.

**SYNTAX:**

```
dhcp server stop
```

**EXAMPLE:**

```
=>dhcp server status
DHCP Server Status: Running
Current configuration:
.....
=>dhcp server stop
=>dhcp server status
DHCP Server Status: Stopped
Current configuration:
.....
=>
```

**RELATED COMMANDS:**

- `dhcp server start` Start DHCP server.
- `dhcp server status` Show current DHCP server configuration.
**dhcp server troff**

Disable verbose console logging. No debug traces are generated anymore.

**SYNTAX:**

```
dhcp server troff
```

**EXAMPLE:**

```
=> dhcp server status
DHCP Server Status: Running
Current configuration:
.....
Tracing:on
.....
=> dhcp server troff
=> dhcp server status
DHCP Server Status: Running
Current configuration:
.....
Tracing:off
.....
=>
```

**RELATED COMMANDS:**

- **dhcp server status**  Show current DHCP server configuration.
- **dhcp server tron**    Enable verbose console logging.
**dhcp server tron**
Enable verbose console logging. Debug traces are generated.

**SYNTAX:**
```
dhcp server tron
```

**EXAMPLE:**
```
=> dhcp server status
DHCP Server Status: Running
Current configuration:
.....
Tracing:off
.....
=> dhcp server tron
=> dhcp server status
DHCP Server Status: Running
Current configuration:
.....
Tracing:on
.....
=>
```

**RELATED COMMANDS:**
- **dhcp server status**  
  Show current DHCP server configuration
- **dhcp server troff**  
  Disable verbose console logging.
7 DNS Commands

dns (to access the DNS level)
dns add
dns clear
dns clrstats
dns delete
dns domain
dns flush
dns fwdadd
dns fwddelete
dns fwdlist
dns fwdtable
dns list
dns load
dns nslookup
dns save
dns start
dns stats
dns status
dns stop
dns toutfwd
dns troff
dns tron
**dns add**
Add a static DNS entry for IP hosts who do not reveal their hostname in the DHCP request, or even worse, not support DHCP.

**SYNTAX:**

```
dns add  hostname = <string>  
        [addr = <ip-address>]
```

<table>
<thead>
<tr>
<th>hostname</th>
<th>The name of the IP host (without the (sub)domain name).</th>
<th>REQUIRED</th>
</tr>
</thead>
<tbody>
<tr>
<td>[addr]</td>
<td>The IP address of the host (without mask).</td>
<td>OPTIONAL</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

```
=>dns list
Domain: business.lan
Nr.  Hostname      IP Address
0    SpeedTouch    *.*.*.*
1    TestHost      10.0.0.140
2    HTTP_Server   10.0.0.8
Total Table Size: 73 entries
Amount used: 3 (4%)
=>dns add hostname=FTP_Server addr=10.0.0.7
=>dns list
Domain: business.lan
Nr.  Hostname      IP Address
0    SpeedTouch    *.*.*.*
1    TestHost      10.0.0.140
2    HTTP_Server   10.0.0.8
3    FTP_Server    10.0.0.7
Total Table Size: 73 entries
Amount used: 4 (5%)
=>
```

**RELATED COMMANDS:**

- **dns list**  List current DNS entries.
- **dns delete**  Delete a DNS entry.
**dns clear**
Delete current DNS entries.

**SYNTAX:**
```
dns clear
```

**EXAMPLE:**
```
=>dns list
Domain: business.lan
Nr.  Hostname    IP Address
  0  SpeedTouch    *.*.*.*
  1  TestHost      10.0.0.140
  2  HTTP_Server   10.0.0.8
  3  FTP_Server    10.0.0.7
Total Table Size: 73 entries
Amount used: 4 (5%)
=>dns clear
=>dns list
Domain: business.lan
Nr.  Hostname    IP Address
Total Table Size: 73 entries
Amount used: 0 (0%)
=>
```

**RELATED COMMANDS:**
- **dns list** List current DNS entries.
**dns clrstats**

Clear DNS statistics.

**SYNTAX:**

```
dns clrstats
```

**EXAMPLE:**

```
=>dns stats
DNS Statistics:
Corrupted packets recv : 0
Local questions resolved : 0
Local neg answers sent : 4
Total DNS packets fwd : 0
External answers recv : 0
Fwd table full, discard : 0
Spurious answers : 0
Unknown query types : 0
Total number of packets received : 4

=>dns clrstats
DNS statistics cleared.
=>dns stats
DNS Statistics:
Corrupted packets recv : 0
Local questions resolved : 0
Local neg answers sent : 0
Total DNS packets fwd : 0
External answers recv : 0
Fwd table full, discard : 0
Spurious answers : 0
Unknown query types : 0
Total number of packets received : 0
=>
```

**RELATED COMMANDS:**

- **dns stats**
  
  Show DNS server/forwarder statistics.
**dns delete**

Delete a DNS entry.

**SYNTAX:**

```

dns delete       index = <number>
```

<table>
<thead>
<tr>
<th>index</th>
<th>Description</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>The index number of the entry to be deleted.</td>
<td>REQUIRED</td>
<td></td>
</tr>
</tbody>
</table>
**dns domain**
Set local DNS (sub)domain name.

**SYNTAX:**
```
dns domain  domain = <string>
```

- `domain`: The local DNS (sub)domain name. **REQUIRED**

**EXAMPLE:**
```
=> dns list
Domain: business.lan
Nr.  Hostname          IP Address
0    SpeedTouch        *.*.*.*
1    TestHost          10.0.0.140
2    HTTP_Server       10.0.0.8
3    FTP_Server        10.0.0.7
Total Table Size: 73 entries
Amount used: 4 (5%)

=> dns domain domain=office.home.lan
=> dns list
Domain: office.home.lan
Nr.  Hostname          IP Address
0    SpeedTouch        *.*.*.*
1    TestHost          10.0.0.140
2    HTTP_Server       10.0.0.8
3    FTP_Server        10.0.0.7
Total Table Size: 73 entries
Amount used: 4 (5%)
```

**RELATED COMMANDS:**
- `dns list` List current DNS entries.
**dns flush**

Flush complete **SpeedTouch™ Pro with Firewall** DNS server/forwarder configuration and static entries.
The flush command does not impact previously saved configurations.

**SYNTAX:**

```
dns flush
```

**EXAMPLE:**

```
=>dns list
Domain: office.home.lan
Nr.  Hostname   IP Address
 4*  Z7V1D8     10.0.0.29
 0   SpeedTouch *.*.*.*
 1   TestHost   10.0.0.140
 2   Default    10.0.0.8
 3   ftpserver  172.16.0.1
Total Table Size: 73 entries
Amount used: 5 (6%)
=>dns flush
=>dns list
Domain: lan
Nr.  Hostname   IP Address
 3*  Z7V1D8     10.0.0.29
Total Table Size: 73 entries
Amount used: 1 (1%)
=>
```

**RELATED COMMANDS:**

- **dns save**  
  Save current DNS server/forwarder configuration and static entries.
- **dns load**  
  Load saved or default DNS server/forwarder configuration and static entries.
**dns fwdadd**

Add a DNS forwarding entry. The entries in the forwarding list determine which DNS server should be used for which PC. If an identification cannot be established within the local LAN, the request is forwarded to another DNS server, on another network (Internet/LAN to LAN connection). The connection is negotiated within a PPP link.

**SYNTAX:**

```
dns fwdadd  
dns = <ip-address>  
src = <ip-address>  
mask = <ip-mask (dotted or cidr)>  
[direct = <number>]
```

- `dns`: The IP address of the (remote) DNS server. **REQUIRED**
- `src`: The source IP address (pool) of the host(s) using this DNS server. **REQUIRED**
- `mask`: The appropriate source IP (sub)netmask. **REQUIRED**
- `[direct]`: Determines whether DNS replies are sent directly back to the client (1) or relayed by the SpeedTouch **Pro with Firewall** DHCP server’s DNS forwarder (0) in case of PPP-to-DHCP spoofing connections. **OPTIONAL**

**EXAMPLE:**

```
=>>dns fwdlist
DNS forwarding servers:
DNS  SRC  MASK  Direct
10.0.0.138 10.0.0.2 255.255.255.0 yes
=>>dns fwdadd dns=10.0.0.138 src=10.0.0.3 mask=24 direct=1
Dns forwarding server added.
=>>dns fwdlist
DNS forwarding servers:
DNS  SRC  MASK  Direct
10.0.0.138 10.0.0.2 255.255.255.0 yes
10.0.0.138 10.0.0.3 255.255.255.0 yes
=>>
```

**RELATED COMMANDS:**

- `dns fwddelete` **Delete a DNS forwarding entry.**
- `dns fwdlist` **Show current DNS forwarding entries.**
**dns fwdxdelete**
Delete a DNS forwarding entry.

**SYNTAX:**

```
dns fwdxdelete src = <ip-address>
  mask = <ip-mask (dotted or cidr)>
  [dns = <ip-address>]
```

- **src**: The source IP address (pool) of the hosts to remove the entry for. **REQUIRED**
- **mask**: The source IP (sub)netmask. **REQUIRED**
- **[dns]**: The IP address of the (remote) DNS server (in case of multiple DNS server entries). **OPTIONAL**

**EXAMPLE:**

```
=> dns fwdxlist
DNS forwarding servers:
DNS SRC  MASK  Direct
10.0.0.138 10.0.0.0 255.255.255.0 yes
192.6.11.150 192.6.11.0 255.255.255.0 yes

=> dns fwdxdelete src=192.6.11.0 mask=24 dns=192.6.11.150
Dns forwarding server deleted.

=> dns fwdxlist
DNS forwarding servers:
DNS SRC  MASK  Direct
10.0.0.138 10.0.0.0 255.255.255.0 yes
```

**RELATED COMMANDS:**

- **dns fwdxadd**: Add a DNS forwarding entry.
- **dns fwdxlist**: Show current DNS forwarding entries.
**dns fwdlist**
Show current DNS forwarding entries.

**SYNTAX:**
```
dns fwdlist
```

**EXAMPLE OUTPUT:**
```
=>dns fwdlist
DNS forwarding servers:
<table>
<thead>
<tr>
<th>DNS</th>
<th>SRC</th>
<th>MASK</th>
<th>Direct</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.0.0.138</td>
<td>10.0.0.0</td>
<td>255.255.255.0</td>
<td>yes</td>
</tr>
<tr>
<td>192.6.11.150</td>
<td>192.6.11.0</td>
<td>255.255.255.0</td>
<td>yes</td>
</tr>
</tbody>
</table>
=>
```

**RELATED COMMANDS:**
- **dns fwdadd**  Add a DNS forwarding entry.
- **dns fwddelete**  Delete a DNS forwarding entry.
- **dns fwdtable**  Show DNS forwarding table.
**dns fwdtable**
Show DNS forwarding table, i.e. list all currently unresolved DNS requests.

**SYNTAX:**
```
dns fwdtable
```

**EXAMPLE OUTPUT:**
```
=>dns fwdtable
Forwarding table:
Nr. Ip Address (port#):id(hex) (expiry) dns server tries
0 10.10.10.12 (54751):8331 (13 sec) 10.10.10.112 1
Timeout: 15 seconds
Table size: 10
amount of table used: 1 (10%)
=>
```

**RELATED COMMANDS:**
- **dns fwdlist**  
  Show current DNS forwarding entries.
**dns list**

Show current DNS entries.

**SYNTAX:**

```
dns list
```

**EXAMPLE OUTPUT:**

```
=>dns list
Domain: office.home.lan
Nr.  Hostname   IP Address
  4*  Z7V1D8     10.0.0.29
  0   SpeedTouch *.*.*.*
  1   TestHost   10.0.0.140
  2   Default    10.0.0.8
  3   ftpserver  172.16.0.1
Total Table Size: 73 entries
Amount used: 5 (6%)
=>
```

**EXAMPLE INPUT/OUTPUT IN A NETWORKED ENVIRONMENT:**

The SpeedTouch™ Pro with Firewall is configured as DNS server.

```
=>dns list
Domain: SpeedLAN.local
Nr.  Hostname   IP Address
  0   SpeedTouch *.*.*.*
  1   Server     10.10.1.1
  2   Client     10.0.0.3
Total Table Size: 73 entries
Amount used: 3 (4%)
=>
```

**RELATED COMMANDS:**

- `dns add` Add a static DNS entry.
- `dns delete` Delete a DNS entry (via its index number).
**dns load**

Load saved or default SpeedTouch™ Pro with Firewall DNS server/forwarder configuration and static DNS entries.

Execute **dns flush** prior to **dns load**.

**SYNTAX:**

```
| dns load | [defaults = <yes|no>] |
```

- **[defaults]**: Load factory defaults (yes) or saved configuration (no).  
  Not specifying this parameter loads the saved configuration.

**EXAMPLE:**

```bash
=> dns list
Domain: office.home.lan
Nr.  Hostname      IP Address
0   SpeedTouch     *.*.*.*
1   TestHost       10.0.0.140
2   Default        10.0.0.8
Total Table Size: 73 entries
Amount used: 4 (5%)
=> dns save
=> dns flush
=> dns list
Domain: office.home.lan
Nr.  Hostname      IP Address
0   SpeedTouch     *.*.*.*
1   TestHost       10.0.0.140
2   Default        10.0.0.8
Total Table Size: 73 entries
Amount used: 4 (5%)
=>
```

**RELATED COMMANDS:**

- **dns flush**: Flush complete DNS server/forwarder configuration and static entries.
- **dns save**: Save current DNS server/forwarder configuration and static entries.
**dns nslookup**

Search the hostname (via a known IP address) or the IP address (via a known hostname) of a DNS host.

**SYNTAX:**

```
dns nslookup lookup = <string>
```

*lookup* The DNS hostname or IP address to query.  REQUIRED

**EXAMPLE:**

```
=>dns list
Domain: office.home.lan
Nr.  Hostname       IP Address
  4*  Z7V1D8         10.0.0.29
  0   SpeedTouch     *.*.*.*
  1   TestHost       10.0.0.140
  2   Default        10.0.0.8
  3   ftpserver      172.16.0.1
Total Table Size: 73 entries
Amount used: 5 (6%)

=>dns nslookup lookup=TestHost
Name:    TestHost
Address: 10.0.0.140

=>dns nslookup lookup=10.0.0.29
Name:    Z7V1D8
Address: 10.0.0.29
```

**RELATED COMMANDS:**

- **dns list** List current DNS entries.
**dns save**

Save current *SpeedTouch™ Pro with Firewall* DNS server/forwarder configuration and static entries.

**SYNTAX:**

```
dns save
```

**EXAMPLE:**

```
=>dns fwdlist
DNS forwarding servers:
DNS    SRC       MASK                        Direct
10.0.0.138  10.0.0.2  255.255.255.0  yes
10.0.0.138  10.0.0.4  255.255.255.0  no
10.0.0.138  10.0.0.3  255.255.255.0  yes
=>dns save
=>dns flush
=>dns fwdlist
=>dns load
=>dns fwdlist
DNS forwarding servers:
DNS    SRC       MASK                        Direct
10.0.0.138  10.0.0.2  255.255.255.0  yes
10.0.0.138  10.0.0.4  255.255.255.0  no
10.0.0.138  10.0.0.3  255.255.255.0  yes
=>
```

**RELATED COMMANDS:**

- **dns flush**
  
  Flush complete DNS server/forwarder configuration and dynamic entries.

- **dns load**
  
  Load saved or default DNS server/forwarder configuration and static entries.
**dns start**

Start SpeedTouch™ Pro with Firewall DNS server/forwarder.

**SYNTAX:**

```
dns start
```

**EXAMPLE:**

```
=>dns status
DNS server status: Stopped
DNS table size : 73, in use: 4, free: 94 %
DNS forwarding table size : 10, in use: 0, free:100 %
DNS forwarding dns servers table size : 25, in use: 4, free: 84 %
No dns cache.
Tracing: off
=>dns start
DNS server started.
=>dns status
DNS server status: Started
DNS table size : 73, in use: 4, free: 94 %
DNS forwarding table size : 10, in use: 0, free:100 %
DNS forwarding dns servers table size : 25, in use: 4, free: 84 %
No dns cache.
Tracing: off
=>
```

**RELATED COMMANDS:**

- **dns status**  
  Show DNS server/forwarder configuration.
- **dns stop**  
  Stop DNS server/forwarder.
**dns stats**

Show SpeedTouch™ Pro with Firewall DNS server/forwarder statistics.

**SYNTAX:**

```
dns stats
```

**EXAMPLE INPUT/OUTPUT IN A NETWORKED ENVIRONMENT:**

The SpeedTouch™ Pro with Firewall is configured as DNS server.

```
=>dns list
Domain: SpeedLAN.local
Nr. Hostname IP Address
0  SpeedTouch  *.*.*.*
1  Server  10.10.1.1
2  Client  10.0.0.3
Total Table Size: 73 entries
Amount used: 3 (4%)
=>dns stats
DNS Statistics:
Corrupted packets recv : 0
Local questions resolved : 1
Local neg answers sent : 0
Total DNS packets fwd : 0
External answers recv : 0
Fwd table full, discard : 0
Spurious answers : 0
Unknown query types : 0
Total number of packets received : 1
=>(PingClient.SpeedLAN.local)
=>(CTRL + Q)
dnsd: Internet class type A request received from 10.10.1.1.
dnsd: Client.SpeedLAN.local found in local database.
dnsd: Client.SpeedLAN.local resolved into 10.0.0.3.
=>(PingServer.SpeedLAN.local)
dnsd: Internet class type A request received from 10.10.1.1.
dnsd: Server.SpeedLAN.local found in local database.
dnsd: Server.SpeedLAN.local resolved into 10.0.0.3.
=>(CTRL + S)
=>dns stats
DNS Statistics:
Corrupted packets recv : 0
Local questions resolved : 3
Local neg answers sent : 0
Total DNS packets fwd : 0
External answers recv : 0
Fwd table full, discard : 0
Spurious answers : 0
Unknown query types : 0
Total number of packets received : 3
=>
```

**RELATED COMMANDS:**

- **dns clrstats** Clear DNS server/forwarder statistics.
**dns status**

Show SpeedTouch™ Pro with Firewall DNS server/forwarder configuration.

**SYNTAX:**

```
dns status
```

**EXAMPLE OUTPUT:**

```
=> dns status
DNS server status: Stopped
DNS table size : 73, in use: 4, free: 94 %
DNS forwarding table size : 10, in use: 0, free: 100 %
DNS forwarding dns servers table size : 25, in use: 4, free: 84 %
No dns cache.
Tracing: off
=>
```

**RELATED COMMANDS:**

- `dns flush` Flush complete DNS server/forwarder configuration and dynamic entries.
- `dns load` Load saved or default DNS server/forwarder configuration and static entries.
- `dns save` Save current DNS server/forwarder configuration and static entries.
**dns stop**
Stop SpeedTouch™ Pro with Firewall DNS server/forwarder.

**SYNTAX:**
```
 dns stop
```

**EXAMPLE:**

```bash
=> dns status
DNS server status: Started
  DNS table size : 73, in use: 4, free: 94 %
  DNS forwarding table size : 10, in use: 0, free: 100 %
  DNS forwarding dns servers table size : 25, in use: 4, free: 84 %
No dns cache.
Tracing: off

=> dns stop
DNS server stopped.

=> dns status
DNS server status: Stopped
  DNS table size : 73, in use: 4, free: 94 %
  DNS forwarding table size : 10, in use: 0, free: 100 %
  DNS forwarding dns servers table size : 25, in use: 4, free: 84 %
No dns cache.
Tracing: off
```

**RELATED COMMANDS:**
- **dns status** Show DNS server/forwarder configuration.
- **dns start** Start DNS server/forwarder.
**dns toutfwd**

Set DNS forwarding timeout.

