Appendix A

TALnet Command Reference

This appendix describes TALnet commands. TALnet commands allow you to configure or monitor your Wireless Router and manage sessions on the Wireless Router.

In general, these commands are entries in the TALnet configuration file, *talnet.cfg*. Depending on your level of authorization, you can modify this file or you can make changes to the configuration interactively without modifying the file. See the **user** command description for more information about authorization levels. See Chapter 3, “Configuring the TALnet Software,” for the sample configuration file that is preloaded on your Wireless Router.

The TALnet software includes the commands listed in Table A-1. Some commands include rudimentary online help; enter the command and a question mark to learn the syntax.

Table A-1 TALnet Commands

abort	**hop**	**reboot**	**telnet**
arp	**hostname**	**rename**	**time**
cd	**icmp**	**repeat**	**tip**
close	**iface**	**reset**	**trace**
copy	**ip**	**rip**	**udp**
delete	**kick**	**rmdir**	**uptime**
device	**log**	**route**	**user**
dir	**logout**	**session**	**version**
domain	**memory**	**snmp**	**view**
echo	**mkdir**	**start**	**who**
exit	**ping**	**stop**	
ftp	**process**	**talk**	
help	**pwd**	**tcp**	

A.1 ABORT

NAME

abort - Stops a file transfer within a File Transfer Protocol (FTP) session initiated from your Wireless Router.

SYNOPSIS

abort [*session-index*]

DESCRIPTION

The **abort** command allows you to stop FTP file transfers without closing the FTP session. This command only applies to FTP sessions created using the TALnet **ftp** command.

To use this command, you must switch from the FTP session to the main TALnet session by entering **Ctrl-w**. Identify the index number of the FTP session by entering the **session** command. Then, issue the **abort** command with the appropriate *session-index* argument. If you do not specify a *session-index* argument, the last active session will be used. If this session is not an FTP session, you receive an error message.

When you issue the **abort** command, the active file transfer aborts and you return to the main TALnet prompt (defined by the **hostname** command). You can use **Ctrl-w** to switch back to the FTP session to perform other FTP tasks.

OPTIONS

None.

EXAMPLES

In the following example, the **session** command displays current sessions. The FTP session is session 1. The **abort 1** command cancels any active file transfers in the FTP session.

```
TAL> session
 #  S#  Type      Rcv-Q Snd-Q State        Ed Ec Remote socket
 0  133 Command       0     2 Established   1 1  Telnet CLI (2.3.3.4)
*1  132 FTP           0     0 Established   1 1  ftp (2.3.3.4)

TAL> abort 1
TAL>
```

DIAGNOSTICS

Invalid session.

The session you provided does not exist. Use the **session** command to obtain the correct session index.

Not an active FTP session

The session you specified was not an FTP session.

No active transfer.

No transfer activity is occurring in the FTP session you specified.

SEE ALSO

ftp, **session**

A.2 ARP

NAME

arp - Manages the parameters and tables associated with the Address Resolution Protocol (ARP).

SYNOPSIS

arp [*option*]

DESCRIPTION

The **arp** command allows you to display or manipulate parameters and tables associated with ARP and Reverse ARP (RARP). Both ARP and RARP provide mapping between IP addresses at the network layer and physical addresses at the link layer. This command allows you to force entries into or out of the table, or ARP cache, maintained by these protocols.

When you enter the **arp** command without arguments, it displays the current contents of the ARP cache. For sample output, see the EXAMPLES section.

OPTIONS

add Adds a new entry to the ARP cache.

arp add *hostid* {**ether** | **talk**} *link-layer-addr*

Where:

- *hostid*—Specifies the DNS name or IP address in four-part dotted decimal format.
- **ether**—Specifies the Ethernet link-layer address.
- **talk**—Specifies the TALtalk link-layer address.
- *link-layer-addr*—Specifies the physical address of the interface, and is assigned by the manufacturer.

drop Removes an entry from the ARP cache.

arp drop *hostid* {**ether** | **talk**}

Where:

- *hostid*—Specifies the DNS name or IP address in four-part dotted decimal format.
- **ether**—Specifies the Ethernet ARP cache.
- **talk**—Specifies the TALtalk ARP cache.

flush Removes all dynamic routing entries from the ARP cache.

arp flush

publish Establishes proxy ARP on a router. Adds the entry to the ARP cache and specifies that the router respond to ARP requests.

arp publish *hostid* {**ether** | **talk**} *link-layer-addr*

Where:

- *hostid*—Specifies the DNS name or IP address in four-part dotted decimal format.
- **ether**—Specifies the Ethernet ARP cache.
- **talk**—Specifies the TALtalk ARP cache.
- *link-layer-addr*—Specifies the physical address of the interface, and is assigned by the manufacturer.

reset Resets the status counters shown when you enter the **arp** command with no arguments.

arp reset

reverse Sends out a RARP request to a host on the specified interface.

arp reverse *iface link-layer-addr*

Where:

- *iface*—Is the symbolic name through which the RARP request is sent.
- *link-layer-addr*—Specifies the physical address of the host, and is assigned by the manufacturer.

EXAMPLES

```
TAL> arp
Total number of ARP & RARP packets received...<#>
Incoming requests for unsupported hardware....<#>
Incoming length field(s) didn't match types...<#>
Bad incoming addresses........................<#>
Bad ARP or RARP op code.......................<#>
Incoming ARP requests for us..................<#>
ARP Replies received..........................<#>
Outgoing ARP requests sent....................<#>
Incoming RARP requests for us.................<#>
RARP Replies received.........................<#>
Outgoing RARP requests sent...................<#>
Outgoing RARP request failures................<#>

IP addr          Type           Time Q Addr
-------          ----           ---- - ----
255.255.255.255  10 Mb Ethernet 0      ff:ff:ff:ff:ff:ff
192.168.179.202  TALK           0      41544D31
192.168.180.2    10 Mb Ethernet 1189   08:00:20:20:27:c6
192.168.180.151  10 Mb Ethernet 1182   00:c0:f6:a0:4e:6e
192.168.180.1    10 Mb Ethernet 1189   00:20:af:30:02:9a
192.168.180.154  10 Mb Ethernet 1153   00:c0:f6:90:70:b7
```

Where:

- `Type`—Refers to the type of network connection. Possible types include TALK and 10 Mb Ethernet.
- `Q`—Refers to the position of the ARP request in the queue.
- `Time`—Refers to the time left until the cache entry expires.

DIAGNOSTICS

Unknown host

The IP address could not be resolved.

ERROR: hardware type <*erroneous string*> **NOT supported**

ARP table manipulation must be done on the Ethernet or TALtalk link layers.

ERROR: Must attach device first

ARP table manipulation was attempted before a device of the specified hardware type was configured.

A.3 CD

NAME

cd - Changes the current working directory.

SYNOPSIS

cd [[*drive*]*directory-name* | **..**]

DESCRIPTION

The **cd** command changes your current working directory to the directory specified with the *directory-name* argument. You can specify two dots (**..**) to change to the parent directory of the current working directory, or you can specify a *drive* argument to change logical disk drives. Note, however, that drive changes are not limited to the current working directory. If you change drives, all activities in every open session occur on the new drive.

To display the current directory, enter the **cd** command without any arguments. Specifying the **cd** command without arguments is the same as specifying the **pwd** command.

OPTIONS

None.

EXAMPLES

In the following example, you change to the drive and directory `D:/TAL`. The system verifies that the directory has changed by listing the new working directory.

```
TAL> cd D:/TAL
D:/TAL
```

In the following example, you enter the **cd** command without any arguments. The resulting display indicates that the current working directory is `D:/TAL`.

```
TAL> cd
D:/TAL
```

DIAGNOSTICS

Can't change directory!