**SYNTAX:**

```
dns toutfwd timeout = <number>
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>A number (seconds). Represents the query forwarding timeout. This parameter determines how long the <strong>SpeedTouch™ Pro with Firewall</strong> DNS server should try to contact a (remote) DNS server before (temporarily) declaring the DNS requests unresolved. By default the timeout is 15 seconds.</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

```
=>dns fwdtable
Forwarding table:
Nr. Ip Address (port#):id(hex) (expiry) dns server tries
0 10.10.10.12 (54751):8331 (13 sec) 10.10.10.112 1
Timeout: 15 seconds
Table size: 10
Amount of table used: 1 (10%)
=> dns toutfwd timeout=20
Timeout set to: 20 seconds
=> dns fwdtable
Forwarding table:
Nr. Ip Address (port#):id(hex) (expiry) dns server tries
0 10.10.10.12 (54751):8331 (13 sec) 10.10.10.112 1
Timeout: 20 seconds
Table size: 10
Amount of table used: 1 (10%)
=>
```

**RELATED COMMANDS:**

- `dns fwdtable`: Show DNS forwarding table.
- `dns fwdlist`: Show current DNS forwarding entries.
- `dns fwdadd`: Add a DNS forwarding entry.
- `dns fwddelete`: Delete a DNS forwarding entry.
**dns troff**

Disable verbose console messaging. No debug traces are generated.

**SYNTAX:**

`dns troff`

**EXAMPLE:**

```text
=>dns status
DNS server status: Started
DNS table size : 73, in use: 4, free: 94 %
DNS forwarding table size : 10, in use: 0, free:100 %
DNS forwarding dns servers table size : 25, in use: 4, free: 84 %
No dns cache.
Tracing: on
=>dns troff
=>dns status
DNS server status: Started
DNS table size : 73, in use: 4, free: 94 %
DNS forwarding table size : 10, in use: 0, free:100 %
DNS forwarding dns servers table size : 25, in use: 4, free: 84 %
No dns cache.
Tracing: off
=>
```

**RELATED COMMANDS:**

- `dns fwdtbl`: Show DNS forwarding table.
- `dns fwdlist`: Show current DNS forwarding entries.
- `dns status`: Show DNS server/forwarder configuration.
- `dns tron`: Enable verbose console messaging.
**dns tron**
Enable verbose console messaging. Debug traces are generated.

**SYNTAX:**

```plaintext
dns tron
```

**EXAMPLE:**

```
=>dns status
DNS server status: Started
DNS table size: 73, in use: 4, free: 94%
DNS forwarding table size: 10, in use: 0, free: 100%
DNS forwarding dns servers table size: 25, in use: 4, free: 84%
No dns cache.
Tracing: off
=>dns tron
Tracing on.
=>dns status
DNS server status: Started
DNS table size: 73, in use: 4, free: 94%
DNS forwarding table size: 10, in use: 0, free: 100%
DNS forwarding dns servers table size: 25, in use: 4, free: 84%
No dns cache.
Tracing: on
```

```
=>(CTRL + Q)
dnsd: Internet class type A request received from 10.0.0.10.
dnsd: aa.aa.be is outside our domain: forward.
dnsd: forwarding request from 10.0.0.10 (1318,0x0001) to 138.203.68.61
  (try=1): 'reply to ant' mode.
dnsd: Internet class type A request received from 10.0.0.10.
dnsd: aa.aa.be is outside our domain: forward.
dnsd: forwarding request from 10.0.0.10 (1318,0x0001) to 138.203.68.11
  (try=2): 'reply to ant' mode.
dnsd: forward answer from 138.203.68.11 to 10.0.0.10 (1318,0001).
dnsd: Internet class type A request received from 10.0.0.10.
dnsd: aa.aa.be.lan unknown: return error.
.....
=>(CTRL + S)
```

**RELATED COMMANDS:**

- `dns fwdtable`: Show DNS forwarding table.
- `dns fwdlist`: Show current DNS forwarding entries.
- `dns status`: Show DNS server/forwarder configuration.
- `dns troff`: Disable verbose console messaging.
8 Firewall Commands

firewall (to access the Firewall level)
firewall assign
firewall flush
firewall list
firewall load
firewall match
firewall save
firewall troff
firewall tron
firewall chain (to access the Firewall Chain level)
firewall chain create
firewall chain delete
firewall chain list
firewall chain load
firewall chain save
firewall rule (to access the Firewall Rule level)
firewall rule clear
firewall rule create
firewall rule delete
firewall rule flush
firewall rule list
firewall rule stats
**firewall assign**

Assign a chain to an entry point. An entry point, also referred to as hook or a Packet Interception Point (PIP) is the location where packets are intercepted to be compared against a chain of rules.

**SYNTAX:**

```plaintext
firewall assign  hook = <input|sink|forward|source|output> >
chain = <string>
```

- **hook**
  - The entry point’s name to assign a chain to.
  - Choose between:
    - **input**: The point off all incoming traffic.
      - At this point it can be determined whether the packet is allowed to reach the *SpeedTouch™ Pro with Firewall* IP router or local host.
    - **sink**: The point off all traffic destined to the *SpeedTouch™ Pro with Firewall* IP router itself.
      - At this point it can be determined whether the packet is allowed to address the local host.
    - **forward**: The point off all traffic to be forwarded by the *SpeedTouch™ Pro with Firewall* IP router.
      - At this point it can be determined whether the packet is allowed to be handled, i.e. routed.
    - **source**: The point off all traffic sourced by the *SpeedTouch™ Pro with Firewall* IP router.
      - At this point it can be determined whether the packet is allowed to leave the local host.
    - **output**: The point off all outgoing traffic.
      - At this point it can be determined whether the packet is allowed to leave the *SpeedTouch™ Pro with Firewall* IP router or local host.

- **chain**
  - The name of the chain to use.

**EXAMPLE:**

```plaintext
=>firewall list
assign hook=sink chain=sink
assign hook=forward chain=forward
assign hook=source chain=source
=>firewall chain create chain Telnet
=>firewall assign hook=sink chain=Telnet
=>firewall list
assign hook=sink chain=Telnet
assign hook=forward chain=forward
assign hook=source chain=source
=>
```

**RELATED COMMANDS:**

- **firewall chain create**: Create a chain.
- **firewall chain list**: Show a list of all current chains.
**firewall flush**

Flush all associations between a hook and its chain(s). The chain itself is not removed. The flush command does not impact previously saved configurations.

**SYNTAX:**

```
firewall flush [hook = \{input|sink|forward|source|output\}]  
```

<table>
<thead>
<tr>
<th>[hook]</th>
<th>the name of the hook to clear.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Optional</td>
</tr>
<tr>
<td>Choose between:</td>
<td></td>
</tr>
<tr>
<td>• input</td>
<td></td>
</tr>
<tr>
<td>• sink</td>
<td></td>
</tr>
<tr>
<td>• forward</td>
<td></td>
</tr>
<tr>
<td>• source</td>
<td></td>
</tr>
<tr>
<td>• output</td>
<td></td>
</tr>
</tbody>
</table>

In case this parameter is not specified all hooks are cleared.

**EXAMPLE:**

```
=>firewall list
assign hook=sink chain=sink
assign hook=forward chain=forward
assign hook=source chain=source
=>firewall flush hook=sink
=>firewall list
assign hook=forward chain=forward
assign hook=source chain=source
=>firewall flush
=>firewall list
=>
```

**RELATED COMMANDS:**

- **firewall assign**
  Assign a chain to an entry point.
firewall list
Show association(s) between all hooks and their chain(s) or of one specified hook

SYNTAX:

| firewall list | [hook = <input|sink|forward|source|output>] |
|---------------|-----------------------------------------|

[hook] the name of the hook to show the associations for. Choose between:
- input
- sink
- forward
- source
- output.
In case this parameter is not specified the associations for all hooks are shown.

EXAMPLE INPUT/OUTPUT:

```
=>firewall list
assign hook=sink chain=sink
assign hook=forward chain=forward
assign hook=source chain=source
=>firewall list hook=input
=>firewall list hook=forward
assign hook=forward chain=forward
=>
```

RELATED COMMANDS:
- `firewall assign` Assign a chain to an entry point.
- `firewall flush` Clear associations for all or a selected entry point(s).
**firewall load**

Load saved (or default) firewall configuration.
Execute `firewall flush` prior to `firewall load`.

**SYNTAX:**

```
firewall load [file = <string>]
[defaults = <yes|no>]
```

- **[file]** The name of the firewall configuration to be loaded.
  Not specifying this parameter loads the default configuration
  OPTIONAL

- **[defaults]** Load factory defaults (yes) or saved configuration (no).
  Not specifying this parameter loads the saved configuration
  OPTIONAL

**EXAMPLE:**

```bash
=>firewall list
assign hook=sink chain=sink
assign hook=forward chain=forward
assign hook=source chain=source
=>firewall save
=>firewall flush
=>firewall list
=>firewall load
=>firewall list
assign hook=sink chain=sink
assign hook=forward chain=forward
assign hook=source chain=source
=>
```

**RELATED COMMANDS:**

- **firewall save** Save current firewall configuration.
- **firewall flush** Clear associations for all or a selected entry point(s).
**firewall match**

Match a specified IP packet. Used to match an IP packet against a chain in order to determine what the reaction of the firewall would be.

This command can be considered as being the same as the **firewall rule create** command, but without the action to be taken.

**SYNTAX:**

```plaintext
firewall match
    chain = <string>
    [srcintf = <string>]
    [src = <ip-address>]
    [srcbridgeport = <number>]
    [dstintf = <string>]
    [dst = <ip-address>]
    [tos = <number>]
    [prot = <tcp|udp|icmp|protocol>]
    [syn = <yes|no>]
    [urg = <yes|no>]
    [ack = <yes|no>]
    [srcport = <ftp|ftp-data|telnet|mail|smtp|dns|domain|ftp|port>]
    [dstport = <ftp|ftp-data|telnet|mail|smtp|dns|domain|ftp|port>]
    [icmp-type = <deny|redirect|echo-request|router-advertisement|
                router-solicitation|time-exceeded|parameter-problems|
                timestamp-request|timestamp-reply|
                information-request|information-reply|
                address-mask-request|address-mask-reply|]
    [icmp-code = <icmpnumber>]
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required/Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>chain</td>
<td>The name of the chain to match the packet against.</td>
<td>REQUIRED</td>
</tr>
<tr>
<td>srcintf</td>
<td>The name of the interface the virtual packet arrived on.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>src</td>
<td>The source IP address the virtual packet is coming from.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>srcbridgeport</td>
<td>A number between 0 and 6. Represents the bridge port the virtual packet arrived on. Execute <strong>bridge iflist</strong> for a list of available bridge ports.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>dstintf</td>
<td>The name of the interface the virtual packet is going to.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>dst</td>
<td>The destination IP address the virtual packet is going to.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>tos</td>
<td>A number between 0 and 255. Represents the Type Of Service specification which should be expected [or NOT expected] in the IP packet. The Type of Service numbering specification is in accordance to the latest version of RFC1700: Assigned numbers.</td>
<td>OPTIONAL</td>
</tr>
</tbody>
</table>
**[prot]**  The protocol (name or number) in the virtual IP packet.  **OPTIONAL**  
Choose between:  
- tcp  
- udp  
- icmp  
Or specify the protocol number in accordance to the latest version of RFC1700: Assigned numbers.

**[syn]**  Set the TCP SYN flag (yes) or not (no).  **OPTIONAL**

**[urg]**  Set the TCP URG flag (yes) or not (no).  **OPTIONAL**

**[ack]**  Set the TCP ACK flag (yes) or not (no).  **OPTIONAL**

**[srcport]**  The TCP/UDP port the virtual packet is coming from.  **OPTIONAL**  
Choose between:  
- ftp  
- ftp-data  
- telnet  
- mail  
- smtp  
- dns  
- domain  
- tftp  
Or specify the port number in accordance to the latest version of RFC1700: Assigned numbers.

**[dstport]**  The TCP/UDP port the virtual packet is going to.  **OPTIONAL**  
Choose between:  
- ftp  
- ftp-data  
- telnet  
- mail  
- smtp  
- dns  
- domain  
- tftp  
Or specify the port number in accordance to the latest version of RFC1700: Assigned numbers.
**[icmptype]**

The ICMP (Internet Control Message Protocol) type (name or number) is optional of the virtual packet.
Choose between:
- echo-reply
- destination-unreachable
- source-quench
- redirect
- echo-request
- route-advertisement
- route-solicitation
- time-exceeded
- parameter-problems
- timestamp-request
- timestamp-reply
- information-request
- information-reply
- address-mask-request
- address-mask-reply
Or specify the ICMP type number in accordance to the latest version of RFC1700: Assigned numbers.

**[icmpcode]**

A number between 0 and 15. Represent the ICMP code of the virtual packet as specified in the latest version of RFC1700: Assigned number.

**EXAMPLE INPUT/OUTPUT:**

```
$ firewall rule list chain=Telnet
$ firewall rule create chain=Telnet index=0 src intfgrp=lan src=10.0.0.0/8 dst=200.200.200.1/32 prot=tcp srcport=1024 srcportend=65535 dstport=telnet action=accept
$ firewall rule create chain=Telnet index=1 src intfgrp=wan src=200.200.200.1/32 dst=10.0.0.0/8 prot=tcp srcport=telnet dstport=1024 dstportend=65535 action=accept
$ firewall rule create chain=Telnet index=2 action=drop
$ firewall match chain=Telnet src=200.200.200.1 dst=10.0.0.1 ack srcport=23 dstport=1023
Packet was ACCEPTED
```

**RELATED COMMANDS:**

```
firewall rule create
```

Create a firewall rule.
**firewall save**

Save all modifications entered by: `firewall assign`. This command only saves the association between hook(s) and chain(s), set by the `firewall assign` command, not the information about chains, rules and their parameters.

**SYNTAX:**

```plaintext
firewall save [file = <string>]  
```

`[file]` A name for the current firewall configuration file to be saved. The name is limited to 9 characters. This parameter allows multiple firewall configurations to be saved under different names. In case this parameter is not specified the configuration is saved as a single configuration.

**EXAMPLE:**

```
=>firewall list
assign hook=sink chain=sink
assign hook=forward chain=forward
assign hook=source chain=source
=>firewall assign hook input chain Test
=>firewall list
assign hook-input chain=Test
assign hook=sink chain=sink
assign hook=forward chain=forward
assign hook=source chain=source
=>firewall save file=TestCnfg
=>firewall flush
=>firewall list
=>firewall load file=TestCnfg
=>firewall list
assign hook-input chain=Test
assign hook=sink chain=sink
assign hook=forward chain=forward
assign hook=source chain=source
=>
```

**RELATED COMMANDS:**

- `firewall load` Load a saved or default firewall configuration.
- `firewall flush` Clear associations for all or a selected entry point(s).
- `firewall chain save` Save current chain(s) configuration.
**firewall troff**
Disable verbose console messaging.

**SYNTAX:**
```
firewall troff
```

**EXAMPLE:**
```
=>firewall troff
```

**RELATED COMMANDS:**
- **firewall tron**  
  Enable verbose console messaging.
**firewall tron**
Enable verbose console messaging.

**SYNTAX:**
```
fir**we**al tron
```

**EXAMPLE:**
```
=> firewall tron
```

**RELATED COMMANDS:**
- **firewall troff** Disable verbose console messaging.
**firewall chain create**

Create a new chain.

**SYNTAX:**

```
firewall chain create  chain = <string>
```

chain

The name of the chain to create.  

**EXAMPLE:**

```
=>firewall chain list
Tempo, source, forward, sink
=>firewall chain create chain=Telnet
=>firewall chain list
Telnet, Tempo, source, forward, sink
=>
```

**RELATED COMMANDS:**

- **firewall assign**
  Assign a chain to an entry point.
- **firewall chain delete**
  Delete a chain.
- **firewall chain list**
  Show a list of all current chains.
**firewall chain delete**

Delete a chain.

**SYNTAX:**

```
firewall chain delete chain = <string>
```

| chain        | The name of the chain to be deleted. | REQUIRED |

**EXAMPLE:**

```
=>firewall chain list
Telnet, Tempo, source, forward, sink
=>firewall chain list
Telnet, Tempo, source, forward, sink
=>firewall chain delete chain=Tempo
=>firewall chain list
Telnet, source, forward, sink
=>
```

**RELATED COMMANDS:**

- **firewall assign** Assign a chain to an entry point.
- **firewall chain create** Create a chain.
- **firewall chain list** Show a list of all chains.
firewall chain list
Show a list of all current chains.

SYNTAX:

```
firewall chain list
```

EXAMPLE INPUT/OUTPUT:

```
=>firewall chain list
source, forward, sink
=>firewall chain create chain Telnet
=>firewall chain list
Telnet, source, forward, sink
=>firewall chain list
Telnet, source, forward, sink
=>
```

RELATED COMMANDS:

- firewall assign
  - Assign a chain to an entry point.
- firewall chain create
  - Create a chain.
- firewall chain delete
  - Delete a chain.
**firewall chain load**
Load saved or default chain(s) configuration (with related rules).

**SYNTAX:**

<table>
<thead>
<tr>
<th>firewall chain load</th>
<th>[file = &lt;string&gt;]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[defaults = &lt;yes</td>
</tr>
</tbody>
</table>

- **[file]**
  - The name of the chain configuration to be loaded.
  - Not specifying this parameter loads the default configuration
  - **OPTIONAL**

- **[defaults]**
  - Load factory defaults (yes) or saved configuration (no).
  - Not specifying this parameter loads the saved configuration
  - **OPTIONAL**

**EXAMPLE:**

```
->firewall flush
->firewall load
->firewall chain list
source, forward, sink
->firewall flush
=>firewall chain load file TelConfig
->firewall chain list
sink, forward, source, Telnet
->
```

**RELATED COMMANDS:**

- **firewall assign**
  - Assign a chain to an entry point.
- **firewall chain create**
  - Create a chain.
- **firewall chain delete**
  - Delete a chain.
- **firewall chain list**
  - Show a list of all current chains.
- **firewall chain save**
  - Save current chain(s) configuration.
**firewall chain save**

Save all modifications entered by: **firewall chain create, firewall chain delete**.  
This command saves the information about chains, rules and their parameters. This is different from **firewall save** which saves only the association between hook(s) and chain(s), set by the **firewall assign** command.

**SYNTAX:**

```
firewall chain save  [file = <string>]
```

*file*  
A name for the current chain configuration file to be saved. 
**OPTIONAL**  
The name is limited to 9 characters. 
This parameter allows multiple chain configurations to be saved under different names. 
In case this parameter is not specified the configuration is saved as single configuration.

**EXAMPLE:**

```
=>firewall flush
=>firewall chain load file TelConfig
=>firewall chain list
Telnet, source, forward, sink
=>firewall chain create chain NewTel
=>firewall chain list
NewTel, Telnet, source, forward, sink
=>firewall chain save file NewTelcnf
=>firewall flush
=>firewall chain load file NewTelcnf
=>firewall chain list
sink, forward, source, Telnet, NewTel
=>
```

**RELATED COMMANDS:**

- **firewall assign**  
  Assign a chain to an entry point.  
- **firewall chain create**  
  Create a chain.  
- **firewall chain delete**  
  Delete a chain.  
- **firewall chain list**  
  Show a list of all current chains.  
- **firewall chain load**  
  Load saved or default chain configuration(s).
firewall rule clear

Clear statistics for a given rule.

SYNTAX:

```
firewall rule clear [chain = <string>] [index = <number>]
```

- **[chain]**: The name of the chain in which the rule is to be found. **OPTIONAL**
- **[index]**: The index number (determined by the position) of the rule in the chain. **OPTIONAL**

EXAMPLE:

```
=>firewall rule stats
ChainTelnet, index0, packets 0, bytes 0
ChainTelnet, index1, packets 0, bytes 0
ChainTelnet, index2, packets 0, bytes 0
Chainsource, index0, packets 203, bytes 15229
Chainsource, index1, packets 0, bytes 0
Chainsource, index2, packets 0, bytes 0
Chainforward, index0, packets 0, bytes 0
Chainsink, index0, packets 202, bytes 10159
Chainsink, index1, packets 0, bytes 0
Chainsink, index2, packets 0, bytes 0
=>firewall rule clear chain=source index=0
=>firewall rule stats
ChainTelnet, index0, packets 0, bytes 0
ChainTelnet, index1, packets 0, bytes 0
ChainTelnet, index2, packets 0, bytes 0
Chainsource, index0, packets 11, bytes 559
Chainsource, index1, packets 0, bytes 0
Chainsource, index2, packets 0, bytes 0
Chainforward, index0, packets 0, bytes 0
Chainsink, index0, packets 409, bytes 21535
Chainsink, index1, packets 0, bytes 0
Chainsink, index2, packets 0, bytes 0
=>
```

RELATED COMMANDS:

- **firewall rule create**: Create a rule.
- **firewall rule delete**: Delete a specified rule in a chain.
- **firewall rule flush**: Delete all rules in a chain.
- **firewall rule list**: Show a list of all (or a specified) chains’ rules.
- **firewall rule stats**: Show statistics for all (or a specified) chains’ rules.
firewall rule create
Create a rule.

SYNTAX:

```
firewall rule create
    chain = <string>
    [index = <number>]
    [srcintf [l]= <string>]
    [srcintfgrp [l]= \{wan|local|lan\}]  
    [srcbridgeport [l]= <number>]
    [src [l]= <ip-address>]
    [srcmsk = <ip-mask(dotted or cidr)>]
    [dstintf [l]= <string>]
    [dstintfgrp [l]= \{wan|local|lan\}]  
    [dst [l]= <ip-address>]
    [dstmsk = <ip-mask(dotted or cidr)>]
    [tos [l]= <number\{1-255\}>]
    [prot [l]= \{tcp|udp|icmp|protocol\}>]
    [syn <yes|no>]
    [urg <yes|no>]
    [ack <yes|no>]
    [srcport [l]= \{ftp|ftp-data|telnet|mail|smtp|dns|domain|ftp|port\}>]
    [srcportend = \{ftp|ftp-data|telnet|mail|smtp|dns|domain|ftp|port\}>]
    [dstport [l]= \{ftp|ftp-data|telnet|mail|smtp|dns|domain|ftp|port\}>]
    [dstportend = \{ftp|ftp-data|telnet|mail|smtp|dns|domain|ftp|port\}>]
    [icmptype [l]= \{echo-reply|destination-unreachable|source-quench|
                      redirect|echo-request|router-advertisement| router-solicitation|time-exceeded|parameter-problems |
                      timestamp-request|timestamp-reply| information-request|information-reply |
                      address-mask-request|address-mask-reply|  icmpnumber\}>]
    [icmpcode [l]= <number\{0-15\}>]
    [icmpcodeend = <number\{0-15\}>]
    [clink = <string>]
    action = \{accept|deny|drop|count\}>  
```

- **chain**: The name of the chain to insert the rule in. **REQUIRED**
- **[index]**: The number of the rule before which the new rule must be added. **OPTIONAL**
- **[srcintf]**: The name of the interface the packet should [or should NOT] arrive on to make this rule apply. **OPTIONAL**
  (NOT applicable if used in a chain assigned to the output hook)
- **[srcintfgrp]**: The interface group the packet should [or should NOT] arrive on. **OPTIONAL**
  Choose between:
  - wan
  - local
  - lan
  (NOT applicable if used in a chain assigned to the output hook)
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>[srcbridgeport]</td>
<td>A number between 0 and 6. Represents the bridge port the virtual packet should [or should NOT] arrive on. Execute <code>bridge iflist</code> for a list of available bridge ports.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>[src]</td>
<td>The source IP address (range) the packet should [or should NOT] come from. (Supports cidr notation).</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>[srcmsk]</td>
<td>The source IP address mask defining the range (see src).</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>[dstintf]</td>
<td>The name of the interface the packet should [or should NOT] be going to. (NOT applicable if used in a chain assigned to the input hook)</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>[dstintfgrp]</td>
<td>The interface group the packet should [or should NOT] be going to. Choose between: <code>wan</code>, <code>local</code>, <code>lan</code> (NOT applicable if used in a chain assigned to the input hook)</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>[dst]</td>
<td>The destination IP address (range) the packet should [or should NOT] be going to. (supports cidr notation).</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>[dstmsk]</td>
<td>The destination IP address mask defining the range (see dst).</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>[tos]</td>
<td>A number between 0 and 255. Represents the Type Of Service specification which should be expected [or NOT expected] in the IP packet. The Type of Service numbering specification is in accordance to the latest version of RFC1700: Assigned numbers.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>[prot]</td>
<td>The protocol (name or number) in the IP packet expected [or NOT expected] in the IP packet. Choose between: <code>tcp</code>, <code>udp</code>, <code>icmp</code> Or specify the protocol number in accordance to the latest version of RFC1700: Assigned numbers.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>[syn]</td>
<td>Expect TCP SYN flag set (yes) or not (no). In combination with TCP ACK this allows selection of incoming versus outgoing TCP connections.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>[urg]</td>
<td>Expect TCP URG flag set (yes) or not (no).</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>[ack]</td>
<td>Expect TCP ACK flag set (yes) or not (no).</td>
<td>OPTIONAL</td>
</tr>
</tbody>
</table>
[srcport] The TCP/UDP port (or beginning of range) the packet should [or should NOT] be from. Choose between:
- ftp
- ftp-data
- telnet
- mail
- smtp
- dns
- domain
- tftp
Or specify the port number in accordance to the latest version of RFC1700: Assigned numbers.

[srcportend] The source TCP/UDP port range end (inclusive). (Only applicable for ranges) OPTIONAL

[dstport] The TCP/UDP port (or beginning of range) the packet should [or should NOT] be going to. Choose between:
- ftp
- ftp-data
- telnet
- mail
- smtp
- dns
- domain
- tftp
Or specify the port number in accordance to the latest version of RFC1700: Assigned numbers.

[dstportend] The destination TCP/UDP port range end (inclusive). (Only applicable for ranges) OPTIONAL

[icmptype] The expected [or NOT expected] ICMP type (name or number) of the packet. Choose between:
- echo-reply
- destination-unreachable
- source-quench
- redirect
- echo-request
- router-advertisement
- router-solicitation
- time-exceeded
- parameter-problems
- timestamp-request
- timestamp-reply
- information-request
- information-reply
- address-mask-request
- address-mask-reply
Or specify the ICMP type number in accordance to the latest version of RFC1700: Assigned numbers.
[icmpcode]  A number between 0 and 15.  
Represents the expected [or NOT expected] ICMP code (or beginning of range) of the packet as specified in the latest version of RFC1700: Assigned number.