You entered an invalid directory name as an argument to the **cd** command.

SEE ALSO

pwd

A.4 CLOSE

NAME

close - Closes a Telnet connection.

SYNOPSIS

close [*session-index*]

DESCRIPTION

The **close** command allows you to close a Telnet connection. To close a Telnet connection, use **Ctrl-w** to switch to the main session on your local Wireless Router. Issue the command **close** to close the connection. Switch back to the Telnet session and press the **Enter** key to close the session.

If you have more than one Telnet connection open on your local Wireless Router, you can issue the *session-index* argument to close a specific connection and session. To determine the session index for the Telnet session you want to close, use the **session** command.

OPTIONS

None.

EXAMPLES

In the following example, the local router's hostname is `TAL>`. The remote router to which you have a Telnet connection is `TAL-REMOTE>`. The key sequence `Ctrl-w` returns you to the local router session. The next line closes the Telnet connection. The second occurrence of `Ctrl-w` returns you to the Telnet session, which you end by pressing the `Enter` key.

```
TAL-REMOTE> Ctrl-w
TAL> close
TAL> Ctrl-w
TAL-REMOTE>
Telnet session 5 closed: Normal
Hit enter to continue: Enter
TAL>
```

SEE ALSO

session, **telnet**

A.5 COPY

NAME

copy - Copies a file.

SYNOPSIS

copy *src-file dst-file*

DESCRIPTION

The **copy** command creates a copy of a source file (specified with the *src-file* argument), and gives it the name specified with the *dst-file* argument. To copy a file from another directory to the current directory, include the directory path in the *src-file* argument. To copy a file from the current directory to another directory, include the directory path in the *dst-file* argument.

⚠ **Caution** TAL routers operate on solid-state disk drives. If you are copying from the C: drive, you must disable the watchdog timer using the **device config** subcommand.

OPTIONS

None.

EXAMPLES

In the following example, you copy the `start.net` file to the `stop.net` file:

```
TAL> copy start.net stop.net
```

In the following example, you copy the `talnet.exe` file from the directory `D:\TAL` to the current directory:

```
TAL> copy d:\tal\talnet.exe talnet.exe
```

DIAGNOSTICS

Usage: copy <srcfile> <dstfile>

You issued the **copy** command but did not use the correct syntax. Check that you included both a source file (srcfile) and a destination file (dstfile) with the command. Also use the **dir** command to ensure that the source file you specified exists in the current directory.

SEE ALSO

device, **dir**, **rename**, **user**

A.6 DELETE

NAME

delete - Deletes a file.

SYNOPSIS

delete *filename* [*filename* ...]

DESCRIPTION

The **delete** command deletes the file specified by the *filename* argument. You can delete more than one file at a time by listing additional files. You can delete a file in another directory by including the directory path as part of the filename. To delete a directory, use the **rmdir** command.

OPTIONS

None.

EXAMPLES

The following example deletes the file `talold.exe`:

```
TAL> delete talold.exe
```

DIAGNOSTICS

Can't delete *string***: No such file or directory**

The file you specified does not exist.

Usage: delete <file>

You did not use the correct syntax. Ensure that you included a *filename* argument.

SEE ALSO

rmdir, **user**

A.7 DEVICE

NAME

device - Configures hardware and link-layer parameters.

SYNOPSIS

device config *device-name* [*global-config***...**]

device kill *device-name*

device make *device-type device-name make-config*

device show {*device-name* | **all**} [**verbose**]

DESCRIPTION

Use the **device** command to create, configure, and remove wire-based hardware interfaces. You must use the **device make** command to identify to the system which hardware interfaces are present, and how they should be used. Assign a symbolic name to each device with this command; you will use these names in other commands to reference hardware devices. The **device** command manipulates interfaces at the hardware level or link layer, whereas the **iface** command manipulates interfaces at the network protocol level.