[icmpcodeend]  A number between 0 and 15.  
Represents the ICMP code range end. Only applicable for ranges.

[clink]  The name of the chain to be parsed when this rule applies. (action is ignored).

action  Action to be taken when this rule applies. Required
Choose between:
- **accept**: the packet may pass.
- **deny**: ICMP error destination unreachable. An error message is sent back to the sender.
- **drop**: packet disappears. It is silently dropped, that is, without sending an error message to the sender.
- **count**: update of statistics. Has no influence on the packet.

**EXAMPLE:**

```plaintext
$ firewall rule list chain=Telnet
$ firewall rule create chain=telnet src=10.0.0.0/8 dst=200.200.200.1 srcintfgrp=lan
   prot=tcp  srcport=1024  dstportend=65535 dstport=23
   action=accept
$ firewall rule create chain=telnet src=200.200.200.1 dst=10.0.0.0/8 srcintfgrp=wlan
   prot=tcp  srcportend=23  dstport=1024  dstportend=65535
   action=accept
$ firewall rule create chain=telnet src=10.0.0.0/8 dst=200.200.200.1 srcintfgrp=wan
   prot=tcp  srcport=23  dstport=1024  dstportend=65535
   action=drop
$ firewall rule list chain=Telnet
:firewall rule create chain=Telnet index=0 srcintfgrp=lan src=10.0.0.0/8
   dst=200.200.200.1/32 prot=tcp  srcport=1024  srcportend=65535 dstport=telnet
   action=accept
:firewall rule create chain=Telnet index=1 srcintfgrp=wlan src=200.200.200.1/32
   dst=10.0.0.0/8 prot=tcp  srcport=telnet  dstport=1024  dstportend=65535
   action=accept
:firewall rule create chain=Telnet index=2
   action=drop

$ ->
```

**RELATED COMMANDS:**

- **firewall rule clear**  Clear statistics of a given rule.
- **firewall rule delete**  Delete a specified rule in a chain.
- **firewall rule flush**  Delete all rules in a chain.
- **firewall rule list**  Show a list of all (or a specified) chains’ rules.
- **firewall rule stats**  Show statistics for all (or a specified) chains’ rules.
**firewall rule delete**

Delete a rule.

**SYNTAX:**

```
firewall rule delete  
  chain = <string>  
  index = <number>
```

- **chain** The name of the chain in which to delete the rule. **REQUIRED**
- **index** The index number of the rule in the chain. **REQUIRED**

**EXAMPLE:**

```bash
=>firewall rule list chain=Telnet
:firewall rule create chain=Telnet index=0 srcintfgrp=lan src=10.0.0.0/8 dst=200.200.200.1/32 prot=tcp srcport=1024 srcportend=65535 dstport=telnet action=accept
:firewall rule create chain=Telnet index=1 srcintfgrp=wan src=200.200.200.1/32 dst=10.0.0.0/8 prot=tcp srcport=telnet dstport=1024 dstportend=65535 action=accept
:firewall rule create chain=Telnet index=2 action=drop
=>firewall rule delete chain=Telnet index=1
=>firewall rule list chain=Telnet
:firewall rule create chain=Telnet index=0 srcintfgrp=lan src=10.0.0.0/8 dst=200.200.200.1/32 prot=tcp srcport=1024 srcportend=65535 dstport=telnet action=accept
:firewall rule create chain=Telnet index=1 action=drop
=>
```

**RELATED COMMANDS:**

- **firewall rule clear** Clear statistics of a given rule.
- **firewall rule create** Create a rule.
- **firewall rule flush** Delete all rules in a chain.
- **firewall rule list** Show a list of all (or a specified) chains’ rules.
- **firewall rule stats** Show statistics for all (or a specified) chains’ rules.
**firewall rule flush**

Flush all rules created for a chain(s). The chain itself is not removed. The flush command does not impact previously saved configurations.

**SYNTAX:**

```
firewall rule flush [chain = <string>]
```

- **[chain]** The name of the chain to empty.
  - In case this parameter is not specified all rules for all chains are deleted.

**EXAMPLE:**

```
=>firewall rule list chain=Telnet
:firewall rule create chain=Telnet index=0 srcintfgrp=lan src=10.0.0.0/8 dst=200.200.200.1/32 prot=tcp srcport=1024 srcportend=65535 dstport=telnet action=accept
:firewall rule create chain=Telnet index=1 srcintfgrp=wan src=200.200.200.1/32 dst=10.0.0.0/8 prot=tcp srcport=telnet dstport=1024 dstportend=65535 action=accept
:firewall rule create chain=Telnet index=2 action=drop
=>firewall rule flush chain=Telnet
=>firewall rule list chain=Telnet
=>
```

**RELATED COMMANDS:**

- **firewall rule clear** Clear statistics of a given rule.
- **firewall rule create** Create a rule.
- **firewall rule delete** Delete a specified rule.
- **firewall rule list** Show a list of all (or a specified) chains’ rules.
- **firewall rule stats** Show statistics for all (or a specified) chains’ rules.
firewall rule list
Show a list of rules.

SYNTAX:

```
firewall rule list [chain = <string>]
```

<table>
<thead>
<tr>
<th>[chain]</th>
<th>The name of the chain to list the rules of. In case this parameter is not specified all rules for all chains are shown.</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTIONAL</td>
<td>----------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>

EXAMPLE INPUT AND OUTPUT:

```
=>firewall rule list chain=Telnet
:firewall rule create chain=Telnet index=0 srcintfgrp=lan src=10.0.0.0/8 dst=200.200.200.1/32 prot=tcp srcport=1024 srcportend=65535 dstport=telnet action=accept
:firewall rule create chain=Telnet index=1 srcintfgrp=wan src=200.200.200.1/32 dst=10.0.0.0/8 prot=tcp srcport=telnet dstport=1024 dstportend=65535 action=accept
:firewall rule create chain=Telnet index=2 action=drop

=>
```

RELATED COMMANDS:

- firewall rule clear: Clear statistics of a given rule.
- firewall rule create: Create a rule.
- firewall rule delete: Delete a specified rule.
- firewall rule flush: Delete all rules in a chain.
- firewall rule stats: Show statistics for all (or a specified) chains’ rules.
**firewall rule stats**
Show statistics, i.e. the number of packets and bytes which have passed the hooks.

**SYNTAX:**
```
firewall rule stats [chain = <string>]
[index = <number>]
```

[chain] The name of the chain of which the statistics must be listed. In case this parameter is not specified the statistics for the rules applicable to all chains are shown. OPTIONAL

[index] The index number of the chain’s rule of which the statistics must be listed. Execute firewall rule list first to determine the index number of the applicable rule. In case this parameter is not specified the statistics for all rules applicable to the specified chain are shown. OPTIONAL

**EXAMPLE OUTPUT:**
```
=> firewall rule list chain=Test
:firewall rule create chain=Test index=0 srcintfgrp=lan src=200.200.0.1/32
dst=200.200.0.2/32 prot=udp srcport=0 srcportend=65535 dstport=telnet
action=deny
=> firewall rule clear
=> firewall rule stats
Chainsink, index0, packets 43, bytes 1743
Chainsink, index1, packets 0, bytes 0
Chainsink, index2, packets 0, bytes 0
Chainsink, index3, packets 0, bytes 0
Chainforward, index0, packets 0, bytes 0
Chainsource, index0, packets 43, bytes 1977
Chainsource, index1, packets 0, bytes 0
Chainsource, index2, packets 0, bytes 0
ChainTest, index0, packets 0, bytes 0
=> firewall rule stats
Chainsink, index0, packets 104, bytes 6143
Chainsink, index1, packets 0, bytes 0
Chainsink, index2, packets 0, bytes 0
Chainsink, index3, packets 0, bytes 0
Chainforward, index0, packets 0, bytes 0
Chainsource, index0, packets 43, bytes 1977
Chainsource, index1, packets 0, bytes 0
Chainsource, index2, packets 0, bytes 0
ChainTest, index0, packets 44, bytes 21032
=>
```

**DESCRIPTION:**
The statistics for the ‘Test’ chain are the result of sending udp packets to the **SpeedTouch™ Pro with Firewall**. The chain ‘Test’ is assigned to the hook ‘input’ and prohibits the sending of udp packets from one host to another.
EXAMPLE INPUT/OUTPUT IN A NETWORKED ENVIRONMENT:

The SpeedTouch™ Pro with Firewall is configured as DHCP client on its Ethernet interface eth0.

```plaintext
=>firewall rule list chain=Sending
:firewall rule create chain=Sending index=0 srcintfgrp=lan src=10.0.0.3/32 dst=10.10.1.1/32 prot=icmp action=count
:firewall rule create chain=Sending index=1 srcintfgrp=lan src=10.10.1.1/32 dst=10.0.0.3/32 prot=icmp action=count

=>firewall rule stats
Chain source, index 0, packets 0, bytes 0
Chain source, index 1, packets 0, bytes 0
Chain source, index 2, packets 0, bytes 0
Chain source, index 3, packets 0, bytes 0
Chain forward, index 0, packets 0, bytes 0
Chain sink, index 0, packets 0, bytes 0
Chain sink, index 1, packets 144, bytes 5844
Chain sink, index 2, packets 0, bytes 0
Chain sink, index 3, packets 0, bytes 0
Chain sink, index 4, packets 0, bytes 0
Chain sink, index 5, packets 0, bytes 0
Chain sending, index 0, packets 0, bytes 0
Chain sending, index 1, packets 0, bytes 0

=>firewall rule clear
=>(Ping from server 10.10.1.1 to client 10.0.0.3)

=>firewall rule stats
Chain source, index 0, packets 0, bytes 0
Chain source, index 1, packets 0, bytes 0
Chain source, index 2, packets 0, bytes 0
Chain source, index 3, packets 0, bytes 0
Chain forward, index 0, packets 0, bytes 0
Chain sink, index 0, packets 0, bytes 0
Chain sink, index 1, packets 42, bytes 1782
Chain sink, index 2, packets 0, bytes 0
Chain sink, index 3, packets 0, bytes 0
Chain sink, index 4, packets 0, bytes 0
Chain sink, index 5, packets 0, bytes 0
Chain sending, index 0, packets 4, bytes 240
Chain sending, index 1, packets 4, bytes 240

=>
```

RELATED COMMANDS:

- `firewall rule clear`: Clear statistics of a given rule.
- `firewall rule create`: Create a rule.
- `firewall rule delete`: Delete a specified rule.
- `firewall rule flush`: Delete all rules in a chain.
- `firewall rule list`: Show a list of all (or a specified) chains’ rules.
9 IP Commands

ip (to access the IP level)
ip apadd
ip apdelete
ip aplist
ip arpadd
ip arpdelete
ip arplist
ip config
ip flush
ip ifconfig
ip iflist
ip load
ip ping
ip rtadd
ip rtdelete
ip rtlist
ip save
ip sendto
**ip apadd**

Assign an IP address to an interface.

**SYNTAX:**

```
ip apadd
    addr = <ip-address>
    [netmask = <ip-mask (dotted or cidr)>]
    intf = <interface name>
    [pointopoint = <ip-address>]
    [broadcastip = <ip-address>]
    [addrtrans = <{none|pat}>]
    [addroute = <{no|yes}>]
    [type = <number>]
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required/Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addr</code></td>
<td>The new IP address to add.</td>
<td>REQUIRED</td>
</tr>
<tr>
<td><code>netmask</code></td>
<td>The subnetmask associated with this address.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td><code>intf</code></td>
<td>The interface name.</td>
<td>REQUIRED</td>
</tr>
<tr>
<td><code>pointopoint</code></td>
<td>The remote IP address in case of a dedicated point-to-point link.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td><code>broadcastip</code></td>
<td>The broadcast IP address. For internal use only.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td><code>addrtrans</code></td>
<td>Indicates whether network address translation mode is allowed (pat) for this IP address or not (none).</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td><code>addroute</code></td>
<td>Add typical net/subnet routes automatically according to the default (or specified) subnet mask (yes) or not (no).</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td><code>type</code></td>
<td>The type of address classification. For internal use only.</td>
<td>OPTIONAL</td>
</tr>
</tbody>
</table>
EXAMPLE:

```bash
=> ip aplist
1  eth0  Type:EthernetHWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:ff
    inet addr: 10.10.10.147 Bcast: 10.10.10.255 Mask: 255.0.0.0
    UP RUNNING pat MTU:1500 ReasmMAX:65535 Group:2
    IPRX bytes:19791886 unicastpkts:11341 brcastpkts:290555 droppkts:0
    HWRX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
    HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
0   loop  Type:0
    inet addr:127.0.0.1 Bcast:127.255.255.255 Mask:255.0.0.0
    UP RUNNING MTU:1500 ReasmMAX:65535 Group:1
    IPRX bytes:116 unicastpkts:0 brcastpkts:2
    IPTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
    HWRX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
    HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
=> ip apadd addr=10.0.0.2 netmask=255.255.255.0 intf=eth0 addrtrans=pat addroute=yes
=> ip aplist
2  eth0  Type:EthernetHWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:ff
    inet addr: 10.0.0.2 Bcast: 10.0.0.255 Mask: 255.255.255.0
    UP RUNNING pat MTU:1500 ReasmMAX:65535 Group:2
    IPRX bytes:0 unicastpkts:0 brcastpkts:0
    IPTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
    HWRX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
    HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
1  eth0  Type:EthernetHWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:ff
    inet addr: 10.10.10.147 Bcast: 10.10.10.255 Mask: 255.0.0.0
    UP RUNNING pat MTU:1500 ReasmMAX:65535 Group:2
    IPRX bytes:19810763 unicastpkts:11515 brcastpkts:290669
    IPTX bytes:853114 unicastpkts:11662 brcastpkts:0 droppkts:0
    HWRX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
    HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
0   loop  Type:0
    inet addr:127.0.0.1 Bcast:127.255.255.255 Mask:255.0.0.0
    UP RUNNING MTU:1500 ReasmMAX:65535 Group:1
    IPRX bytes:116 unicastpkts:0 brcastpkts:2
    IPTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
    HWRX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
    HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
=>
```

RELATED COMMANDS:

- `ip apdelete` Remove an IP address from an interface.
- `ip aplist` Show current IP addresses.
**ip apdelete**

Remove an IP address from an interface.

**SYNTAX:**

```
ip apdelete addr = <ip-address>
```

- **addr** The IP address to delete. REQUIRED

**EXAMPLE:**

```bash
=> ip aplist
2 eth0  Type:EthernetHWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:
    inet addr: 10.0.0.2 Bcast: 10.0.0.255 Mask: 255.255.255.0
    UP RUNNING pat MTU:1500 ReasmMAX:65535 Group:2
    IPRX bytes:0 unicastpkts:0 bcastpkts:0
    IPTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
    HWRX bytes:0 unicastpkts:0 bcastpkts:0
    HWTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
1 eth0  Type:EthernetHWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:
    inet addr: 10.10.10.147 Bcast: 10.10.10.255 Mask: 255.0.0.0
    UP RUNNING pat MTU:1500 ReasmMAX:65535 Group:2
    IPRX bytes:839550 unicastpkts:11477 bcastpkts:0 droppkts:0
    IPTX bytes:0 unicastpkts:0 bcastpkts:0
    HWRX bytes:0 unicastpkts:0 bcastpkts:0
    HWTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
0 loop  Type:0
    inet addr:127.0.0.1 Bcast:127.255.255.255 Mask:255.0.0.0
    UP RUNNING MTU:1500 ReasmMAX:65535 Group:1
    IPRX bytes:116 unicastpkts:0 bcastpkts:2
    IPTX bytes:0 unicastpkts:0 bcastpkts:0
    HWRX bytes:0 unicastpkts:0 bcastpkts:0
    HWTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
=> ip apdelete addr=10.0.0.2
=> ip aplist
1 eth0  Type:EthernetHWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:
    inet addr: 10.10.10.147 Bcast: 10.10.10.255 Mask: 255.0.0.0
    UP RUNNING pat MTU:1500 ReasmMAX:65535 Group:2
    IPRX bytes:839550 unicastpkts:11477 bcastpkts:0 droppkts:0
    IPTX bytes:0 unicastpkts:0 bcastpkts:0
    HWRX bytes:0 unicastpkts:0 bcastpkts:0
    HWTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
0 loop  Type:0
    inet addr:127.0.0.1 Bcast:127.255.255.255 Mask:255.0.0.0
    UP RUNNING MTU:1500 ReasmMAX:65535 Group:1
    IPRX bytes:116 unicastpkts:0 bcastpkts:2
    IPTX bytes:0 unicastpkts:0 bcastpkts:0
    HWRX bytes:0 unicastpkts:0 bcastpkts:0
    HWTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
=>
```

**RELATED COMMANDS:**

- **ip apadd** Add an IP address to an interface.
- **ip aplist** Show current IP addresses.
**ip aplist**
Show a list of all configured IP addresses.

**SYNTAX:**

```
ip aplist
```

**EXAMPLE:**

```
=>ip aplist
2  eth0   Type:Ethernet HWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:ff
   inet addr:10.0.0.2   Bcast: 10.0.0.255  Mask: 255.255.255.0
  UP  RUNNING  mtu:1500  ReasmMAX:65535  Group:2
  IPRX bytes:0  unicastpkts:0  bcastpkts:0  drcastpkts:0
  IPTX bytes:0  unicastpkts:0  bcastpkts:0  drcastpkts:0
  HWRX bytes:0  unicastpkts:0  bcastpkts:0  drcastpkts:0
  HTTX bytes:0  unicastpkts:0  bcastpkts:0  drcastpkts:0
1  eth0   Type:Ethernet HWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:ff
   inet addr:10.10.10.147  Bcast: 10.10.10.255  Mask: 255.0.0.0
  UP  RUNNING  mtu:1500  ReasmMAX:65535  Group:2
  IPRX bytes:19791886  unicastpkts:11341  bcastpkts:290555
  IPTX bytes:839550  unicastpkts:11477  bcastpkts:0  drcastpkts:0
  HWRX bytes:0  unicastpkts:0  bcastpkts:0  drcastpkts:0
  HTTX bytes:0  unicastpkts:0  bcastpkts:0  drcastpkts:0
0  loop   Type:0
   inet addr:127.0.0.1  Bcast:127.255.255.255  Mask: 255.0.0.0
  UP  RUNNING  mtu:1500  ReasmMAX:65535  Group:1
  IPRX bytes:116  unicastpkts:0  bcastpkts:2
  IPTX bytes:0  unicastpkts:0  bcastpkts:0  drcastpkts:0
  HWRX bytes:0  unicastpkts:0  bcastpkts:0  drcastpkts:0
  HTTX bytes:0  unicastpkts:0  bcastpkts:0  drcastpkts:0
=>
```

**RELATED COMMANDS:**

- **ip apadd** Add an IP address to an interface.
- **ip apdelete** Remove an IP address from an interface.
**ip arpadd**

Add a static entry to the SpeedTouch™ Pro with Firewall ARP cache.

**SYNTAX:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip arpadd</td>
<td>ip = &lt;interface name&gt;</td>
</tr>
<tr>
<td></td>
<td>ip = &lt;ip-address&gt;</td>
</tr>
<tr>
<td></td>
<td>[hwaddr = &lt;hardware-address&gt;]</td>
</tr>
</tbody>
</table>

- **inf**  The interface name. REQUIRED
- **ip**   The IP address. REQUIRED
- **[hwaddr]** The hardware address (e.g. the Ethernet MAC address). OPTIONAL

**EXAMPLE:**

`=> ip arplist`

<table>
<thead>
<tr>
<th>Intf</th>
<th>IP–address</th>
<th>HW–address</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>eth0</td>
<td>10.0.0.1</td>
<td>00:01:42:5f:7d:81</td>
<td>DYNAMIC</td>
</tr>
<tr>
<td>eth0</td>
<td>10.0.0.8</td>
<td>00:a0:24:ae:66:e1</td>
<td>DYNAMIC</td>
</tr>
<tr>
<td>eth0</td>
<td>10.0.1.99</td>
<td>52:41:53:20:20:4d</td>
<td>STATIC</td>
</tr>
<tr>
<td>eth0</td>
<td>10.0.1.100</td>
<td>52:41:53:20:06:90</td>
<td>STATIC</td>
</tr>
</tbody>
</table>

`=> ip arpadd intf=eth0 ip=10.0.0.2 hwaddr=00:10:a4:d0:9a:db`

`=> ip arplist`

<table>
<thead>
<tr>
<th>Intf</th>
<th>IP–address</th>
<th>HW–address</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>eth0</td>
<td>10.0.0.1</td>
<td>00:01:42:5f:7d:81</td>
<td>DYNAMIC</td>
</tr>
<tr>
<td>eth0</td>
<td>10.0.0.8</td>
<td>00:a0:24:ae:66:e1</td>
<td>DYNAMIC</td>
</tr>
<tr>
<td>eth0</td>
<td>10.0.1.99</td>
<td>52:41:53:20:20:4d</td>
<td>STATIC</td>
</tr>
<tr>
<td>eth0</td>
<td>10.0.1.100</td>
<td>52:41:53:20:06:90</td>
<td>STATIC</td>
</tr>
<tr>
<td>eth0</td>
<td>10.0.0.2</td>
<td>00:10:a4:d0:9a:db</td>
<td>STATIC</td>
</tr>
</tbody>
</table>

**RELATED COMMANDS:**

- **ip arpdelete**      Delete an ARP entry.
- **ip arplist**        Show current ARP cache.
**ip arpdelete**

Remove an entry from the SpeedTouch™ Pro with Firewall ARP cache.

**SYNTAX:**

```plaintext
ip arpdelete

intf = <interface name>
ip = <ip-address>
[hwaddr = <hardware-address>]```

**EXEMPLARY:**

```
=> ip arplist
Intf  IP-address  HW-address  Type
eth0  10.0.0.1    00:01:42:5f:7d:81  DYNAMIC
eth0  10.0.0.8    00:a0:24:ae:66:e1  DYNAMIC
eth0  10.0.1.99   52:41:53:20:20:4d  STATIC
eth0  10.0.1.100  52:41:53:20:f0:90  STATIC
eth0  10.0.0.2    00:10:a4:d0:9a:db  STATIC
=> ip arpdelete intf=eth0 ip=10.0.0.2 hwaddr=00:10:a4:d0:9a:db
=> ip arplist
Intf  IP-address  HW-address  Type
eth0  10.0.0.1    00:01:42:5f:7d:81  DYNAMIC
eth0  10.0.0.8    00:a0:24:ae:66:e1  DYNAMIC
eth0  10.0.1.99   52:41:53:20:20:4d  STATIC
eth0  10.0.1.100  52:41:53:20:f0:90  STATIC
eth0  10.0.0.2    00:10:a4:d0:9a:db  STATIC
=>
```

**RELATED COMMANDS:**

- **ip arpad**
  - Add a static ARP entry.
- **ip arplist**
  - Show current ARP cache.
**ip arplist**
Show the SpeedTouch™ Pro with Firewall ARP cache.

**SYNTAX:**

`ip arplist`

**EXAMPLE OUTPUT:**

```
=> ip arplist
Intf  IP-address   HW-address      Type
eth0  10.0.0.1     00:01:42:5f:7d:81  DYNAMIC
eth0  10.0.0.8     00:a0:24:ae:66:e1  DYNAMIC
eth0  10.0.1.99    52:41:53:20:20:4d  STATIC
eth0  10.0.1.100   52:41:53:20:f0:90  STATIC
eth0  10.0.0.2     00:10:a4:d0:9a:db  STATIC
=>
```

**RELATED COMMANDS:**

- `ip arpadd` Add a static entry to the ARP cache.
- `ip arpdelete` Delete an entry from the ARP cache.
ip config
Show/set global IP stack configuration options.

SYNTAX:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>forwarding = off</td>
<td>Disable (off) or enable (on) the IP routing functionality.</td>
</tr>
<tr>
<td>firewalling = off</td>
<td>Enable (on) or disable (off) IP firewalling (master switch).</td>
</tr>
<tr>
<td>redirects = off</td>
<td>Disable (off) or enable (on) the sending of ICMP redirect messages.</td>
</tr>
<tr>
<td>sourcerouting = off</td>
<td>Disallow (off) or allow (on) IP source routed packets.</td>
</tr>
<tr>
<td>netbroadcasts = off</td>
<td>Disallow (off) or allow (on) net directed broadcasts.</td>
</tr>
<tr>
<td>ttl = 0 – 255</td>
<td>A number between 0 and 255. Specifies the default time-to-live (ttl).</td>
</tr>
<tr>
<td>fraglimit = 1 – 1024</td>
<td>A number between 1 and 1024. Represents the maximum number of IP packet fragments waiting for completion.</td>
</tr>
</tbody>
</table>

[forwarding] Disable (off) or enable (on) the IP routing functionality. OPTIONAL

[firewalling] Enable (on) or disable (off) IP firewalling (master switch). If applicable the CLI firewall level allows configuration of the SpeedTouch™ Pro with Firewall firewall. For security reasons this parameter is enabled per default. It is strongly recommended never to disable the SpeedTouch™ Pro with Firewall firewall.

[redirects] Disable (off) or enable (on) the sending of ICMP redirect messages. A router can send a redirect message in case a shorter path than the path followed is discovered. For security reasons this parameter is disabled per default.

[sourcerouting] Disallow (off) or allow (on) IP source routed packets. IP source routed packets are packets with the route to follow specified in the header. For security reasons this parameter is disabled per default.

[netbroadcasts] Disallow (off) or allow (on) net directed broadcasts. This parameter is per default disabled. In case netbroadcasts are allowed no traces of netbroadcasts are generated.

[ttl] A number between 0 and 255. Represents the default time-to-live (ttl) for locally generated IP packets. This parameter determines the number of hop-counts the IP packet may pass before it is dropped. Generally the time-to-live is 64 hop-counts. By limiting the time-to-live continuous circulation of IP packets on the network without ever reaching a destination is avoided.

[fraglimit] A number between 1 and 1024. Represents the maximum number of IP packet fragments waiting for completion. Generally the fragmentation limit is 64. By limiting the fragmentation limit the depletion of the buffer is avoided.
[defragmode] Define which packets are reassembled under which circumstances. 
Choose between:
- **normal**
  - Packets to be forwarded will not be reassembled.
  - Packets with local destination, i.e., destined for the **SpeedTouch™ Pro with Firewall**, are reassembled.
- **always**
  - Packets are always reassembled.
- **nat**
  - Same behaviour as **normal** except for packets to be forwarded through the NAT engine.
  - Packets on which address translation is performed are reassembled as the NAT engine requires the entire packet.

[addrcheck] Set the level of IP address checks. 
Choose between:
- **off**
  - No address checking is performed.
  - For advanced users only; in normal circumstances there should always be some kind of address checking.
- **own**
  - Minimum level of checking.
  - Only the address configuration on the **SpeedTouch™ Pro with Firewall** is checked.
- **static**
  - Checking of the address configuration of the **SpeedTouch™ Pro with Firewall** and also of traffic: addresses of incoming packets; this checking is related to constants (e.g., an address may not be entirely composed of one’s or zero’s).
- **dynamic**
  - Besides the address configuration of the **SpeedTouch™ Pro with Firewall** itself, and besides the checking of traffic on a constants level, additional checking is performed on the IP addresses that are determined by the configuration, more specifically by the network.

[mssclamping] Disable (off) or enable (on) mss clamping for low mtu interfaces. 
Mss clamping assures that the size of a TCP packet never exceeds the available mtu of the outgoing interface.
It is recommended not to disable this parameter.
EXAMPLE:

```bash
=> ip config
Forwarding on
Firewalling off
Sendredirects off
Sourcerouting on
NetBroadcasts off
Default TTL 128
Fraglimit 32 fragments
Fragcount currently 0 fragments
Defragment mode: always
Address checks: static
Mss clamping: on
=> ip config firewalling=on ttl=64 fraglimit=64 defragmode=nat

=> ip config
Forwarding on
Firewalling on
Sendredirects off
Sourcerouting on
NetBroadcasts off
Default TTL 64
Fraglimit 64 fragments
Fragcount currently 0 fragments
Defragment mode: nat
Address checks: static
Mss clamping: on
=>
```

RELATED COMMANDS:

- `ip ifconfig` Configure interface parameters.
**ip flush**

Flush complete IP configuration. Dynamic configurations (e.g. from PPP or CIP links) remain. The flush command does not impact previously saved configurations.

As an ip flush causes all local IP connectivity to be deleted, do not execute this command during an IP based local connection, e.g. a Telnet CLI session, or web based CLI access.

**SYNTAX:**

```
ip flush
```

**EXAMPLE:**

```
==ip aplist
3 cipl Type:ATM
 inet addr:172.16.0.5 Bcast:172.16.0.255 Mask:255.255.255.0
 UP RUNNING pat MTU:9180 ReasmMAX:65535 Group:0
 IPRX bytes:0 unicastpkts:0 bcastpkts:0
 IPTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
 HWRX bytes:0 unicastpkts:0 bcastpkts:0
 HWTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
2 eth0 Type:EthernetHWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:ff
 inet addr:10.0.0.2 Bcast:10.0.0.255 Mask:255.255.255.0
 UP RUNNING pat MTU:1500 ReasmMAX:65535 Group:2
 IPRX bytes:0 unicastpkts:0 bcastpkts:0
 IPTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
 HWRX bytes:0 unicastpkts:0 bcastpkts:0
 HWTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
0 loop Type:0
 inet addr:127.0.0.1 Bcast:127.255.255.255 Mask:255.0.0.0
 UP RUNNING MTU:1500 ReasmMAX:65535 Group:1
 IPRX bytes:116 unicastpkts:0 bcastpkts:2
 IPTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
 HWRX bytes:0 unicastpkts:0 bcastpkts:0
 HWTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
==ip flush
==ip aplist
3 cipl Type:ATM
 inet addr:172.16.0.5 Bcast:172.16.0.255 Mask:255.255.255.0
 UP RUNNING pat MTU:9180 ReasmMAX:65535 Group:0
 IPRX bytes:0 unicastpkts:0 bcastpkts:0
 IPTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
 HWRX bytes:0 unicastpkts:0 bcastpkts:0
 HWTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
0 loop Type:0
 inet addr:127.0.0.1 Bcast:127.255.255.255 Mask:255.0.0.0
 UP RUNNING MTU:1500 ReasmMAX:65535 Group:1
 IPRX bytes:116 unicastpkts:0 bcastpkts:2
 IPTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
 HWRX bytes:0 unicastpkts:0 bcastpkts:0
 HWTX bytes:0 unicastpkts:0 bcastpkts:0 droppkts:0
==>

**RELATED COMMANDS:**

- **ip load** Load saved or default IP configuration.
- **ip save** Save current IP configuration.
**ip ifconfig**

Configure interface parameters.

**SYNTAX:**

```
ip ifconfig
  intf = <interface name>
  [mtu = <number{293–20000}>]
  [status = {down|up}]
  [hwaddr = <hardware-address>]
  [group = {<wan|local|lan}>]
```

- **intf** The IP interface name. REQUIRED
- **[mtu]** A number between 293 and 20000. Represents the maximum transmission unit, i.e. the maximum packet size (including IP header) to use on this interface. The default value depends on the connection and packet service for which the interface was created. OPTIONAL
- **[status]** The administrative status of the interface. Choose between:
  - down
  - up
  OPTIONAL
- **[hwaddr]** The hardware address (e.g. the Ethernet MAC address) of this interface. OPTIONAL
- **[group]** The group this interface belongs to (e.g. for oriented firewalling). OPTIONAL

**EXAMPLE:**

```
=> ip iflist
Interface  GRP MTU RX  TX  TX–DROP STATUS  HWADDR
0  loop 1 1500 116 0 0 0  UP
1  eth0 2 3000 21045795 1019664 0 0 0  UP 00:80:9f:24:ab:cf
2  NewMer 0 1500 0 0 0 0 0  UP 00:80:9f:24:ab:cf
5  cip0 0 9180 0 0 0 0 0  UP

=> ip ifconfig intf=eth0 mtu=1500
=> ip iflist
Interface  GRP MTU RX  TX  TX–DROP STATUS  HWADDR
0  loop 1 1500 116 0 0 0  UP
1  eth0 2 1500 21054963 1025417 0 0 0  UP 00:80:9f:24:ab:cf
2  NewMer 0 1500 0 0 0 0 0  UP 00:80:9f:24:ab:cf
5  cip0 0 9180 0 0 0 0 0  UP
=>
```

**RELATED COMMANDS:**

- **ip config** Show/set global IP stack configuration options.
**ip iflist**

Show all current interfaces.

**SYNTAX:**

```
ip iflist
```

**EXAMPLE OUTPUT:**

```
=>ip iflist
Interface   GRP MTU RX TX   TX-DROP STATUS    HWADDR
0 loop      1 1500 116 0    0           UP   00:80:9f:24:ab:cf
1 eth0      2 3000 21045795 1019664 0 0  UP   00:80:9f:24:ab:cf
2 NewMer    0 1500 0 0 0  UP 00:80:9f:24:ab:cf
5 cip0      0 9180 0 0 0  UP
=>
```

**RELATED COMMANDS:**

`ip ifconfig` Configure interface parameters.
**ip load**

Load saved (or default) IP configuration.
Execute **ip flush** prior to **ip load**.

**SYNTAX:**

```
ip load [defaults = <yes|no>]
```

**EXAMPLE:**

```
=> ip rtlist
+-----------------+-----------------+-----------------+-------+
| Destination     | Source          | Gateway         | Mtrc  |
| 192.16.11.0/24  | 192.16.11.0/24  | 192.16.11.140eth0 | 0     |
| 172.16.1.1/32   | 0.0.0.0/0       | 172.16.1.1      | cip0  0|
| 127.0.0.1/32    | 0.0.0.0/0       | 127.0.0.1       | loop  0|
| 172.16.10.24    | 0.0.0.0/0       | 172.16.1.1      | cip0  0|
| 192.16.11.0/24  | 0.0.0.0/0       | 192.16.11.140eth0 | 0     |

=> ip save
=> ip rtlist
+-----------------+-----------------+-----------------+-------+
| Destination     | Source          | Gateway         | Mtrc  |
| 172.16.1.1/32   | 0.0.0.0/0       | 172.16.1.1      | cip0  0|
| 127.0.0.1/32    | 0.0.0.0/0       | 127.0.0.1       | loop  0|
| 172.16.10.24    | 0.0.0.0/0       | 172.16.1.1      | cip0  0|

=> ip load defaults=yes
=> ip rtlist
+-----------------+-----------------+-----------------+-------+
| Destination     | Source          | Gateway         | Mtrc  |
| 10.0.0.0/8      | 10.0.0.0/8      | 10.0.0.138 eth0 | 0     |
| 255.255.255.255/32 | 0.0.0.0/0       | 10.0.0.138 eth0 | 0     |
| 10.0.0.138/32   | 0.0.0.0/0       | 10.0.0.138 eth0 | 0     |
| 127.0.0.1/32    | 0.0.0.0/0       | 127.0.0.1       | loop  0|
| 172.16.1.1/24   | 0.0.0.0/0       | 172.16.1.1      | cip0  0|
| 10.0.0.0/8      | 0.0.0.0/0       | 10.0.0.138 eth0 | 0     |
| 0.0.0.0/0       | 0.0.0.0/0       | 10.0.0.138 eth0 | 0     |

=> ip load
=> ip rtlist
+-----------------+-----------------+-----------------+-------+
| Destination     | Source          | Gateway         | Mtrc  |
| 192.16.11.0/24  | 192.16.11.0/24  | 192.16.11.140eth0 | 0     |
| 172.16.1.1/32   | 0.0.0.0/0       | 172.16.1.1      | cip0  0|
| 192.16.11.140/32 | 0.0.0.0/0       | 192.16.11.140eth0 | 0     |
| 127.0.0.1/32    | 0.0.0.0/0       | 127.0.0.1       | loop  0|
| 172.16.1.1/24   | 0.0.0.0/0       | 172.16.1.1      | cip0  0|
| 192.16.11.0/24  | 0.0.0.0/0       | 192.16.11.140eth0 | 0     |

=>
```

**RELATED COMMANDS:**

- **ip flush**  
  Flush complete IP configuration.
- **ip save**  
  Save current IP configuration.
**ip ping**
Send ICMP ECHO_REQUEST packets.

**SYNTAX:**

```
ip ping  
   addr = <ip-address>  
   [count = <number{1–1000000}>]  
   [size = <number{1–20000}>]  
   [interval = <number{100–1000000}>]  
   [listen = <(off|on)>]
```

- **addr**  
The destination IP address.  
REQUIRED

- **[count]**  
A number between 1 and 1000000.  
Represents the number of pings to send.  
OPTIONAL

- **[size]**  
A number between 1 and 20000 (bytes).  
Represents the size of the ping packet(s).  
OPTIONAL

- **[interval]**  
A number between 100 and 1000000 (milliseconds).  
Represents the intermediate interval between two sent ICMP packets.  
OPTIONAL

- **[listen]**  
Listen for incoming ICMP packets (on) or only send ICMP packets (off).  
OPTIONAL

**EXAMPLE:**

```
=> ip ping addr=10.0.0.148 listen=off
=> ip ping addr=10.0.0.148 listen=on
9 bytes from 10.0.0.148: Echo Request
=> ip ping addr=10.0.0.148 count=15 listen=on
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
9 bytes from 10.0.0.148: Echo Request
```
**ip rtadd**

Add a route to the SpeedTouch™ Pro with Firewall routing table.

**SYNTAX:**

```
ip rtadd dst = <ip-address>
   [dstmsk = <ip-mask(dotted or cidr)>]
   [src = <ip-address>]
   [srccmsk = <ip-mask(dotted or cidr)>]
   [gateway = <ip-address>]
   [intf = <interface name>]
   [metric = <number(0–100)>]
   [type = <number>]
```

- `dst`: The destination IP address(es) for this route. Supports cidr notation. **REQUIRED**
- `[dstmsk]`: The destination IP address mask. **OPTIONAL**
- `[src]`: The source IP address(es) allowed to use this route. Supports cidr notation. **OPTIONAL**
- `[srccmsk]`: The source IP address mask. **OPTIONAL**
- `[gateway]`: The IP address of the next hop. Must be directly connected. The parameters ‘gateway’ and ‘intf’ are mutually exclusive. **OPTIONAL**
- `[intf]`: Only for special interface routes: the outgoing IP interface name. The parameters ‘gateway’ and ‘intf’ are mutually exclusive. **OPTIONAL**
- `[metric]`: The metric for this route (currently not used). **OPTIONAL**
- `[type]`: Route classification. For internal use only. **OPTIONAL**

**EXAMPLE:**

```
=> ip rtlist
   Destination  Source  Gateway  Intf  Mtrc
   10.0.0.0/24  10.0.0.0/24  10.0.0.140 eth0  0
   10.0.0.140/32  0.0.0.0/0  10.0.0.140 eth0  0
   127.0.0.1/32  0.0.0.0/0  127.0.0.1 loop  0
=> ip rtadd dst=10.10.0.0/24 src=10.0.0.0/24 gateway=10.0.0.140
=> ip rtlist
   Destination  Source  Gateway  Intf  Mtrc
   10.0.0.0/24  10.0.0.0/24  10.0.0.140 eth0  0
   10.10.0.0/24  10.0.0.0/24  10.0.0.140 eth0  0
   10.0.0.140/32  0.0.0.0/0  10.0.0.140 eth0  0
   127.0.0.1/32  0.0.0.0/0  127.0.0.1 loop  0
=>
```

**RELATED COMMANDS:**

- **ip rtdelete**: Remove a route from the routing table.
- **ip rtlist**: Show current routing table.
**ip rtdelete**
Delete a route from the SpeedTouch™ Pro with Firewall routing table.

**SYNTAX:**

```
ip rtdelete
   dst = <ip-address>
   [dstmsk = <ip-mask(dotted or cidr)>]
   [src = <ip-address>]
   [srcmsk = <ip-mask(dotted or cidr)>]
   [gateway = <ip-address>]
   [intf = <interface name>]
```

- **dst**: The destination IP address(es) of the route. Supports cidr notation. REQUIRED
- **[dstmsk]**: The destination IP address mask. OPTIONAL
- **[src]**: The source IP address(es) of the route. Supports cidr notation. OPTIONAL
- **[srcmsk]**: The source IP address mask. OPTIONAL
- **[gateway]**: The IP address of the next hop. Must be directly connected. The parameters ‘gateway’ and ‘intf’ are mutually exclusive. OPTIONAL
- **[intf]**: Only for special interface routes : the outgoing IP interface name. The parameters ‘gateway’ and ‘intf’ are mutually exclusive. OPTIONAL

**EXAMPLE:**

```
=> ip rtlist
    Destination          Source        Gateway      Intf  Mtrc
    10.0.0.0/24           10.0.0.0/24   10.0.0.140  eth0  0
    10.10.0.0/24          10.0.0.0/24   10.0.0.140  eth0  0
    10.0.0.140/32         0.0.0.0/0     10.0.0.140  eth0  0
    127.0.0.1/32          0.0.0.0/0     127.0.0.1   loop  0
=> ip rtdelete dst=10.0.0.0/24 src=10.0.0.0/24 gateway=10.0.0.140
=> ip rtlist
    Destination          Source        Gateway      Intf  Mtrc
    10.0.0.0/24           10.0.0.0/24   10.0.0.140  eth0  0
    127.0.0.1/32          0.0.0.0/0     127.0.0.1   loop  0
=>
```

**RELATED COMMANDS:**

- **ip rtadd**: Add a route to the routing table.
- **ip rtlist**: Show current routing table.
**ip rtlist**
Show current SpeedTouch™ Pro with Firewall routing table.

**SYNTAX:**
```
ip rtlist
```

**EXAMPLE OUTPUT:**
```
-> ip rtlist
  Destination    Source     Gateway       Intf  Mtrc
  10.0.0.0/24    10.0.0.0/24 10.0.0.140   eth0  0
  172.16.0.5/32  0.0.0.0/0   172.16.0.5  cip1  0
  0.0.0.140/32   0.0.0.0/0   10.0.0.140   eth0  0
  127.0.0.1/32   0.0.0.0/0   127.0.0.1    loop  0
  10.0.0.0/24    0.0.0.0/0   10.0.0.140   eth0  0
  172.16.0.0/24  0.0.0.0/0   172.16.0.5   cip1  1
->
```

**RELATED COMMANDS:**
- **ip rtadd** Add a route to the routing table.
- **ip rtdelete** Remove a route from the routing table.
**ip save**

Save current IP configuration.

**SYNTAX:**

```
ip save
```

**EXAMPLE:**

```
=>ip rttlist
  Destination   Source     Gateway     Intf  Mtrc
  192.16.11.0/24 192.16.11.0/24 192.16.11.140eth0 0
  172.16.1.1/32  0.0.0.0/0   172.16.1.1  cip0  0
  192.16.11.140/32 0.0.0.0/0  192.16.11.140eth0 0
  127.0.0.1/32   0.0.0.0/0   127.0.0.1  loop  0
  172.16.1.1/24  0.0.0.0/0   172.16.1.1  cip0  0
  192.16.11.0/24 0.0.0.0/0   192.16.11.140eth0 0
=>ip save
=>ip flush
=>ip rttlist
  Destination   Source     Gateway     Intf  Mtrc
  172.16.1.1/32  0.0.0.0/0   172.16.1.1  cip0  0
  127.0.0.1/32   0.0.0.0/0   127.0.0.1  loop  0
  172.16.1.0/24  0.0.0.0/0   172.16.1.1  cip0  0
=>ip load
=>ip rttlist
  Destination   Source     Gateway     Intf  Mtrc
  192.16.11.0/24 192.16.11.0/24 192.16.11.140eth0 0
  172.16.1.1/32  0.0.0.0/0   172.16.1.1  cip0  0
  192.16.11.140/32 0.0.0.0/0  192.16.11.140eth0 0
  127.0.0.1/32   0.0.0.0/0   127.0.0.1  loop  0
  172.16.1.1/24  0.0.0.0/0   172.16.1.1  cip0  0
  192.16.11.0/24 0.0.0.0/0   192.16.11.140eth0 0
=>
```

**RELATED COMMANDS:**

- `ip flush`  
  Flush complete IP configuration.

- `ip load`  
  Load saved or default IP configuration.
**ip sendto**

Send UDP packets.

**SYNTAX:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip sendto</td>
<td></td>
<td><code>addr = &lt;ip-address&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>[count = &lt;number{1–100000}&gt;]</code></td>
<td>A number between 1 and 1000000. Represents the number of UDP packets to send.</td>
</tr>
<tr>
<td></td>
<td><code>[size = &lt;number{1–20000}&gt;]</code></td>
<td>A number between 1 and 20000 (bytes). Represents the size of the ping packet(s).</td>
</tr>
<tr>
<td></td>
<td><code>[interval = &lt;number{100–1000000}&gt;]</code></td>
<td>A number between 100 and 1000000 (milliseconds). Represents the intermediate interval between two sent UDP packets.</td>
</tr>
<tr>
<td></td>
<td>`[listen = {off</td>
<td>on}&gt;]`</td>
</tr>
<tr>
<td></td>
<td><code>[srcport = &lt;number{1–65535}&gt;]</code></td>
<td>The UDP source port number to use.</td>
</tr>
<tr>
<td></td>
<td><code>[dstport = &lt;number{1–65535}&gt;]</code></td>
<td>The UDP destination port number to send to.</td>
</tr>
</tbody>
</table>

**addr**

The destination IP address. **REQUIRED**

**[count]**

A number between 1 and 1000000. Represents the number of UDP packets to send. **OPTIONAL**

**[size]**

A number between 1 and 20000 (bytes). Represents the size of the ping packet(s). **OPTIONAL**

**[interval]**

A number between 100 and 1000000 (milliseconds). Represents the intermediate interval between two sent UDP packets. **OPTIONAL**

**[listen]**

Listen for incoming UDP packets (on) or only send UDP packets (off). **OPTIONAL**

**[srcport]**

The UDP source port number to use. **OPTIONAL**

**[dstport]**

The UDP destination port number to send to. **REQUIRED**

**EXAMPLE:**