OPTIONS

config

Use the **device config** command to configure various parameters for a named device. You can control all parameters in one **device config** command, or issue multiple commands for clarity.

device config *device-name* [*global-config***...**]

Where:

- *device-name*—Is a string that is used as a symbolic name for the device; for example, *ether0*.
- *global-config*—Indicates globally configured parameters, as follows. You can include more than one global parameter.

disable	Disables the processing of interrupts for incoming data.
down	Marks the device as down; the device will not transmit or receive data.
enable	Enables the processing of interrupts for incoming data.
up	Marks the device as up; the device can start to transmit or receive data.

speed *bit-rate* Sets the speed at which the device should run. This option is only applicable to **com** devices.

trace *trace-mode* Sets up tracing on the device. This enables the logging of detailed information about packet transfers. The *trace-mode* argument can be one of the following:

- **console**—Specifies that tracing should start and that all trace output messages should be displayed on the console.
- **stop**—Specifies that tracing should stop.
- **start**—Specifies that tracing should start and that all trace output messages should be displayed where standard output is currently directed. This might be the console, or it might be a Telnet session.
- **promiscuous**—Specifies that tracing should enter a mode where every packet received by the device is displayed, whether it is addressed to this system or not. No routing of packets will occur.
- *filename*—Specifies that tracing should start and that all trace output messages should be written to the named file.

trigger *char* Sets the debug/trace trigger character to *char.*

cts Enables hardware flow control. This option is only applicable to **com** devices.

dtroff Deactivates the **dtr** signal. This option is only applicable to **com** devices.

dtron Activates the **dtr** signal. This option is only applicable to **com** devices.

init "*string***"** Sends the specified initialization string to an attached serial device such as a modem whenever the Wireless Router reboots. The argument *string* can be any command that is valid on the serial device, or one of the following special characters:

- **\p**—Send a one-second pause.
- **\r**—Send a carriage return.
- \\—Send a backslash.
- **\"**—Send a quotation mark.

You must issue the **init** keyword before you enable a **com** device with the **enable** keyword. This option is only applicable to **com** devices.

modem Indicates to the system that a modem is attached to this device. This option is only applicable to **com** devices.

noflow Specifies that no flow control is to be used on this device. This option is only applicable to **com** devices.

nomodem Indicates to the system that no modem is attached to this device. This option is only applicable to **com** devices.

reset Resets the counters for the communications control board for the wireless interface.

rtsoff Deactivates the **rts** signal. This option is only applicable to **com** devices.

rtson Activates the **rts** signal. This option is only applicable to **com** devices.

rxfifo *bytes* Sets the size of the transmit/receive buffer, in bytes. The default is 1024. You must disable the device before you modify this option. This option is only applicable to **com** devices.

xon Specifies that software flow control is to be used on this device. This option is only applicable to **com** devices.

kill Removes the named device from the system. The device must be marked as down before you can remove it.

device kill *device-name*

Where:

- *device-name*—Is a string that is used as a symbolic name for the device.

make Adds a new device to the system. Use this command to assign a symbolic name to the device; the name will be used in other configuration commands. You can also perform some preliminary configuration of the device using this command. This release of the TALnet software supports two wire-based device types and one fault recovery device type; therefore, there are three forms of this command.

make com To create a device that provides an asynchronous serial connection to a host or another router running the Point-to-Point Protocol (PPP), use the **device make com** command.

device make com *device-name I/O-base IRQ* [**chain**]

Where:

- *device-name*—Is a string that is used as a symbolic name for the device.
- *I/O-base*—Is a number that tells the system how to address the **com** device. This number must match the jumper and switch settings on the device itself.
- *IRQ*—Is a number that tells the system on which interrupt request line to expect interrupts for the **com** device. This number must match the jumper and switch settings on the device itself.
- **chain**—Is an optional parameter; use it if interrupt/IRQ chaining is required.