```plaintext
=> ip sendto addr=10.0.0.148 listen=on srcport=19 dstport=1025
1 bytes from 10.0.0.148:1025
41                                              A

=> ip sendto addr=10.0.0.148 count=3 listen=on srcport=19 dstport=1025
1 bytes from 10.0.0.148:1025
41                                              A
1 bytes from 10.0.0.148:1025
41                                              A
1 bytes from 10.0.0.148:1025
41                                              A
=>
```

**RELATED COMMANDS:**

**ip ping**

Send ICMP ECHO_REQUEST packets.
10 MER Commands

mer (to access the MER level)
mer flush
mer ifadd
mer ifattach
mer ifconfig
mer ifdelete
mer ifdetach
mer iflist
mer load
mer save
**mer flush**
Flush complete MER configuration.
The flush command does not impact previously saved configurations.

**SYNTAX:**

```
mer flush
```

**EXAMPLE:**

```
=>mer iflist
NewMer  :  dest : Br3
    Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
    Connection State : connected
    RX bytes : 0  frames : 0
    TX bytes : 0  frames : 0  dropframes : 0
=>mer flush
=>mer iflist
=>
```

**RELATED COMMANDS:**

- **mer load**  
  Load saved or default MER configuration.
- **mer save**  
  Save current MER configuration.
**mer ifadd**

Create a new MER interface.

**SYNTAX:**

```
mer ifadd [intf = <string>]
[dest = <phonebook entry>]
```

- **[intf]**
  - The name for the new MER interface.
  - If not specified, the destination parameter must be specified. In this case the name of the destination will double as interface name.
  - **OPTIONAL**

- **[dest]**
  - The destination for the new MER interface.
  - Typically, an phonebook entry.
  - **OPTIONAL**

**EXAMPLE:**

```
=>mer iflist
NewMer : dest : Br3
  Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
  Connection State : connected
  RX bytes: 0   frames: 0
  TX bytes: 0   frames: 0   dropframes: 0

=>phonebook list
Name   Type    Use  Address
Br1    bridge 1  8.35
Br2    bridge 1  8.36
CIPPVC3 cip    1  8.82
CIPPVC4 cip    1  8.83

=>mer ifadd intf=MoreMer dest=Br4
=>mer iflist
NewMer : dest : Br3
  Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
  Connection State : connected
  RX bytes: 0   frames: 0
  TX bytes: 0   frames: 0   dropframes: 0
MoreMer : dest : Br4
  Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
  Connection State : not-connected
=>
```

**RELATED COMMANDS:**

- **mer ifattach**
  - Attach a MER interface.
- **mer ifconfig**
  - Configure a MER interface.
- **mer ifdelete**
  - Delete a MER interface.
- **mer ifdetach**
  - Detach a MER interface.
- **mer iflist**
  - Show current MER interfaces.
mer ifattach
Attach (i.e. connect) a MER interface.

SYNTAX:

```
mer ifattach    intf = <ifname>
```

- `intf` The name of the MER interface to attach.

EXAMPLE:

```
=>mer iflist
NewMer        : dest : Br3
  Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
  Connection State : connected
  RX bytes: 0     frames: 0
  TX bytes: 0     frames: 0     dropframes: 0
MoreMer       : dest : Br4
  Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
  Connection State : not-connected
=>mer ifattach intf=MoreMer
=>mer iflist
NewMer        : dest : Br3
  Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
  Connection State : connected
  RX bytes: 0     frames: 0
  TX bytes: 0     frames: 0     dropframes: 0
MoreMer       : dest : Br4
  Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
  Connection State : connected
  RX bytes: 0     frames: 0
  TX bytes: 0     frames: 0     dropframes: 0
=>
```

RELATED COMMANDS:

- **mer ifadd** Create a new MER interface.
- **mer ifconfig** Configure a MER interface.
- **mer ifdelete** Delete a MER interface.
- **mer ifdetach** Detach a MER interface.
- **mer iflist** Show current MER interfaces.
**mer ifconfig**

Configure a MER interface.

**SYNTAX:**

```
mer ifconfig   intf = <ifname>
               [dest = <ifname>]
               [qos = <string>]
               [encaps = <(llc/snap|vcmux)>]
               [retry = <number {0-65535}>]
```

- `intf`: The name of the MER interface to configure. **REQUIRED**
- `dest`: The destination for this interface. Typically a phonebook entry. This parameter needs only to be specified in case of an interface created without specified destination. **OPTIONAL**
- `qos`: The name of a configured Quality Of Service book entry. This parameter never needs to be specified. **OPTIONAL**
- `encaps`: The type of encapsulation to be used for this bridge interface. Choose between:
  - llc/snap
  - vcmux  **OPTIONAL**
- `retry`: A number between 0 and 65535. Represents the number of WAN connection setup retries before giving up. By default the retry value is 10. **OPTIONAL**

**EXAMPLE:**

```
=> mer iflist
MoreMer : dest : Br4
          Retry : 10  QoS : default  Encaps : vcmux  Fcs : off
          Connection State : connected
          RX bytes: 0    frames: 0
          TX bytes: 0    frames: 0    dropframes: 0

=> mer ifconfig intf=MoreMer encaps=llc/snap retry=15
=> mer iflist
MoreMer : dest : Br4
          Retry : 15  QoS : default  Encaps : llc/snap  Fcs : off
          Connection State : connected
          RX bytes: 0    frames: 0
          TX bytes: 0    frames: 0    dropframes: 0
```

**RELATED COMMANDS:**

- `mer ifadd`: Create a new MER interface.
- `mer ifattach`: Attach a MER interface.
- `mer ifdelete`: Delete a MER interface.
- `mer ifdetach`: Detach a MER interface.
- `mer iflist`: Show current MER interfaces.
mer ifdelete
Delete a MER interface.

SYNTAX:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mer ifdelete</code></td>
<td>The name of the MER interface. REQUIRED</td>
</tr>
</tbody>
</table>

EXAMPLE:

```plaintext
=> mer iflist
NewMer : dest : Br3
    Retry : 10 QoS : default Encaps : llc/snap Fcs : off
    Connection State : connected
    RX bytes: 0 frames: 0
    TX bytes: 0 frames: 0 dropframes: 0
MoreMer: dest : Br4
    Retry : 10 QoS : default Encaps : llc/snap Fcs : off
    Connection State : not-connected
=> mer ifdelete intf=MoreMer
=> mer iflist
NewMer : dest : Br3
    Retry : 10 QoS : default Encaps : llc/snap Fcs : off
    Connection State : connected
    RX bytes: 0 frames: 0
    TX bytes: 0 frames: 0 dropframes: 0
=>
```

RELATED COMMANDS:

- **mer ifadd**: Create a new MER interface.
- **mer ifattach**: Attach a MER interface.
- **mer ifconfig**: Configure a MER interface.
- **mer ifdetach**: Detach a MER interface.
- **mer iflist**: Show current MER interfaces.
**mer ifdetach**

Detach a MER interface.

**SYNTAX:**

```
mer ifdetach  intf = <ifname>
```

- `intf` The name of the MER interface. REQUIRED

**EXAMPLE:**

```plaintext
=>mer iflist
NewMer    :  dest : Br3
    Retry : 10    QoS : default    Encaps : llc/snap    Fcs : off
    Connection State : connected
    RX bytes: 0    frames: 0
    TX bytes: 0    frames: 0    dropframes: 0
MoreMer   :  dest : Br4
    Retry : 10    QoS : default    Encaps : llc/snap    Fcs : off
    Connection State : connected
    RX bytes: 0    frames: 0
    TX bytes: 0    frames: 0    dropframes: 0

=>mer ifdetach intf=MoreMer
=>mer iflist
NewMer    :  dest : Br3
    Retry : 10    QoS : default    Encaps : llc/snap    Fcs : off
    Connection State : connected
    RX bytes: 0    frames: 0
    TX bytes: 0    frames: 0    dropframes: 0
MoreMer   :  dest : Br4
    Retry : 10    QoS : default    Encaps : llc/snap    Fcs : off
    Connection State : not-connected
```

**RELATED COMMANDS:**

- **mer ifadd** Create a new MER interface.
- **mer ifattach** Attach a MER interface.
- **mer ifconfig** Configure a MER interface.
- **mer ifdelete** Delete a MER interface.
- **mer iflist** Show current MER interfaces.
mer iflist
Show all or a specified MER interface(s).

SYNTAX:
\[
\text{mer iflist} \quad [\text{intf} = \text{ifname}] \\
\]

| [intf] | The name of the MER interface. If not specified all MER interfaces are listed. | OPTIONAL |

EXAMPLE OUTPUT:

```
=> mer iflist
NewMer : dest : Br3
  Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
  Connection State : connected
  RX bytes: 0  frames: 0
  TX bytes: 0  frames: 0  dropframes: 0
MoreMer : dest : Br4
  Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
  Connection State : connected
  RX bytes: 0  frames: 0
  TX bytes: 0  frames: 0  dropframes: 0
=>
```

RELATED COMMANDS:
- **mer ifadd**: Create a new MER interface.
- **mer ifattach**: Attach a MER interface.
- **mer ifconfig**: Configure a MER interface.
- **mer ifdelete**: Delete a MER interface.
- **mer detach**: Detach a MER interface.
**mer load**

Load saved (or default) MER configuration.
Execute `mer flush` prior to `mer load`.

SYNTAX:

```
mer load [defaults <yes|no>]
```

- `[defaults]` Load factory defaults (yes) or saved configuration (no). **OPTIONAL**
- Not specifying this parameter loads the saved configuration.

EXAMPLE:

```
=> mer iflist
NewMer : dest : Br3
    Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
    Connection State : connected
    RX bytes: 0  frames: 0
    TX bytes: 0  frames: 0  dropframes: 0
MoreMer : dest : Br4
    Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
    Connection State : not-connected
=> mer save
=> mer flush
=> mer iflist
=> mer load
=> mer iflist
NewMer : dest : Br3
    Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
    Connection State : connected
    RX bytes: 0  frames: 0
    TX bytes: 0  frames: 0  dropframes: 0
MoreMer : dest : Br4
    Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
    Connection State : not-connected
=>
```

RELATED COMMANDS:

- **mer flush**  
  Flush complete MER configuration.
- **mer save**  
  Save current MER configuration.
mer save
Save current MER configuration.

SYNTAX:

mer save

EXAMPLE:

```bash
=>mer iflist
NewMer : dest : Br3
        Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
        Connection State : connected
        RX bytes: 0       frames: 0
        TX bytes: 0       frames: 0       dropframes: 0
MoreMer : dest : Br4
        Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
        Connection State : not-connected
```

```bash
=>mer save
=>mer flush
=>mer iflist
=>mer load
=>mer iflist
NewMer : dest : Br3
        Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
        Connection State : connected
        RX bytes: 0       frames: 0
        TX bytes: 0       frames: 0       dropframes: 0
MoreMer : dest : Br4
        Retry : 10  QoS : default  Encaps : llc/snap  Fcs : off
        Connection State : not-connected
=>>
```

RELATED COMMANDS:

mer flush  Flush complete MER configuration.
mer load   Load saved or default MER configuration.
11 NAT Commands

nat (to access the NAT level)
nat applist
nat bind
nat bindlist
nat create
nat defserver
nat delete
nat disable
nat enable
nat flush
nat list
nat load
nat save
nat unbind
nat applist
List available NAPT protocol helpers.
Certain protocols are ‘sensitive’ to NAPT in that they do not function properly when dealing with it. This list shows which ‘NAPT-sensitive’ applications are supported on the SpeedTouch™ Pro with Firewall, i.e. the inherent knowledge of the SpeedTouch™ Pro with Firewall on this matter.

SYNTAX:

nat applist

EXAMPLE OUTPUT:

=>nat applist
Application  Proto  DefaultPort
H254         tcp    0
H323         tcp    1720
RAUDIO(PNA)  tcp    7070
RTSP         tcp    554
IRC          tcp    6667
FTP          tcp    21
=>

RELATED COMMANDS:

nat bind  Create a new helper/port binding.
nat bindlist  List current NAPT helper/port bindings.
nat unbind  Delete an existing helper/port binding.
**nat bind**
Create a new helper/port binding.

**SYNTAX:**

```
nat bind application = <string> port = <TCP/UDP service name or port number>
```

- **application**: The name of a NAPT application helper.
  The name must be spelled exactly as listed in the application list (nat applist).
  REQUIRED

- **port**: The port number this application handler should work on.
  REQUIRED

**EXAMPLE INPUT:**

```
->nat applist
Application Proto DefaultPort
H254 tcp 0
H323 tcp 1720
RAUDIO(PNA) tcp 7070
RTSP tcp 554
IRC tcp 6667
FTP tcp 21
->nat bindlist
Application Proto Port
H323 tcp 1720
FTP tcp 21
RTSP tcp 554
IRC tcp 6667
RAUDIO(PNA) tcp 7070
->nat bind application=RAUDIO(PNA) port=7071
->nat bindlist
Application Proto Port
RAUDIO(PNA) tcp 7071
H323 tcp 1720
FTP tcp 21
RTSP tcp 554
IRC tcp 6667
RAUDIO(PNA) tcp 7070
->
```

**RELATED COMMANDS:**

- **nat applist**: List available NAPT protocol helpers.
- **nat bindlist**: List current NAPT helper/port bindings.
- **nat unbind**: Delete an existing helper/port binding.
**nat bindlist**

List current NAPT helper/port bindings.

**SYNTAX:**

```
nat bindlist
```

**EXAMPLE OUTPUT:**

```
=>nat bindlist
Application  Proto  Port
RAUDIO(PNA)  tcp    7071
H323         tcp    1720
FTP          tcp    21
RTSP         tcp    554
IRC          tcp    6667
RAUDIO(PNA)  tcp    7070
=>
```

**RELATED COMMANDS:**

- `nat applist` List available NAPT protocol helpers.
- `nat bind` Create a new NAPT helper/port binding.
- `nat unbind` Delete an existing helper/port binding.
nat create
Create a static NAPT entry. Typically used to install specific servers behind the SpeedTouch™ Pro with Firewall’s NAPT device.

SYNTAX:

```
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>nat create</td>
<td>protocol = &lt;IP protocol name or number&gt;</td>
<td></td>
</tr>
<tr>
<td>inside_addr = &lt;ip-address&gt;</td>
<td>The IP address of the local host (intended to receive the incoming traffic) behind the SpeedTouch™ Pro with Firewall’s NAPT device. Typically, a private IP address.</td>
<td>REQUIRED</td>
</tr>
<tr>
<td>[inside_port = &lt;TCP/UDP service name or port number&gt;]</td>
<td>The port number of the application on the local host. Applicable for TCP and UDP protocols. All other protocols do not need a port to be specified.</td>
<td>OPTIONAL</td>
</tr>
<tr>
<td>outside_addr = &lt;ip-address&gt;</td>
<td>The apparent host IP address this application is running on, i.e. the NAPT enabled WAN IP address of the SpeedTouch™ Pro with Firewall. Use ‘0’ to create a template. Such template will then be valid for any of SpeedTouch™ Pro with Firewall’s NAPT enabled IP addresses, e.g. also dynamically assigned/negotiated IP addresses.</td>
<td>REQUIRED</td>
</tr>
<tr>
<td>[outside_port = &lt;TCP/UDP service name or port number&gt;]</td>
<td>The apparent port number this application is running on. Applicable for TCP and UDP protocols. All other protocols do not need a port to be specified.</td>
<td>OPTIONAL</td>
</tr>
</tbody>
</table>
```
EXAMPLE:

```plaintext
=>nat list
=>ip apolist
1 eth0    Type:EthernetHWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:ff
inet addr:10.10.10.147 Bcast: 10.10.10.255 Mask: 255.0.0.0
UP RUNNING pat MTU:1500 ReasmMAX:65535 Group:2
IPRX bytes:19791886 unicastpkts:11341 brcastpkts:290555
IPTX bytes:839550 unicastpkts:11477 brcastpkts:0 droppkts:0
HWRX bytes:0 unicastpkts:0 brcastpkts:0
HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
5 cip0   Type:ATM
inet addr:172.16.0.5 Bcast:127.16.0.255 Mask:255.255.255.0
UP RUNNING MTU:1500 ReasmMAX:65535 Group:0
IPRX bytes:0 unicastpkts:0 brcastpkts:0
IPTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
HWRX bytes:0 unicastpkts:0 brcastpkts:0
HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
0 loop   Type:0
inet addr:127.0.0.1 Bcast:127.255.255.255 Mask:255.0.0.0
UP RUNNING MTU:1500 ReasmMAX:65535 Group:1
IPRX bytes:116 unicastpkts:0 brcastpkts:0
IPTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
HWRX bytes:0 unicastpkts:0 brcastpkts:0
HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
=>nat enable addr=172.16.0.5 type=pat
=>nat create protocol=tcp inside_addr=10.0.0.1 inside_port=80 outside_addr=172.16.0.5 outside_port=1080
=>nat list
1 6 10.0.0.138:80 172.16.0.5:1080 0.0.0.0:0 19 8 9
=>
```

RELATED COMMANDS:

- **nat delete**: Delete a static NAT entry.
- **nat disable**: Disable NAT on the specified IP address.
- **nat enable**: Enable NAT on one of the devices own IP addresses.
- **nat list**: List NAT connection database.
**nat defserver**

Define the default server behind the SpeedTouch™ Pro with Firewall NAPT device that receives all (unknown) incoming packets.

In typical LAN configurations one local ‘default’ server will be responsible for all WAN-LAN mail, http, ftp, ... connectivity. This command allows to specify this server. For specific services, use the **nat create** command.

**SYNTAX:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nat defserver</td>
<td>Define the default server to route incoming packets to.</td>
</tr>
<tr>
<td>[addr = &lt;ip-address&gt;]</td>
<td>The IP address of the server (on the ‘inside’) that will receive all (unknown) incoming packets. If not specified the current default server is shown.</td>
</tr>
</tbody>
</table>

**EXAMPLE INPUT/OUTPUT:**

```
=>nat defserver
Default server is undefined
=>nat defserver addr=10.0.0.1
=>nat defserver
Default server is 10.0.0.1
=>
```
nat delete
Delete a static NAPT entry.

SYNTAX:

```
nat delete protocol = <IP protocol name or number>
inside_addr = <ip–address>
[inside_port = <TCP/UDP service name or port number>]
outside_addr = <ip–address>
[outside_port = <TCP/UDP service name or port number>]
```

- `protocol`: The IP protocol name (or number) of the NAT entry. REQUIRED
- `inside_addr`: The IP address of the NAT entry. REQUIRED
- `[inside_port]`: The port number of the NAT entry. OPTIONAL
- `outside_addr`: The apparent host IP address of the NAT entry. REQUIRED
- `[outside_port]`: The apparent port number of the NAT entry. OPTIONAL

EXAMPLE:

```
=> nat list
1 6 10.0.0.138:80 172.16.0.5:1080 0 0 0 0 19 8 9
2 17 10.0.0.138:138 10.0.0.140:138 10.0.0.20:138 11 20 10
3 17 10.0.0.138:137 10.0.0.140:137 10.0.0.254:137 11 20 10
4 17 10.0.0.138:7938 10.0.0.140:7938 10.0.0.96:4756 11 20 10
5 17 10.0.0.138:513 10.0.0.140:513 10.0.0.109:513 11 20 10
6 17 10.0.0.138:111 10.0.0.140:111 10.0.0.96:4756 11 20 10
=> nat delete protocol=tcp inside_addr=10.0.0.138 inside_port=80 outside_addr=172.16.0.5
outside_port 1080
=> nat list
1 17 10.0.0.138:138 10.0.0.140:138 10.0.0.20:138 11 20 10
2 17 10.0.0.138:137 10.0.0.140:137 10.0.0.254:137 11 20 10
3 17 10.0.0.138:7938 10.0.0.140:7938 10.0.0.96:4756 11 20 10
4 17 10.0.0.138:513 10.0.0.140:513 10.0.0.109:513 11 20 10
5 17 10.0.0.138:111 10.0.0.140:111 10.0.0.96:4756 11 20 10
=>
```

RELATED COMMANDS:

- **nat create**: Create a static NAPT entry.
- **nat disable**: Disable NAPT on one of the **SpeedTouch™ Pro with Firewall** IP addresses.
- **nat enable**: Enable NAPT on one of the **SpeedTouch™ Pro with Firewall** IP addresses.
- **nat list**: List NAPT connection database.
nat disable
Disable NAPT on a SpeedTouch™ Pro with Firewall IP address.

SYNTAX:
```bash
nat disable addr = <ip-address>
```

<table>
<thead>
<tr>
<th>addr</th>
<th>One of SpeedTouch™ Pro with Firewall's IP addresses one which NAPT is enabled. REQUIRED</th>
</tr>
</thead>
</table>

EXAMPLE:
```
=>nat list
<table>
<thead>
<tr>
<th>Index</th>
<th>Prot</th>
<th>Inside-addr:Port</th>
<th>Outside-addr:Port</th>
<th>Foreign-addr:Port</th>
<th>Flgs</th>
<th>Exp</th>
<th>State</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>10.0.0.138:80</td>
<td>172.16.0.5:1080</td>
<td>0.0.0.0:0</td>
<td>19</td>
<td>8</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>17</td>
<td>10.0.0.138:138</td>
<td>10.0.0.140:138</td>
<td>10.0.0.20:138</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>17</td>
<td>10.0.0.138:137</td>
<td>10.0.0.140:137</td>
<td>10.0.0.254:137</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>17</td>
<td>10.0.0.138:7938</td>
<td>10.0.0.140:7938</td>
<td>10.0.0.96:4756</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>17</td>
<td>10.0.0.138:513</td>
<td>10.0.0.140:513</td>
<td>10.0.0.109:513</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>17</td>
<td>10.0.0.138:111</td>
<td>10.0.0.140:111</td>
<td>10.0.0.96:4756</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>
```
```
=>nat disable addr 172.16.0.5
```
```
=>nat list
<table>
<thead>
<tr>
<th>Index</th>
<th>Prot</th>
<th>Inside-addr:Port</th>
<th>Outside-addr:Port</th>
<th>Foreign-addr:Port</th>
<th>Flgs</th>
<th>Exp</th>
<th>State</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>17</td>
<td>10.0.0.138:138</td>
<td>10.0.0.140:138</td>
<td>10.0.0.20:138</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>17</td>
<td>10.0.0.138:137</td>
<td>10.0.0.140:137</td>
<td>10.0.0.254:137</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>17</td>
<td>10.0.0.138:7938</td>
<td>10.0.0.140:7938</td>
<td>10.0.0.96:4756</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>17</td>
<td>10.0.0.138:513</td>
<td>10.0.0.140:513</td>
<td>10.0.0.109:513</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>17</td>
<td>10.0.0.138:111</td>
<td>10.0.0.140:111</td>
<td>10.0.0.96:4756</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>
```
```
RELATED COMMANDS:
```
nat create
Create a static NAPT entry.
nat delete
Delete a static NAPT entry.
nat enable
Enable NAPT on one of the SpeedTouch™ Pro with Firewall IP addresses.
nat list
List NAPT connection database.
### nat enable

Enable NAPT on a SpeedTouch™ Pro with Firewall IP address.

**SYNTAX:**

```plaintext
nat enable addr = <ip-address>  
(type = <{none}|pat}>)
```

- **addr**
  - The SpeedTouch™ Pro with Firewall IP address on which NAPT must be applied. REQUIRED
- **[type]**
  - Enable port translation (pat) or not (none). OPTIONAL

**EXAMPLE:**

```bash
-> ip aplist
1  eth0 Type:EthernetHWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:ff
   inet addr:10.10.10.147 Bcast:10.10.10.255 Mask:255.0.0.0
      UP RUNNING MTU:1500 ReasmMAX:65535 Group:2
    IPRX bytes:19791886 unicastpkts:11341 brcastpkts:290555
    IPTX bytes:839550 unicastpkts:11477 brcastpkts:0 droppkts:0
    HWRX bytes:0 unicastpkts:0 brcastpkts:0
    HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
0  loop Type:0
   inet addr:127.0.0.1 Bcast:127.255.255.255 Mask:255.0.0.0
      UP RUNNING MTU:1500 ReasmMAX:65535 Group:1
    IPRX bytes:116 unicastpkts:0 brcastpkts:2
    IPTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
    HWRX bytes:0 unicastpkts:0 brcastpkts:0
    HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
-> nat enable addr=10.10.10.147 type=pat
-> ip aplist
1  eth0 Type:EthernetHWaddr 00:80:9f:24:ab:cf BRHWaddr ff:ff:ff:ff:ff:ff
   inet addr:10.10.10.147 Bcast:10.10.10.255 Mask:255.0.0.0
      UP RUNNING pat MTU:1500 ReasmMAX:65535 Group:2
    IPRX bytes:19791886 unicastpkts:11341 brcastpkts:290555
    IPTX bytes:839550 unicastpkts:11477 brcastpkts:0 droppkts:0
    HWRX bytes:0 unicastpkts:0 brcastpkts:0
    HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
0  loop Type:0
   inet addr:127.0.0.1 Bcast:127.255.255.255 Mask:255.0.0.0
      UP RUNNING MTU:1500 ReasmMAX:65535 Group:1
    IPRX bytes:116 unicastpkts:0 brcastpkts:2
    IPTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
    HWRX bytes:0 unicastpkts:0 brcastpkts:0
    HWTX bytes:0 unicastpkts:0 brcastpkts:0 droppkts:0
->
```

**RELATED COMMANDS:**

- nat create
  - Create a static NAPT entry.
- nat delete
  - Delete a static NAPT entry.
- nat disable
  - Disable NAPT on one of the SpeedTouch™ Pro with Firewall IP addresses.
- nat list
  - List NAPT connection database.
**nat flush**
Flush complete NAPT configuration.
The flush command does not impact previously saved configurations.

**SYNTAX:**

```
nat flush
```

**EXAMPLE:**

```
=>nat list
Indx Prot Inside-addr:PortOutside-addr:Port Foreign-addr:Port Flgs Expir State Control
1 17 10.0.0.138:138 10.0.0.140:138 10.0.0.20:138 11 20 10
2 17 10.0.0.138:137 10.0.0.140:137 10.0.0.254:137 11 20 10
3 17 10.0.0.138:7938 10.0.0.140:7938 10.0.0.96:4756 11 20 10
4 17 10.0.0.138:513 10.0.0.140:513 10.0.0.109:513 11 20 10
5 17 10.0.0.138:111 10.0.0.140:111 10.0.0.96:4756 11 20 10

=>nat bindlist
Application Proto Port
RAUDIO(PNA) tcp 7071
H323 tcp 1720
FTP tcp 21
RTSP tcp 554
IRC tcp 6667
RAUDIO(PNA) tcp 7070

=>nat flush

=>nat list
```

**RELATED COMMANDS:**

- **nat load**
  Load saved or default NAPT configuration.
- **nat save**
  Save current NAPT configuration.
nat list
Show NAPT connection database.

SYNTAX:

<table>
<thead>
<tr>
<th>nat list</th>
<th>[addr = &lt;ip-address&gt;]</th>
</tr>
</thead>
</table>

- [addr] The SpeedTouch™ Pro with Firewall IP address for which the NAPT connection database must be shown.
- OPTIONAL In case the parameter is not specified the NAPT connection database for all IP addresses is shown.

EXAMPLE INPUT/OUTPUT:

```
=> nat list
  Index  Prot Inside-addr:PortOutside-addr:Port Foreign-addr:Port Flgs Expir State Control
  1      6  10.0.0.138:80  172.16.0.5:1080 0.0.0.0:0 19 8 9
  2      17 10.0.0.138:135  10.0.0.140:135 10.0.0.155:1034 11 20 10
  3      17 10.0.0.138:138  10.0.0.140:138 10.0.0.20:138 11 20 10
  4      17 10.0.0.138:137  10.0.0.140:137 10.0.0.254:13711 20 10
  5      17 10.0.0.138:7938  10.0.0.140:7938 10.0.0.96:4756 11 20 10
  6      17 10.0.0.138:513  10.0.0.140:513 10.0.0.109:513 11 20 10
  7      17 10.0.0.138:111  10.0.0.140:111 10.0.0.96:4756 11 20 10
=>
```

RELATED COMMANDS:

- **nat create** Create a static NAPT entry.
- **nat delete** Delete a static NAPT entry.
- **nat disable** Disable NAPT on one of the SpeedTouch™ Pro with Firewall IP addresses.
- **nat enable** Enable NAPT on one of the SpeedTouch™ Pro with Firewall IP addresses.
**nat load**

Load saved (or default) NAPT configuration.

Execute **nat flush** prior to **nat load**.

**SYNTAX:**

```
nat load [defaults = <yes | no>]
```

[defaults] Load factory defaults (yes) or saved configuration (no).

Not specifying this parameter loads the saved configuration

**EXAMPLE:**

```
=>nat list
Indx Prot Inside-addr:PortOutside-addr:Port Foreign-addr:Port Flgs Expir State Control
1 17 10.0.0.138:138 10.0.0.140:138 10.0.0.20:138 11 20 10
2 17 10.0.0.138:137 10.0.0.140:137 10.0.0.254:137 11 20 10
3 17 10.0.0.138:7938 10.0.0.140:7938 10.0.0.96:4756 11 20 10
4 17 10.0.0.138:513 10.0.0.140:513 10.0.0.109:513 11 20 10
5 17 10.0.0.138:111 10.0.0.140:111 10.0.0.96:4756 11 20 10
=>nat save
=>nat flush
=>nat list
```

**RELATED COMMANDS:**

- **nat flush**  Flush complete NAPT configuration.
- **nat save**  Save current NAPT configuration.
nat save
Save current NAPT configuration.

SYNTAX:

nat save

EXAMPLE:

=>nat list
<table>
<thead>
<tr>
<th>Index</th>
<th>Protocol</th>
<th>Inside-addr:Port</th>
<th>Outside-addr:Port</th>
<th>Foreign-addr:Port</th>
<th>Flgs</th>
<th>Expir</th>
<th>State</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>17</td>
<td>10.0.0.138:138</td>
<td>10.0.0.140:138</td>
<td>10.0.0.20:138</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>17</td>
<td>10.0.0.138:137</td>
<td>10.0.0.140:137</td>
<td>10.0.0.254:137</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>17</td>
<td>10.0.0.138:7938</td>
<td>10.0.0.140:7938</td>
<td>10.0.0.96:4756</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>17</td>
<td>10.0.0.138:513</td>
<td>10.0.0.140:513</td>
<td>10.0.0.109:513</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>17</td>
<td>10.0.0.138:111</td>
<td>10.0.0.140:111</td>
<td>10.0.0.96:4756</td>
<td>11</td>
<td>20</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

=>nat save

=>nat flush

=>nat list

=>nat load

=>nat list

RELATED COMMANDS:

nat flush
Flush complete NAPT configuration.

nat load
Load saved or default NAPT configuration.
**nat unbind**
Delete an existing helper/port binding.

**SYNTAX:**

```
nat unbind  
  application = <string>  
  port = <TCP/UDP service name or port number>
```

- **application**  
The name of a NAPT application helper.  
The name must be spelled exactly as listed in the application list (**nat applist**).

- **port**  
The port number this application handler should work on.

**EXAMPLE:**

```
=> nat applist
  Application   Proto  DefaultPort
  ilsa           tcp    0 OUTGOING
  H254           tcp    0 OUTGOINGINCOMING
  H323           tcp    1720 OUTGOINGINCOMING
  RAUDIO(PNA)    tcp    7070 OUTGOING
  RTSP           tcp    554  OUTGOING
  IRC            tcp    6667 OUTGOING
  FTP            tcp    21   OUTGOINGINCOMING

=> nat bindlist
  Application   Proto  Port
  RAUDIO(PNA)    tcp    7071
  H323           tcp    1720
  FTP            tcp    21
  RTSP           tcp    554
  IRC            tcp    6667
  RAUDIO(PNA)    tcp    7070

=> nat unbind application=RAUDIO(PNA) port=7071

=> nat bindlist
  Application   Proto  Port
  H323           tcp    1720
  FTP            tcp    21
  RTSP           tcp    554
  IRC            tcp    6667
  RAUDIO(PNA)    tcp    7070
```

**RELATED COMMANDS:**

- **nat applist**  
List available NAPT protocol helpers.

- **nat bindlist**  
List current NAPT helper/port bindings.

- **nat bind**  
Create a new helper/port binding.
12 Phonebook Commands

- phonebook (to access the Phonebook level)
- phonebook add
- phonebook autolist
- phonebook delete
- phonebook flush
- phonebook list
- phonebook load
- phonebook save
**phonebook add**

Add a phonebook entry.
The number of entries is limited to 64. The number of active connections is limited to 12, but more may be configured at the same time.

**SYNTAX:**

```
| phonebook add | name = <string> |
|              | addr = <[port.]vpi.vci> |
|              | type = <{any|bridge|ppp|cip|ans|pppt}> |
```

**name**

A free to choose phonebook name for the destination.
Two limitations apply:
- The name of a phonebook entry intended for the **Relayed PPPoA** (PPPoA-to-PPTP Relaying) packet service may not start with capital P or capital T.
- The name of a phonebook entry intended for the **PPP-to-DHCP spoofing** packet service must start with DHCP, e.g. ‘DHCP_Spoof01’.

**addr**

The ATM address for this destination.
It is composed of a Virtual Path Identifier (VPI) and a Virtual Channel Identifier (VCI) identifying ATM virtual channels.
In most cases the values are provided by the Service Provider.
Accepted VPI: a number between 0 and 15
Accepted VCI: a number between 0 and 511.

**type**

The Connection Service supported by the destination.
Choose between:
- any (All Packet Services)
- bridge (Bridging, Routed Ethernet, Bridged PPPoE, Routed PPPoE)
- ppp (Routed PPPoA and Relayed PPPoA)
- cip (Classical IP & IP Routing)
- ans (ATM Name Service)
- pppp (Relayed PPPoA, PPPoA-to-PPTP Relaying).
EXAMPLE:

`=>phonebook list
Name    Type  Use Address
PVC1    any   1    8.35
PVC2    bridge 0    8.36
Br3     bridge 0    8.36
Br4     bridge 0    8.38
CIPPVC3 cip   1    8.82
CIPPVC4 cip   1    8.83

=>phonebook add name=Alcatel addr=8.68 type=ppp
=>phonebook list
Name    Type  Use Address
PVC1    any   1    8.35
PVC2    bridge 0    8.36
Br3     bridge 0    8.36
Br4     bridge 0    8.38
CIPPVC3 cip   1    8.82
CIPPVC4 cip   1    8.83
Alcatel  ppp   0    8.68
=>`

RELATED COMMANDS:

- **phonebook delete**  Remove a phonebook entry.
- **phonebook list**      Show current phonebook.
phonebook autolist
Show auto PVCs, if supported by the Central Office DSLAM. (Only applicable for Alcatel ASAM DSLAMs).

SYNTAX:
```
phonebook autolist
```

EXAMPLE INPUT/OUTPUT:
```
=> phonebook autolist
  8.35
=>
```

RELATED COMMANDS:
phonebook list Show current phonebook.
**phonebook delete**

Remove an unused phonebook entry.

**SYNTAX:**

```
phonebook delete name = <string>
```

*name* 
the name of the phonebook entry to delete. 

**REQUIRED**

Only applicable for phonebook entries that are not used, i.e. not configured for any packet service. 

Execute **phonebook list** to check whether the entry is used (Use=1) or not (Use=0).

**EXAMPLE:**

```
-->phonebook list
Name    Type  Use Address
PVC1    any  1  8.35
PVC2    bridge  0  8.36
Br3     bridge  0  8.36
Br4     bridge  0  8.38
CIPPVC3 cip  1  8.82
CIPPVC4 cip  1  8.83
Alcatel ppp  0  8.68

-->phonebook delete name=Alcatel
-->phonebook list
Name    Type  Use Address
PVC1    any  1  8.35
PVC2    bridge  0  8.36
Br3     bridge  0  8.36
Br4     bridge  0  8.38
CIPPVC3 cip  1  8.82
CIPPVC4 cip  1  8.83

--> 
```

**RELATED COMMANDS:**

- **phonebook add** Add a phonebook entry.
- **phonebook list** Show current phonebook.
**phonebook flush**

Flush complete phonebook.
The flush command does not impact previously saved configurations.

**SYNTAX:**

```
phonebook flush
```

**EXAMPLE:**

```
=>phonebook list
Name    Type Use Address
PVC1    any 1    8.35
PVC2    bridge 0   8.36
Br3     bridge 0    8.36
Br4     bridge 0   8.38
CIPPVC3  cip 1      8.82
CIPPVC4  cip 1   8.83
Alcatel  ppp 0     8.68
=>phonebook flush
=>phonebook list
Name    Type Use Address
=>
```

**RELATED COMMANDS:**

- **phonebook load**  
  Load saved or default phonebook.
- **phonebook save**  
  Save current phonebook.
**phonebook list**
Show current phonebook.

**SYNTAX:**

```
phonebook list [opt = <{long}>]
```

[**opt**] Select output format. For internal use only.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Use</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>PVC1</td>
<td>any</td>
<td>1</td>
<td>8.35</td>
</tr>
<tr>
<td>PVC2</td>
<td>bridge</td>
<td>0</td>
<td>8.36</td>
</tr>
<tr>
<td>Br3</td>
<td>bridge</td>
<td>0</td>
<td>8.36</td>
</tr>
<tr>
<td>Br4</td>
<td>bridge</td>
<td>0</td>
<td>8.38</td>
</tr>
<tr>
<td>CIPPVC3</td>
<td>cip</td>
<td>1</td>
<td>8.82</td>
</tr>
<tr>
<td>CIPPVC4</td>
<td>cip</td>
<td>1</td>
<td>8.83</td>
</tr>
<tr>
<td>Alcatel</td>
<td>ppp</td>
<td>0</td>
<td>8.68</td>
</tr>
</tbody>
</table>

**RELATED COMMANDS:**

- **phonebook add**  Add a phonebook entry.
- **phonebook autolist**  Show auto PVCs.
- **phonebook delete**  Remove a phonebook entry.
### phonebook load

Load saved (or default) phonebook.

Execute `phonebook flush` prior to `phonebook load`.

**SYNTAX:**

| `phonebook load` | `[defaults <yes|no>]` |
|-----------------|----------------------|
| **[defaults]**  | Load factory defaults (yes) or saved configuration (no). | OPTIONAL |
|                 | Not specifying this parameter loads the saved configuration |

**EXAMPLE:**

```plaintext
=> phonebook list
Name    Type  Use   Address
PVC2    bridge 0  8.36
Br4     bridge 0  8.38
CIPPVC4  cip  1  8.83
Alcatel  ppp  1  8.68
=> phonebook save
=> phonebook flush
=> phonebook list
Name    Type  Use   Address
=> phonebook load defaults=yes
=> phonebook list
Name    Type  Use   Address
Br1     bridge 0  8.35
Br2     bridge 0  8.36
Br3     bridge 0  8.37
Br4     bridge 0  8.38
RELAY_PPP1 ppp  0  8.48
RELAY_PPP2 ppp  0  8.49
RELAY_PPP3 ppp  0  8.50
RELAY_PPP4 ppp  0  8.51
DIAL_PPP1 ppp  0  8.64
DIAL_PPP2 ppp  0  8.65
DIAL_PPP3 ppp  0  8.66
DHCP_SPOOF ppp  0  8.67
CIPPVC1  cip  0  8.80
CIPPVC2  cip  0  8.81
CIPPVC3  cip  1  8.82
CIPPVC4  cip  1  8.83
=> phonebook load defaults=no
=> phonebook list
Name    Type  Use   Address
PVC2    bridge 0  8.36
Br4     bridge 0  8.38
CIPPVC4  cip  1  8.83
Alcatel  ppp  0  8.68
=> phonebook save
```

**RELATED COMMANDS:**

- `phonebook flush`  
  Flush complete phonebook.
- `phonebook save`  
  Save current phonebook.
**phonebook save**

Save current phonebook.

**SYNTAX:**

```
phonebook save
```

**EXAMPLE:**

```
=>phonebook list
Name   Type  Use  Address
PVC2   bridge 0  8.36
Br4    bridge 0  8.38
CIPPVC4 cip 1  8.83
Alcatel ppp 1  8.68

=>phonebook save
=>phonebook list
Name   Type  Use  Address
PVC2   bridge 0  8.36
Br4    bridge 0  8.38
CIPPVC4 cip 1  8.83
Alcatel ppp 0  8.68

=>phonebook save
```

**RELATED COMMANDS:**

- **phonebook flush**  
  Flush complete phonebook.
- **phonebook load**  
  Load saved or default phonebook.
13 PPP Commands

ppp (to access the PPP level)
ppp flush
ppp ifadd
ppp ifattach
ppp ifconfig
ppp ifdelete
ppp ifdetach
ppp iflist
ppp load
ppp rtadd
ppp rtdelete
ppp save
**ppp flush**
Flush complete PPP configuration. The flush command does not impact previously saved configurations.

**SYNTAX:**
```plaintext
ppp flush
```

**EXAMPLE:**
```plaintext
=> ppp iflist
PP1:  dest : PPP1
   Retry: 10  QoS default encaps VC-MUX
   mode = IP Routing
   flags= echo magicacccomp mru addr routes savepwd PPPOA
   trans addr = pat mru = 1500
   route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
   user name = guest  password= ********
   admin state= down  oper state= down  link state= not-connected
   LCP : state= initial retransm= 10 term.reason =
   IPCP : state= initial retransm= 0 term.reason =
=> ppp flush
=> ppp iflist
=>
```

**RELATED COMMANDS:**
- **ppp load**  Load saved or default PPP configuration.
- **ppp save**  Save current PPP configuration.
**ppp ifadd**
Create a new PPP interface.

SYNTAX:

```
ppp ifadd  [intf = <string>]
            [dest = <phonebook entry>]
            [encaps = {vcmux|llc}]
            [speed = <number(4800-10000000)>]
```

- **[intf]**  The name for the new PPP interface. If not specified, the destination parameter must be specified. In this case the name of the destination will double as interface name.
  
- **[dest]**  The destination for the new PPP interface. Typically, an phonebook entry.
  
- **[encaps]**  The type of encapsulation to be used for this PPP interface. Choose between:
  - vcmux
  - llc/snap

- **[speed]**  A number between 4800 and 10000000 (bits per second). Represents the speed of the peer-to-peer connection. Use for backward compatibility. Use Quality Of Service instead.
EXAMPLE:

```bash
=> ppp iflist
PPP1:  dest : PPP1
  Retry : 10  QoS default encaps VC-MUX
  mode = IP Routing
  flags = echo magicaccomp mru addr routesavepwd PPPOA
  transaddr = pat mru = 1500
  route = 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
  user name = guest password= ********
  admin state = down oper state = down link state = not-connected
  LCP : state = initial retransm= 10 term.reason =
  IPCP : state = initial retransm= 0 term.reason =

=> ppp ifadd intf=PPP2 dest=PVC2
=> ppp iflist
PPP1:  dest : PPP1
  Retry : 10  QoS default encaps VC-MUX
  mode = IP Routing
  flags = echo magicaccomp mru addr routesavepwd PPPOA
  transaddr = pat mru = 1500
  route = 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
  user name = guest password= ********
  admin state = down oper state = down link state = not-connected
  LCP : state = initial retransm= 10 term.reason =
  IPCP : state = initial retransm= 0 term.reason =

PPP2:  dest : PVC2
  Retry : 10  QoS default encaps VC-MUX
  mode = IP Routing
  flags = echo magicaccomp restart mru addr savepwd PPPOA
  mru = 1500
  user name = password=
  admin state = down oper state = down link state = not-connected
  LCP : state = initial retransm= 10 term.reason =
  IPCP : state = initial retransm= 0 term.reason =
```

RELATED COMMANDS:

- `ppp ifattach`: Attach a PPP interface.
- `ppp ifconfig`: Configure a PPP interface.
- `ppp ifdelete`: Delete a PPP interface.
- `ppp ifdetach`: Detach a PPP interface.
- `ppp iflist`: Show current PPP configuration.
**ppp ifattach**

Attach (i.e. connect) a PPP interface.

**SYNTAX:**

```
ppp ifattach   intf = <ifname>
```

- `intf`  The name of the PPP interface to attach. **REQUIRED**

**EXAMPLE:**

```
=> ppp iflist
PPP1:   dest : PPP1
          Retry: 10  QoS default encaps LLC
          mode = IP Routing
          flags= echo magic accomp restart mru addr routes savepwd PPPOE
          trans addr = pat mru = 1492
          route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
          user name = guest password= ********
          adminstate= down oper state= down  link state= not-connected
          LCP : state= initial retransm= 10 term.reason =
          IPCP : state= initial retransm= 0 term.reason =

=> ppp ifattach =intf=PPP1
=> ppp iflist
PPP1:   dest : PPP1
          Retry: 10  QoS default encaps LLC
          mode = IP Routing
          flags= echo magic accomp restart mru addr routes savepwd PPPOE
          trans addr = pat mru = 1492
          route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
          user name = guest password= ********
          adminstate= up oper state= down  link state= connected
          LCP : state= reqsent retransm= 10 term.reason =
          IPCP : state= initial retransm= 10 term.reason =

=> ppp iflist
PPP1:   dest : PPP1
          Retry: 10  QoS default encaps LLC
          mode = IP Routing
          flags= echo magic accomp restart mru addr routes savepwd PPPOE
          trans addr = pat mru = 1492
          route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
          user name = guest password= ********
          adminstate= up oper state= up  link state= connected
          LCP : state= opened retransm= 0 term.reason =
          IPCP : state= opened retransm= 10 term.reason =
          acname : , service = .
```

**RELATED COMMANDS:**

- `ppp ifadd`  Create a PPP interface.
- `ppp ifconfig`  Configure a PPP interface.
- `ppp ifdelete`  Delete a PPP interface.
- `ppp ifdetach`  Detach a PPP interface.
- `ppp iflist`  Show current PPP configuration.
**ppp ifconfig**
Configure a PPP interface. As the PPP interface to be configured may not be connected at the time of configuration, execute **ppp ifdetach** prior to executing the **ppp ifconfig** command.

**SYNTAX:**