make packet To create a device that is generic or provides a connection that might vary, use the **device make packet** command. The software allows you to configure up to three packet devices; however, in this release of the TALnet software, only a Class 1 Ethernet packet device is supported.

device make packet *device-name software-interrupt-number*

Where:

- *device-name*—Is a string that is used as a symbolic name for the device.
- *software-interrupt-number*—Is a number that tells the software how to communicate with the **packet** device. It must match the *software-interrupt-number* used in the packet driver invocation found in the *autoexec.bat* boot script.

make watchdog To create a device that provides a software failure recovery capability, use the **device make watchdog** command. This device starts a timer which will cause a reset if the TALnet software becomes unstable. Note that this timer needs to be refreshed by software at least four times per second. Therefore, if certain file operations are performed that take more than 250 milliseconds (ms), you should disable the watchdog timer prior to this operation. Only one device of type **watchdog** should be created.

device make watchdog *device-name* [*I/O-base*]

Where:

- *device-name*—Is a string that is used as a symbolic name for the device.
- *I/O-base*—Is a number that tells the system how to address the **watchdog** device. This number must be either 0x510 or 0x590, depending on whether jumper E13 is in or out of the motherboard. The default is 0x510.

show Displays the configuration and some statistics of the specified devices. If you specify the **all** keyword instead of a particular device name, status on all configured devices will display. If you specify the **verbose** keyword, more detailed information and statistics will display.

device show {*device-name* | **all**} [**verbose**]

EXAMPLES

The following example creates and configures an asynchronous serial device for PPP connections. The first line creates the device, assigns a symbolic name of `com2`, tells the system to address the device using `0x2F8`, and assigns an interrupt request line number of `3`.

The second line configures the device. The keywords `cts` and `rtson` enable outgoing and incoming hardware flow control, respectively. The keyword `dtron` activates the DTR flag, which tells the modem that the Wireless Router is ready to accept calls. The size of the transmit/receive buffer is set to `1024` bytes. The `modem` keyword indicates that a modem is attached to the device. The speed at which the device should run is set to `9600` baud. Finally, the keyword `up` marks the device as running, and the `enable` keyword enables the processing of interrupts on incoming data.

```
TAL> device make com com2 0x2F8 3
TAL> device config com2 cts rtson dtron rxfifo 1024 modem speed 9600
  up enable
```

The following example configures an Ethernet Class 1 device (`pkt0`). The first line creates the device, assigns the symbolic name of `pkt0`, and specifies a software interrupt number of `0x65`. The second line marks the device as running (`up`), and enables the processing of interrupts on incoming data (`enable`).

```
TAL> device make packet pkt0 0x65
TAL> device config pkt0 up enable
```

The following example configures the watchdog timer that causes a reset if the TALnet software becomes unstable. The first line creates the device and assigns the symbolic name `wdt0`. The second line enables the watchdog timer device.

```
TAL> device make watchdog wdt0
TAL> device config wdt0 enable
```

The following example shows typical output on all configured devices on your Wireless Router:

```
TAL> device show all
Name    Type    Class  Index Flags  Owner    Hardware Address
pkt0    PACKET         0     C007   ether0   00:20:0C:10:22:FF
radio0  SCC            0     C007   radio0
wd0     WATCHDOG       0            8000
```

DIAGNOSTICS

Usage: device config <*name*> **{**<*option*> **{**<*value*>**}}**

Not enough arguments were entered to the **device config** command.

Device *name* **unknown**

The *name* in a **device config** command is not that of a device previously created with a **device make** command.

Unknown option *option***, command aborted**

An invalid argument was entered to the **device config** command.

Command aborted

An invalid argument was entered to the **device config** command.

Option *option* **does not apply to** *name*

The *option* entered on the **device config** *name* command does not apply to a **com** device.

Device *name* **must be disabled to change fifo size**

An attempt was made to change the FIFO size of a **com** device using the **device config** *name* command, but the device was not disabled first.