```
ppp ifconfig
  intf = <ifname>
  [dest = <phonebook entry>]
  [user = <string>]
  [password = <string>]
  [qos = <string>]
  [proto = <{pppoe|pppoa}>]
  [acname = <string>]
  [servicename = <string>]
  [encaps = <{vcmux|lcl}>]
  [pcomp = <{off|on}>]
  [accomp = <{on|off|negotiate}>]
  [trace = <{off|on}>]
  [pap = <{off|on}>]
  [restart = <{off|on}>]
  [retryinterval = <number{0–65535}>]
  [passive = <{off|on}>]
  [silent = <{off|on}>]
  [echo = <{off|on}>]
  [mru = <number{293–8192}>]
  [laddr = <ip–address>]
  [raddr = <ip–address>]
  [savepwd = <{off|on}>]
  [demanddial = <{off|on}>]
  [primdns = <ip–address>]
  [secdns = <ip–address>]
  [idle = <number{0–1000000}>]
  [addrtrans = <{none|pat}>]
  [unnumbered = <{off|on}>]
  [poolstart = <ip–address>]
  [poolend = <ip–address>]
  [status = <{down|up}>]
```

**intf**
The name of the PPP interface to configure. **REQUIRED**

**[dest]**
The destination for this PPP interface. Typically, a phonebook entry.

Use:
- PPPoA (ppp) phonebook entries
  For the Routed PPPoA (PPP & IP Routing) packet service.
- ETHoA (bridge) phonebook entries
  For the Routed PPPoE packet service.

**[user]**
The user name for remote PAP/CHAP authentication. **OPTIONAL**

**[password]**
The password for remote PAP/CHAP authentication. **OPTIONAL**
[qos] The name of a configured Quality Of Service book entry. This parameter never needs to be specified.

[proto] The encapsulation method for the PPP frames, i.e. the applicable packet service for the connection. Select:
- pppoa
  For a Routed PPPoA (PPP & IP Routing) connection.
- pppoe
  For a Routed PPPoE connection.
Per default the PPPoA protocol applies.

[acname] The Access Concentrator name for a Routed PPPoE connection. This parameter is applicable only for Routed PPPoE PPP interfaces (proto=pppoe). Execute the **ppp ifscan** command to see the names of available access concentrators, if any.

[servicename] The Service Name for a Routed PPPoE connection. This parameter is applicable only for Routed PPPoE PPP interfaces (proto=pppoe). Execute the **ppp ifscan** command to see the available service names, if any.

[encaps] The type of encapsulation to be used for this PPP interface. Choose between:
- vcmux (default)
  Standard encapsulation method for PPPoA (ppp) frames.
- ilc
  Standard encapsulation method for ETHoA (bridge) frames.

[pcomp] Try (on) or do not try (off) to negotiate PPP protocol compression (LCP PCOMP). Per default the negotiation is disabled (off).

[accomp] Try (on), do never try (off) or negotiate (negotiate) to negotiate PPP address & control field compression (LCP ACCOMP). In the very most cases LCP ACCOMP should not be disabled nor negotiated, i.e. the address field FF-03 should not be sent over ATM. Therefore by default this parameter is enabled (on). In case the accomp parameter is set ‘negotiate’ the local side of the PPP connection demands to do ACCOMP and adapts itself to the result of this negotiation.

[trace] Enable (on) or disable (off) verbose console logging. By default tracing is disabled (off).

[pap] Force PAP based authentication (on) or use CHAP based authentication, if available (off). For security reasons PAP negotiation is disabled (off) per default.

[restart] Automatically restart the connection when LCP link goes down (on) or do not restart automatically (off). By default restart is disabled (off).
[retryinterval] A number between 0 and 65535 (seconds). Represents the intermediate interval between two retries to establish the connection on ATM level. Only applicable in an SVC environment.

[passive] Put the link in listening state in case LCP times out (on) or not (off). This parameter allows to determine whether the link should be left open to wait for incoming messages from the remote side after 10 unsuccessful tries to establish the connection or not. Per default the listening state is disabled.

[silent] Do not send anything at startup and just listen for incoming LCP messages (on) or retry up to 10 times to establish the connection (off). Per default the silent state is disabled.

[echo] Send LCP echo requests at regular intervals (on) or not (off). Per default the sending of LCP echo requests is enabled.

[mru] A number between 293 and 8192. Represents the maximum packet size the SpeedTouch™ Pro with Firewall should negotiate to be able to receive.

[laddr] The local IP address of the peer-to-peer connection. Specifying a local IP address forces the remote side of the PPP link (if it allows to) to accept this IP address as the SpeedTouch™ Pro with Firewall PPP session IP address. If not specified, the SpeedTouch™ Pro with Firewall will accept any IP address. Typically the local IP address parameter is not specified.

[raddr] The remote IP address of the peer-to-peer connection. Specifying a remote IP address forces the remote side of the PPP link (if it allows to) to accept this IP address as its PPP session IP address. If not specified, the SpeedTouch™ Pro with Firewall will accept any IP address. Typically the remote IP address parameter is not specified.

[savepwd] Save password (on), if supplied, or do not save the password (off). Per default the saving of the password is disabled.

[demanddial] Enable (on) or disable (off) the dial-on-demand feature.

[primdns] The IP address of the primary DNS server. In case a primary DNS server is specified the SpeedTouch™ Pro with Firewall will negotiate this IP address with the remote side. If not specified, the SpeedTouch™ Pro with Firewall will accept any IP address.

[secdns] The IP address of the (optional) secondary DNS server. In case a secondary DNS server is specified the SpeedTouch™ Pro with Firewall will negotiate this IP address with the remote side. If not specified, the SpeedTouch™ Pro with Firewall will accept any IP address.
[idle] A number between 1 and 1000000 (seconds). Represents after how many seconds an idle link goes down. OPTIONAL

[addrtrans] Automatically enable address translation for the IP address of this link (pat) or do not use address translation (none). OPTIONAL

[unnumbered] Takes the local IP address from ‘laddr’ field and remote IP address from the IP address pool assigned to the incoming PPP link. In case the unnumbered parameter is disabled, the same IP address is used for each connection on the server side, thus reducing the number of IP addresses used. OPTIONAL

[poolstart] The lower bound of the IP address pool assigned to the incoming PPP link. OPTIONAL

[poolend] The upper bound of the IP address pool assigned to the incoming PPP link. OPTIONAL

[status] Force automatically to attach the PPP interface (up) or use the regular ppp ifattach command (down). Per default the startup status is down (recommended). OPTIONAL

EXAMPLE:

```
-> ppp iflist
PPP1: dest : PPP1
   Retry: 10 QoS default encrypt LLC
   mode = IP routing
   flags= echo magic accomp restart mru addr route savepwd PPPOE
   trans addr = pat mru = 1492
   route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
   user name = My_Connection@MY_ISP password= *******
   admin state= down oper state= down link state= not-connected
   LCP : state= initial retransm= 10 term.reason =
   IPCP : state= initial retransm= 0 term.reason =

=> ppp ifconfig interf=PPP1 prot=pppoe encrypt=vc-mux
-> ppp iflist
PPP1: dest : PPP1
   Retry: 10 QoS default encrypt VC-MUX
   mode = IP routing
   flags= echo magic accomp restart mru addr route savepwd PPPOE
   trans addr = pat mru = 1492
   route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
   user name = My_Connection@MY_ISP password= *******
   admin state= down oper state= down link state= not-connected
   LCP : state= initial retransm= 10 term.reason =
   IPCP : state= initial retransm= 0 term.reason =
```

RELATED COMMANDS:

- **ppp ifadd** Create a PPP interface.
- **ppp ifattach** Attach a PPP interface.
- **ppp ifdelete** Delete a PPP interface.
- **ppp ifdetach** Detach a PPP interface.
- **ppp iflist** Show current PPP configuration.
**ppp ifdelete**
Delete a PPP interface.

**SYNTAX:**
```
ppp ifdelete  intf = <ifname>
```

* intf The name of the PPP interface to delete.

**EXAMPLE:**
```
=> ppp iflist
PPP1:  dest : PPP1
  Retry : 10  QoS default encaps VC-MUX
  mode = IP Routing
  flags= echo magicacomp mru addr routes savepwd PPPOA
  trans addr = pat mru = 1500
  route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
  user name = guest  password= ********
  admin state= down  oper state= down  link state= not-connected
  LCP : state= initial retransm= 10 term.reason =
  IPCP : state= initial retransm= 0 term.reason =

PPP2:  dest : PVC2
  Retry : 10  QoS default encaps VC-MUX
  mode = IP Routing
  flags= echo magicacomp restart mru addr savepwd PPPOA
  mru = 1500
  user name = password=
  admin state= down  oper state= down  link state= not-connected
  LCP : state= initial retransm= 10 term.reason =
  IPCP : state= initial retransm= 0 term.reason =
=> ppp ifdelete intf=PPP2
=> ppp iflist
PPP1:  dest : PPP1
  Retry : 10  QoS default encaps VC-MUX
  mode = IP Routing
  flags= echo magicacomp mru addr routes savepwd PPPOA
  trans addr = pat mru = 1500
  route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
  user name = guest  password= ********
  admin state= down  oper state= down  link state= not-connected
  LCP : state= initial retransm= 10 term.reason =
  IPCP : state= initial retransm= 0 term.reason =
=>
```

**RELATED COMMANDS:**
- `ppp ifadd` Create a PPP interface.
- `ppp ifattach` Attach a PPP interface.
- `ppp ifconfig` Configure a PPP interface.
- `ppp ifdetach` Detach a PPP interface.
- `ppp iflist` Show current PPP configuration.
**ppp ifdetach**

Detach a PPP interface.

**SYNTAX:**

```
ppp ifdetach  intf = <ifname>
```

*ppp*  

The name of the PPP interface.  

**EXAMPLE:**

```
=> ppp iflist
PPPI:  dest : PPP1
   Retry: 10  QoS default encap LLC
   mode = IP Routing
   flags= echo magicaccomp restart mru addr routesavedpwd PPPOA
   trans addr = pat  mru = 1492
   route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
   user name = guest  password= ********
   admin state= up  oper state= up  link state= connected
   LCP : state= opened retrans= 0 term.reason =
   IPCP : state= opened retrans= 10 term.reason =
   acname : , service = .
=> ppp ifdetach =intf=PPPI
=> ppp iflist
PPPI:  dest : PPP1
   Retry: 10  QoS default encap LLC
   mode = IP Routing
   flags= echo magicaccomp restart mru addr routesavedpwd PPPOE
   trans addr = pat  mru = 1492
   route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
   user name = guest  password= ********
   admin state= down  oper state= down  link state= not-connected
   LCP : state= initial retrans= 10 term.reason =
   IPCP : state= initial retrans= 0 term.reason =
=>
```

**RELATED COMMANDS:**

- **ppp ifadd**  
  Create a PPP interface.
- **ppp ifattach**  
  Attach a PPP interface.
- **ppp ifconfig**  
  Configure a PPP interface.
- **ppp ifdelete**  
  Delete a PPP interface.
- **ppp iflist**  
  Show current PPP configuration.
ppp iflist
Show current configuration of all or a specified PPP interface(s).

SYNTAX:

```
ppp iflist [intf = <ifname>]
```

- `intf <ifname>` the name of the PPP interface. In case this parameter is not specified all PPP interfaces are shown.

EXAMPLE INPUT/OUTPUT:

```
=> ppp iflist
PPP1:  dest : PPP1
  Retry : 10  QoS default encaps VC-MUX
  mode = IP Routing
  flags= echo magicaccomp mru addr routesavepwd PPPOA
  transaddr = pat mru = 1500
  route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
  user name = guest password= ********
  admin state= down oper state= down link state= not-connected
  LCP : state= initial retransm= 10 term.reason =
  IPCP : state= initial retransm= 0 term.reason =

PPP2:  dest : PVC2
  Retry : 10  QoS default encaps VC-MUX
  mode = IP Routing
  flags= echo magicaccomp restart mru addr savespwd PPPOA
  mru = 1500
  user name = password=
  admin state= down oper state= down link state= not-connected
  LCP : state= initial retransm= 10 term.reason =
  IPCP : state= initial retransm= 0 term.reason =
=>
```

RELATED COMMANDS:

- `ppp ifadd` Create a PPP interface.
- `ppp ifattach` Attach a PPP interface.
- `ppp ifconfig` Configure a PPP interface.
- `ppp ifdelete` Delete a PPP interface.
- `ppp ifdetach` Detach a PPP interface.
**ppp ifscan**

Scan a PPPoE interface (proto=pppoe) for available Access Concentrator names and Service Names.

Execute the **ppp ifdetach** command for this interface prior to perform a scan on it.

**SYNTAX:**

```
ppp ifscan  intf = <ifname>
   [time = <number{0–36000}>]
   [kit = <number{0–8}>]
```

- **intf**  
  The name of the PPP interface to scan.  
  REQUIRED

- **[time]**  
  A number between 0 and 36000 (seconds).  
  Represents the time to scan for services.  
  OPTIONAL

- **[kit]**  
  A number between 0 and 8.  
  Represents the way the scan progress is visually indicated.  
  Per default no progress indicator is applied (kit=0).  
  kit=1 up to kit=8 are diverse progress indicators. Try it!
  OPTIONAL

**EXAMPLE:**

```
=> ppp iflist
PPPI:  dest : PPP1
    Retry: 10  QoS default encaps LLC
    mode = IP Routing
    flags = echo magic accomp restart mru addr routesavedw PPPOE
    trans addr = pat mru = 1492
    route = 0.0.0.0/0 – 0.0.0.0/0 (metric 0)
    user name = guest password= ********
    admin state = guest password= ********
    link state = not-connected
    LCP : state = initial retransm= 10 term.reason =
    IPCP : state = initial retransm= 0 term.reason =

=> ppp ifscan intf=PPP1 time=45
    Service Name Access Concentrator
```

**RELATED COMMANDS:**

- **ppp ifconfig**  
  Configure a PPP interface.
**ppp load**

Load saved (or default) PPP configuration.

Execute **ppp flush** prior to **ppp load**.

**SYNTAX:**

```
ppp load [defaults = <yes|no>]
```

- **[defaults]**
  - Load factory defaults (yes) or saved configuration (no).
  - Not specifying this parameter loads the saved configuration

**EXAMPLE:**

```
=> ppp iflist
PPP1: dest : PPP1
    Retry: 10 QoS default encap VC-MUX
    mode = IP Routing
    flags = echo magicaccomp mrup addr routes savepwd PPPOA
    trans addr = pat  mru = 1500
    route = 0.0.0.0/0  0.0.0.0/0 (metric 0)
    user name = guest password= ********
    admin state = down oper state = down link state = not-connected
    LCP : state = initial retransm= 10 term. reason =
    IPCP : state = initial retransm= 0 term. reason =
=> ppp save
=> ppp flush
=> ppp iflist
=>** ppp load
=>** ppp iflist
PPP1: dest : PPP1
    Retry: 10 QoS default encap VC-MUX
    mode = IP Routing
    flags = echo magicaccomp mrup addr routes savepwd PPPOA
    trans addr = pat  mru = 1500
    route = 0.0.0.0/0  0.0.0.0/0 (metric 0)
    user name = guest password= ********
    admin state = down oper state = down link state = not-connected
    LCP : state = initial retransm= 10 term. reason =
    IPCP : state = initial retransm= 0 term. reason =
=>
```

**RELATED COMMANDS:**

- **ppp flush**
  - Flush complete PPP configuration.
- **ppp save**
  - Save current PPP configuration.
**ppp rtadd**

Automatically add a route configuration to the routing table in case the specified PPP interface link comes up.

This route configuration will determine which local hosts are allowed to use this link and/or which remote destinations should be or should not be reachable.

Execute the **ppp ifdetach** command for this interface prior to configuring routes.

**SYNTAX:**

<table>
<thead>
<tr>
<th><strong>ppp rtadd</strong></th>
<th><strong>intf</strong> = <strong>&lt;ifname&gt;</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>dst</strong> = <strong>&lt;ip-address&gt;</strong></td>
</tr>
<tr>
<td></td>
<td><strong>[dstmsk</strong> = <strong>&lt;ip-mask(dotted or cidr)&gt;]</strong></td>
</tr>
<tr>
<td></td>
<td><strong>[src</strong> = <strong>&lt;ip-address&gt;]</strong></td>
</tr>
<tr>
<td></td>
<td><strong>[srcmsk</strong> = <strong>&lt;ip-mask(dotted or cidr)&gt;]</strong></td>
</tr>
<tr>
<td></td>
<td><strong>[metric</strong> = <strong>&lt;number(0–100)&gt;]</strong></td>
</tr>
</tbody>
</table>

- **intf** The name of the PPP interface. REQUIRED
- **dst** The destination IP address specification for the route to be added when the link comes up. REQUIRED
- **[dstmsk]** The destination IP mask. Depending on the destination netmask:
  - Any remote destination is reachable, i.e. the PPP connection acts as default route (dstmsk=0)
  - Only the remote (sub)net is reachable (dstmsk=1)
    - The actual destination mask will be the default netmask applicable for destination IP address
  - Only the single remote host is reachable (dstmsk=32)
  - Any valid (contiguous) netmask in case of VLSM. OPTIONAL
- **[src]** The source IP address specification for the route to be added when the link comes up. OPTIONAL
- **[srcmsk]** The source IP mask. Depending on the source netmask:
  - Everybody is allowed to use this PPP connection (dstmsk=0)
  - Only members of the same subnet as the host which opened the PPP connection are allowed to use the PPP connection (dstmsk=1)
    - The actual destination mask will be the netmask applicable for the IP address of the host which opened the PPP connection.
  - Only the host which opened the PPP connection is allowed to use the PPP connection. (dstmsk=32)
  - Any valid (contiguous) netmask in case of VLSM. OPTIONAL
- **[metric]** The route metric, i.e. the cost factor of the route. Practically, the cost is determined by the hop count.
  - It is recommended not to use this parameter. OPTIONAL
EXAMPLE:

```plaintext
=> ppp iflist
PPP1: dest : PVC3
Retry: 10 QoS default encaps LLC
mode = IP Routing
flags= echo magicaccomp restart mru addr routesavepwd PPPOE
transaddr = pat mru = 1492
user name = guest password= ********
admin state = down oper state = down link state = not-connected
LCP : state = initial retransm= 10 term.reason =
IPCP : state = initial retransm= 0 term.reason =
=> ppp rtadd intf=PPP1 dst=172.16.0.5 dstmsk=24 src=10.0.0.2 srcmask=24
=> ppp iflist
PPP1: dest : PVC3
Retry: 10 QoS default encaps LLC
mode = IP Routing
flags= echo magicaccomp restart mru addr routesavepwd PPPOE
transaddr = pat mru = 1492
route = 10.0.0.2/24 – 172.16.0.5/24 (metric 1)
user name = guest password= ********
admin state = down oper state = down link state = not-connected
LCP : state = initial retransm= 10 term.reason =
IPCP : state = initial retransm= 0 term.reason =
=>
```

RELATED COMMANDS:

- **ppp rtdelete**  
  Delete the route specification for an upcoming PPP link.
**ppp rtdelete**
Delete the route specification for a PPP link.
Execute the **ppp ifdetach** command for this interface prior to deleting route configurations.

**SYNTAX:**

<table>
<thead>
<tr>
<th><strong>ppp rtdelete</strong></th>
<th><strong>inf = &lt;ifname&gt;</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>inf</strong></td>
<td>The PPP interface name for which to delete the route settings. REQUIRED</td>
</tr>
</tbody>
</table>

**EXAMPLE:**

```plaintext
=> ppp iflist
PPPI:  dest : PVC3
    Retry: 10  QoS default encaps LLC
    mode = IP Routing
    flags= echo magicaccomp restart mru addr routesavepwd PPPOE
    trans addr = pat mru = 1492
    route= 10.0.0.2/24 - 172.16.0.5/24 (metric 1)
    user name = guest  password= ********
    admin state= down oper state= down link state= not-connected
    LCP : state= initial retransm= 10 term.reason =
    IPCP : state= initial retransm= 0 term.reason =

=> ppp rtdelete inf=PPPI
=> ppp iflist
PPPI:  dest : PVC3
    Retry: 10  QoS default encaps LLC
    mode = IP Routing
    flags= echo magicaccomp restart mru addr routesavepwd PPPOE
    trans addr = pat mru = 1492
    user name = guest  password= ********
    admin state= down oper state= down link state= not-connected
    LCP : state= initial retransm= 10 term.reason =
    IPCP : state= initial retransm= 0 term.reason =
=>
```

**RELATED COMMANDS:**

**ppp rtadd** Configure a route specification for an upcoming PPP link.
**ppp save**

Save current PPP configuration.

**SYNTAX:**

```
ppp save
```

**EXAMPLE:**

```
=>ppp iflist
PPP1:  dest : PPP1
   Retry : 10  QoS default encap VC-MUX
   mode = IP Routing
     flags = echo magicaccomp mru addr routesavedpw PPPOA
   transaddr = pat mru = 1500
   route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
   user name = guest password= ********
   admin state = down oper state = down link state = not-connected
   LCP : state= initial retransm= 10 term.reason =
   IPCP : state= initial retransm= 0 term.reason =

=>ppp save
=>ppp flush
=>ppp iflist
=>ppp load
=>ppp iflist
PPP1:  dest : PPP1
   Retry : 10  QoS default encap VC-MUX
   mode = IP Routing
     flags = echo magicaccomp mru addr routesavedpw PPPOA
   transaddr = pat mru = 1500
   route= 0.0.0.0/0 - 0.0.0.0/0 (metric 0)
   user name = guest password= ********
   admin state = down oper state = down link state = not-connected
   LCP : state= initial retransm= 10 term.reason =
   IPCP : state= initial retransm= 0 term.reason =
```

**RELATED COMMANDS:**

- **ppp flush**  Flush complete PPP configuration.
- **ppp load**   Load saved or default PPP configuration.
14 PPTP Commands

- `ppptp` (to access the PPTP level)
- `ppptp flush`
- `ppptp list`
- `ppptp load`
- `ppptp profadd`
- `ppptp profdelete`
- `ppptp proflist`
- `ppptp save`
**pptp flush**
Flush complete PPTP configuration.
The flush command does not impact previously saved configurations.

SYNTAX:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pptp flush</code></td>
<td></td>
</tr>
</tbody>
</table>

EXAMPLE:

```
=>pptp profadd name=Relay_PPP1 encap=nlpid ac=always
=>pptp proflist
Profile  QoS  Encaps  AC
Relay_PPP1  default  nlpid  always
=>pptp flush
=>pptp proflist
=>
```

RELATED COMMANDS:

- **pptp load** Load saved or default PPTP configuration.
- **pptp save** Save current PPTP configuration.
**pptp list**
Show current PPTP configuration.

SYNTAX:

```
pptp list
```

EXAMPLE INPUT/OUTPUT:

```
=>pptp list
Dialstr  Destination  QoS  Encaps  AC  State    User
=>      DIALUP_PPP3  default  vcmux  never  CONNECTED  (10.0.0.2)
```
**pptp load**
Load saved (or default) PPTP configuration.
Execute *pptp flush* prior to *pptp load*.

**SYNTAX:**

| *pptp load* | *(defaults = <yes|no>)* |
|-------------|-------------------------|

*[defaults]*
Load factory defaults (yes) or saved configuration (no).
Not specifying this parameter loads the saved configuration

**EXAMPLE:**

```plaintext
=>pptp proflist
Profile   QoS     Encaps  AC
Relay_PPP1 default nlpid  always
PPTPLink   default vcmux  never

=>pptp save
=>pptp flush
=>pptp proflist

=>pptp load defaults=yes
Profile   QoS     Encaps  AC
default   default vcmux  never

=>pptp load defaults=no
=>pptp proflist
Profile   QoS     Encaps  AC
Relay_PPP1 default nlpid  always
PPTPLink   default vcmux  never
```

**RELATED COMMANDS:**

- **pptp flush**
  Flush complete PPTP configuration.
- **pptp save**
  Save current PPTP configuration.
**pptp profadd**
Define a new PPTP profile.

**SYNTAX:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Syntax</th>
</tr>
</thead>
</table>
| **pptp profadd** | *name* = *<string>*  \[
|                 |   *qos* = *<string>*  \[
|                 |   *encaps* = *<{vcmux|nlpid}>*  \[
|                 |   *ac* = *<{never|always|keep}>*  \] |

- **name**  The name for the PPTP profile. `REQUIRED`
- **qos**  The name of the Quality Of Service book entry. This parameter never needs to be specified. `OPTIONAL`
- **encaps**  The type of encapsulation applicable to Relayed PPPoA interfaces using this PPTP profile. Choose between: `vcmux` `nlpid` `OPTIONAL`
- **ac**  The HDLC framing option applicable to Relayed PPPoA interfaces using this PPTP profile. Before relaying the encapsulated PPP frames over the PPPoA link, make sure that the address and control field (0xFF03) is always in front of the frames (always), make sure the address and control field will never be found in front of the frames (never) or do not change the frames arriving via the PPTP tunnel (keep). By default the address and control field is never sent (compliant to RFC2364). It is recommended to keep this setting. `OPTIONAL`

**EXAMPLE:**

```plaintext
=>pptp proflist
Profile        QoS      Encaps     AC
Relay_PPP1     default  nlpid     always

=>pptp profadd name=PPTPLink encaps=vcmux ac=never
=>pptp proflist
Profile        QoS      Encaps     AC
Relay_PPP1     default  nlpid     always
PPTPLink       default  vcmux     never
=>
```

**RELATED COMMANDS:**
- **pptp profdelete**  Delete a PPTP profile.
- **pptp proflist**  Show current PPTP profiles.
**pptp profdelete**
Delete a PPTP profile.

**SYNTAX:**

```
pptp profdelete name <string>
```

- `name`: The name for the PPTP profile. **REQUIRED**

**EXAMPLE:**

```
=>pptp proflist
Profile          QoS   Encaps  AC
Relay_PPP1       default nlpid always
PPTPLink         default vcmux never

=>pptp profdelete name=PPTPLink
=>pptp proflist
Profile          QoS   Encaps  AC
Relay_PPP1       default nlpid always
=>
```

**RELATED COMMANDS:**

- `pptp profadd` Define a new PPTP profile.
- `pptp proflist` Show current PPTP profiles.
**puppet proflist**

Show all current PPTP profiles.

**EXAMPLE:**

```
=>puppet proflist
Profile QoS Encaps AC
Relay_PPPI default nlpid always
PPTPLink default vcmux never
```

**RELATED COMMANDS:**

- **puppet profadd**
  - Define a new PPTP profile.
- **puppet profdelete**
  - Delete a PPTP profile.
**pptp save**
Save current PPTP configuration.

**SYNTAX:**

```
pptp save
```

**EXAMPLE:**

```
=>pptp proflist
Profile               QoS Encaps   AC
Relay_PPP1            default nlpid always
PPTPLink              default vcmux never
=>
pptp save
=>pptp flush
=>pptp proflist
defaults=no
=>pptp proflist
Profile               QoS Encaps   AC
Relay_PPP1            default nlpid always
PPTPLink              default vcmux never
=>
```

**RELATED COMMANDS:**

- **pptp flush**
  Flush complete PPTP configuration.
- **pptp load**
  Load saved or default PPTP configuration.
15 Software Commands

- software (to access the Software level)
- software cleanup
- software deletepassive
- software setpassive
- software switch
- software version
software cleanup
Remove all unused files from the passive software subdirectory.
This command frees the passive software subdirectory from corrupted software files and configuration files. Software marked as passive software is not deleted.

SYNTAX:

```
software cleanup
```

EXAMPLE:

```
=> software cleanup
=>
```

RELATED COMMANDS:

- `software deletepassive`  Delete the passive software.
- `software setpassive`     Mark an uploaded file as passive software version.
**software deletepassive**

Delete passive software.

**SYNTAX:**
```
software deletepassive
```

**EXAMPLE:**
```
=> Software version
Active : Sascha3.424 Passive : Bene3.420
=> Software deletepassive
=> Software version
Active : Sascha3.424 Passive :
=>
```

**RELATED COMMANDS:**
- **software cleanup**  
  Remove all unused files from the passive software subdirectory.
- **software setpassive**  
  Mark a file as passive software version.
**software setpassive**

Mark a file as passive software version. Only correctly uploaded software, valid for the **SpeedTouch™ Pro with Firewall** can be marked as passive software.

**SYNTAX:**

```
software setpassive  file = <string>
```

- `file` the filename (without directory path) of the software package. **REQUIRED**

**EXAMPLE:**

```
=>Software version
Active : Sascha3.423  Passive : Bene3.420
=>Software deletepassive
=>Software version
Active : Sascha3.423  Passive :
......
(FTP file transfer or upload via the **SpeedTouch™ Pro with Firewall** pages of new software Sascha3.428)
......
=>software setpassive file=Sascha3.722
=>Software version
Active : Sascha3.423  Passive : Sascha3.428
=>
```

**RELATED COMMANDS:**

- **software cleanup** Remove all unused files from the passive software subdirectory.
- **software deletepassive** Delete passive software.
**software switch**

Switch active and passive versions and reboot the Speed Touch™ Pro with Firewall. Because rebooting implies a flush of all non-saved configurations it is highly recommended to save the current configuration if needed, e.g. by executing the `config save` command prior to executing a software switch.

SYNTAX:

```plaintext
software switch
```

EXAMPLE:

```plaintext
=>Software version
Active : Sascha3.423    Passive : Sascha3.428
=>software switch
.....
(after reboot and re-opening the Telnet session)
.....
=>Software version
Active : Sascha3.428    Passive : Sascha3.423
=>
```

RELATED COMMANDS:

- `software version` Show active and passive software versions.
- `system reboot` Reboot the Speed Touch™ Pro with Firewall.
software version
Show active and passive software versions.

SYNTAX:

```
software version
```

EXAMPLE:

```
=>Software version
Active : Sascha3.428
Passive : Sascha3.423
=>
```

RELATED COMMANDS:

- **software switch**: Switch active and passive software versions and reboot the Speed Touch™ Pro with Firewall.
16 SHDSL Commands

The shdsl command group is only applicable to the Speed Touch™ Pro with Firewall SHDSL variant, NOT to the Speed Touch™ Pro with Firewall ADSL variants.

shdsl (to access the SHDSL level)
shdsl info
shdsl load
shdsl psd
shdsl save
**shdsl info**

Show SHDSL statistics and information about the **SpeedTouch™ Pro with Firewall** status.

**SYNTAX:**

```
shdsl info
```

**EXAMPLE:**

```
=>shdsl info
WAN interface : SHDSL
Line Status    : UP at 2304 kbit/s
Total bytes since power on:
   Downstream : 9932253  Upstream : 244701
PSD mask      : Symmetric PSD mask for Europe
Shdsl uptime  : 47:55:28
SNR margin    : 7 dB
Attenuation   : 24 dB
Transmit Power: 15 dBm
=>
```
**shdsl load**

Load saved (or default) SHDSL configuration.

**SYNTAX:**

```
shdsl load [defaults = <yes|no>]
```

- `[defaults]`  
  Load factory defaults (yes) or saved configuration (no). 
  Not specifying this parameter loads the saved configuration.

**EXAMPLE:**

```bash
=>shdsl load default=yes
=>shdsl info
WAN interface : SHDSL
Line Status : UP at 2304 kbit/s
Total bytes since power on:
  Downstream : 9932253  Upstream : 244701
PSD mask : Symmetric PSD mask for Europe
Shdsl uptime : 0:0:21
SNR margin : 24 dB
Attenuation : 24 dB
Transmit Power : 15 dBm
=>shdsl psd mask=asym_EU_2304
=>shdsl info
WAN interface : SHDSL
Line Status : DATA MODE
Total bytes since power on:
  Downstream : 0  Upstream : 0
PSD mask : Asymmetric PSD mask for 2304kbps (European)
Shdsl uptime : 0:00:00
=>shdsl save
=>shdsl load defaults=yes
=>shdsl info
WAN interface : SHDSL
Line Status : UP at 2304 kbit/s
Total bytes since power on:
  Downstream : 895  Upstream : 632
PSD mask : Symmetric PSD mask for Europe
Shdsl uptime : 00:00:15
SNR margin : 7 dB
Attenuation : 24 dB
Transmit Power : 15 dBm
=>shdsl load defaults=no
=>shdsl info
WAN interface : SHDSL
Line Status : DATA MODE
Total bytes since power on:
  Downstream : 0  Upstream : 0
PSD mask : Asymmetric PSD mask for 2304kbps (European)
Shdsl uptime : 0:00:00
=>
```

**RELATED COMMANDS:**

- **shdsl save**  
  Save current SHDSL configuration.
**shdsl psd**

Configure the PSD mask to be used, then restart the SHDSL line.

**SYNTAX:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>shdsl psd</strong></td>
<td>mask = {sym_NA</td>
</tr>
</tbody>
</table>

- **mask**
  - The selected PSD mask must comply to the regional demands and to the required payload rate.
  - Choose between:
    - **sym\_NA**
      - Symmetric PSD mask for North America
    - **sym\_EU**
      - Symmetric PSD mask for Europe
    - **asym\_NA\_768**
      - Asymmetric PSD mask for 768kbps (North American)
    - **asym\_NA\_1536**
      - Asymmetric PSD mask for 1536kbps (North American)
    - **asym\_EU\_2048**
      - Asymmetric PSD mask for 2048kbps (European)
    - **asym\_EU\_2304**
      - Asymmetric PSD mask for 2304kbps (European)

**EXAMPLE:**

```
=>shdsl info
WAN interface : SHDSL
Line Status : UP at 2304 kbit/s
Total bytes since power on:
  Downstream : 9932253  Upstream : 244701
PSD mask : Symmetric PSD mask for Europe
Shdsl uptime : 0:0:21
SNR margin : 7 dB
Attenuation : 24 dB
Transmit Power : 15 dBm
=>shdsl psd mask=asym\_EU\_2304
=>shdsl info
WAN interface : SHDSL
Line Status : DATA MODE
Total bytes since power on:
  Downstream : 0  Upstream : 0
PSD mask : Asymmetric PSD mask for 2304kbps (European)
Shdsl uptime : 0:00:00
=>
```
**shdsl save**

Save current SHDSL configuration.

**SYNTAX:**

`shdsl save`

**EXAMPLE:**

```
=>shdsl info
WAN interface : SHDSL
Line Status : UP at 2304 kbit/s
Total bytes since power on:
  Downstream : 9932253   Upstream : 244701
PSD mask : Symmetric PSD mask for Europe
Shdsl uptime : 47:55:28
SNR margin : 7 dB
Attenuation : 24 dB
Transmit Power : 15 dBm
=>shdsl psd mask=asym_EU_2304
=>shdsl info
WAN interface : SHDSL
Line Status : DATA MODE
Total bytes since power on:
  Downstream : 0   Upstream : 0
PSD mask : Asymmetric PSD mask for 2304kbps (European)
Shdsl uptime : 0:00:00
=>shdsl save
=>shdsl flush
=>shdsl load defaults=yes
=>shdsl info
WAN interface : SHDSL
Line Status : UP at 2304 kbit/s
Total bytes since power on:
  Downstream : 895   Upstream : 632
PSD mask : Symmetric PSD mask for Europe
Shdsl uptime : 00:00:15
SNR margin : 7 dB
Attenuation : 24 dB
Transmit Power : 15 dBm
=>shdsl flush
=>shdsl load defaults=no
=>shdsl info
WAN interface : SHDSL
Line Status : DATA MODE
Total bytes since power on:
  Downstream : 0   Upstream : 0
PSD mask : Asymmetric PSD mask for 2304kbps (European)
Shdsl uptime : 0:00:03
=>
```

**RELATED COMMANDS:**

`shdsl load`  
Load saved (or default) SHDSL configuration.
17 System Commands

system (to access the System level)
system clearpassword
system flush
system load
system reboot
system save
system setpassword
**system clearpassword**

Clear current SpeedTouch™ Pro with Firewall system password.

To avoid unrestricted and unauthorized access to the SpeedTouch™ Pro with Firewall it is highly recommended always to make sure that it is protected by a SpeedTouch™ Pro with Firewall system password (by executing `system setpassword`) and to change the password regularly.

SYNTAX:
```
system clearpassword
```

EXAMPLE:
```
=> system clearpassword
->
```

RELATED COMMANDS:
```
system setpassword  Set/change current system password.
```
**system flush**
Flush current *SpeedTouch™ Pro with Firewall* system configuration, i.e. the System password. The flush command does not impact previously saved configurations.

To avoid unrestricted and unauthorized access to the *SpeedTouch™ Pro with Firewall* it is highly recommended always to make sure that it is protected by a *SpeedTouch™ Pro with Firewall* system password (by executing *system setpassword*) and to change the password regularly.

**SYNTAX:**

```
  system flush
```

**EXAMPLE:**

```
=>system flush
->
```

**RELATED COMMANDS:**

- **system load**
  - Load saved or default system configuration.
- **system save**
  - Save current system configuration.
**system load**

Load saved (or default) system configuration. Execute **system flush** prior to **system load**.

In most cases loading the default system configuration causes the **SpeedTouch™ Pro with Firewall** system password to be CLEARED. Therefore, to avoid unrestricted and unauthorized access to the **SpeedTouch™ Pro with Firewall** it is highly recommended always to make sure that it is protected by a **SpeedTouch™ Pro with Firewall** system password (by executing **system setpassword**) and to change the password regularly.

**SYNTAX:**

```
  system load [defaults = <yes|no>]
```

- **[defaults]**: Load factory defaults (yes) or saved configuration (no). Not specifying this parameter loads the saved configuration.

**EXAMPLE:**

```
=> system load defaults=no
=>
```

**RELATED COMMANDS:**

- **system flush**: Flush complete system configuration.
- **system save**: Save current system configuration.
**system reboot**

Reboot the **SpeedTouch™ Pro with Firewall**.
Because rebooting implies a flush of all non-saving configurations it is highly recommended to save the current configuration by executing `config save`.

To avoid unrestricted and unauthorized access to the **SpeedTouch™ Pro with Firewall** it is highly recommended always to make sure that the system is protected by a **SpeedTouch™ Pro with Firewall** system password (by executing `system setpassword`) and to save it (by executing `system save`) prior to executing this command.

**SYNTAX:**

```
system reboot
```

**EXAMPLE:**

```
=> system reboot
......
(lost session connectivity due to reboot)
......
```
**system save**

Save current system configuration, i.e. the System password.

To avoid unrestricted and unauthorized access to the **SpeedTouch™ Pro with Firewall** it is highly recommended always to make sure that it is protected by a **SpeedTouch™ Pro with Firewall** system password (by executing **system setpassword**) and to save it (by executing **system save**) prior to executing this command.

**SYNTAX:**

```
**system save**
```

**EXAMPLE:**

```
=> system save

=>
```

**RELATED COMMANDS:**

- **system load** Load saved or default system configuration.
- **system flush** Flush complete system configuration.
**system setpassword**
Set/change the current SpeedTouch™ Pro with Firewall system password.

To avoid unrestricted and unauthorized access to the SpeedTouch™ Pro with Firewall it is highly recommended always to make sure that it is protected by a SpeedTouch™ Pro with Firewall system password and to change it regularly.

**SYNTAX:**

```
system setpassword  password = <string>
```

**password**
A free to choose password `<string>`. The password is limited to a maximum of 15 characters. It is advised only to use characters from the ranges a–z, A–Z, 0–9.

**EXAMPLE:**

```
=> system setpassword password=Sascha
=> system save
=> system reboot
......
```

After reboot a telnet session is opened:

```
/home/doejohn{1}$ telnet 10.0.0.138
Trying 10.0.0.138...
Connected to 10.0.0.138.
Escape character is ‘^]’.
User: SpeedTouch (00–90–D0–00–01–23–45)
Password: Sascha
```

```
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
* * *
```

Alcatel Speed Touch Pro
with Firewall DSL Router
Version R3.4

Copyright (c) 1999–2001 Alcatel

```
=>
```
18 TD Commands

**td** (to access this level)
**td call**
**td call**
Call a ‘Trace & Debug’ command. For qualified personnel only.

**SYNTAX:**

<table>
<thead>
<tr>
<th>td call</th>
<th>cmd = &lt;string&gt;</th>
</tr>
</thead>
</table>

*cmd*  
The quoted trace & debug command string.  
REQUIRED
Alcatel

SpeedTouch™ Pro

with Firewall

CLI Command Index
Index

A
add
  atmfl add, 24
  dhcp server add, 81
  dns add, 102
  phonebook add, 198
apadd, ip apadd, 150
apdelete, ip apdelete, 152
aplist, ip aplist, 153
applist, nat applist, 182
arpadd, ip arppad, 154
arpdelete, ip arpp delete, 155
arpplist, ip arpp list, 156
assign, firewall assign, 124
autolist, phonebook autolist, 200

B
bind, nat bind, 183
bindlist, nat bindlist, 184

C
call, td call, 254
cleanup, software cleanup, 234
clear
  dhcp client clear, 65
  dns clear, 103
  firewall rule clear, 139
clearpassword, system clearpassword, 246
client, dhcp server client, 82
clrs stats
  dhcp server clrs stats, 83
  dns clrs stats, 104
config
  bridge config, 32
  dhcp client config, 66
  dhcp server config, 84
  ip config, 157
create
  firewall chain create, 134
  firewall rule create, 140
  nat create, 185

D
defserver, nat defserver, 187
delete
  atmfl delete, 25
  dhcp server delete, 86
  dns delete, 105
  firewall chain delete, 135
  firewall rule delete, 144
  nat delete, 188
  phonebook delete, 201
delete passive, software delete passive, 235
disable, nat disable, 189
domain, dns domain, 106

E
enable, nat enable, 190
erase, config erase, 58

F
flush
  atmfl flush, 26
  bridge flush, 33
  cip flush, 48
  config flush, 59
  dhcp client flush, 67
  dhcp server flush, 87
  dns flush, 107
  firewall flush, 125
  firewall rule flush, 145
  ip flush, 160
  mer flush, 172
  nat flush, 191
  phonebook flush, 202
  ppp flush, 208
  pptp flush, 226
  system flush, 247
fwdadd, dns fwdadd, 108
fwddelete, dns fwddelete, 109
fwdlist, dns fwdlist, 110
fwdtable, dns fwdtable, 111

H
help, 12

I
ifadd
  bridge ifadd, 34
cip ifadd, 49
dhcp client ifadd, 68
mer ifadd, 173
ppp ifadd, 209
ifattach
  bridge ifattach, 35
dhcp client ifattach, 69
mer ifattach, 174
ppp ifattach, 211
ifconfig
  bridge ifconfig, 36
dhcp client ifconfig, 70
ip ifconfig, 161
mer ifconfig, 175
ppp ifconfig, 212
ifdelete
  bridge ifdelete, 38
cip ifdelete, 50
dhcp client ifdelete, 72
mer ifdelete, 176
ppp ifdelete, 216
ifdetach
  bridge ifdetach, 39
mer ifdetach, 177
ppp ifdetach, 217
iflist
  bridge iflist, 40
cip iflist, 51
dhcp client iflist, 73
ip iflist, 162
mer iflist, 178
ppp iflist, 218
ifrelease, dhcp client ifrelease, 74
ifrenew, dhcp client ifrenew, 76
ifscan, ppp ifscan, 219
info
  adsl info, 22
  shdsl info, 240

L
list
  atmflist, 27
dhcp server list, 88
dns list, 112
firewall chain list, 136
firewall list, 126
firewall rule list, 146
nat list, 192
phonebook list, 203
pptp list, 227
load
  atmfload, 28
  bridge load, 41
cip load, 52
config load, 60
dhcp client load, 78
dhcp server load, 89
dns load, 113
firewall chain load, 137
firewall load, 127
ip load, 163
mer load, 179
nat load, 193
phonebook load, 204
ppp load, 220
pptp load, 228
shdsl load, 241
system load, 248

M
macadd, bridge macadd, 42
macdelete, bridge macdelete, 43
maclist, bridge maclist, 44
match, firewall match, 128

N
nslookup, dns nslookup, 114
P
ping, ip ping, 164
policy, dhcp server policy, 90
profadd, pptp profadd, 229
profdelete, pptp profdelete, 230
proflist, pptp proflist, 231
psd, shdsl psd, 242
pvcadd, cip pvcadd, 53
pvcdelete, cip pvcedele, 54
pvclist, cip pvclist, 55

R
reboot, system reboot, 249
reset, config reset, 61
rtadd
   ip rtadd, 165
   ppp rtadd, 221
rtdelete
   ip rtdelete, 166
   ppp rtdelete, 223
rtlist, ip rtlist, 167

S
save
   atmfsave, 29
   bridge save, 45
   cip save, 56
   config save, 62
   dhcp client save, 79
   dhcp server save, 91
dns save, 115
firewall chain save, 138
firewall save, 131
ip save, 168
mer save, 180
nat save, 194
phonebook save, 205
ppp save, 224
pptp save, 232
shdsl save, 243
system save, 250
sendto, ip sendto, 169
setpassive, software setpassive, 236
setpassword, system setpassword, 251
spoof, dhcp server spoof, 92
start
   dhcp server start, 94
dns start, 116
stats
   dhcp client stats, 80
dhcp server stats, 95
dns stats, 117
firewall rule stats, 147
status
   dhcp server status, 97
dns status, 118
stop
   dhcp server stop, 98
dns stop, 119
switch, software switch, 237

T
toutfwd, dns toutfwd, 120
troff
   dhcp server troff, 99
dns troff, 121
tron
   dhcp server tron, 100
dns tron, 122

U
unbind, nat unbind, 195

V
version, software version, 238