Usage: device kill *name*

Not enough arguments were entered to the **device kill** command.

Device *name* **does not exist.**

The *name* in a **device kill** command is not that of a device previously created with a **device make** command.

Usage: device make <type> <name>

Not enough arguments were entered to the **device make** command.

Usage:... <I/O base> <IRQ>

Not enough arguments were entered to the **device make com** command.

I/O base *hex number* **already in use**

The I/O base address entered into a **device make com** command is already in use.

Too many 8250 devices

The maximum of six **com** devices has been exceeded by the invocation of a **device make com** command.

Invalid device type *type*

The *type* entered to the **device make** command was not one of **com** or **packet**.

Device *name* **already exists.**

The *name* entered to a **device make** command matches that entered on a previous **device make** command.

Usage: ... <SWInt>

Not enough arguments were entered to the **device make packet** command.

Int *hex number* **already in use.**

The Software Interrupt entered on a **device make packet** command is already in use by some other device.

Too many packet drivers

The maximum of three packet driver devices was exceeded by the invocation of a **device make packet** command.

No packet driver loaded at int *hex number*

After a **device make packet** command was entered, the system attempted to contact the packet driver at software interrupt *hex number*, but none was found.

Unable to obtain driver info

After a **device make packet** command was entered, the system attempted to obtain some basic information from the driver, but none was found.

Unable to access driver

After a **device make packet** command was entered, the system attempted to obtain some basic information from the driver, but none was found.

Unable to get address for device *name*

After a **device make packet** command was entered, the system attempted to obtain the link-layer address from the driver, but none was found.

Warning! Device *name* **has a multicast address: (***multicast address***)**

After a **device make packet** command was entered, the system attempted to obtain the link-layer address from the driver; it was found but it was in a multicast format.

Usage: device show <*dev***>|all [verbose]**

Not enough arguments were entered to the **device show** command.

Device *name* **unknown**

The *name* in a **device show** command is not that of a device previously created with a **device make** command.

Can’t set device (link) address for *name*

An attempt was made to assign a link-layer address to a device that does not support link-layer address assignment.

Unable to parse address *link layer address*

A link-layer address was entered into the **device config** *name* address command, but the format of that address was incorrect.

ERROR: Could not open trace file *file*

The **device config** *name* **trace** *file* command was entered, but the *file* for storing trace results could not be created.

SEE ALSO

iface

A.8 DIR

NAME

dir - Displays the contents of a directory.

SYNOPSIS

dir [*directory-name* | **..**]

DESCRIPTION

The **dir** command displays the contents of the directory specified with the *directory-name* argument. If you do not specify an argument, the command displays the contents of the current directory. If you specify two dots (**..**), the command displays the contents of the parent directory.

OPTIONS

None.

EXAMPLES

The following example displays the contents of the current working directory. The resulting display lists one line for each file or directory. The first character in each line indicates the type of entry; for example, a `d` indicates a directory and a hyphen (`-`) indicates a file. The name of the file or directory appears in the last column. In this example, the current working directory contains two subdirectories (`tal/` and `tmp/`), and one file (`sample`).

```
TAL> dir
drwxr-x---  1 tal     router           0 Dec 21 07:36 tal/
drwxr-x---  1 tal     router           0 Dec 21 07:36 tmp/
-rwxr-x---  1 tal     router          12 Dec 21 12:06 sample

3 files. 768,512 bytes free. Disk size 778,752 bytes.
```

A.9 DOMAIN

NAME

domain - Manages the parameters and tables associated with the Domain Name System (DNS).

SYNOPSIS

domain *option*

DESCRIPTION

The **domain** command manages parameters associated with DNS. DNS provides the mapping between IP addresses and hostnames. This command allows you to force entries into or out of the table maintained by DNS (the DNS server list) as well as execute DNS queries.

OPTIONS

addserver Adds a new name server entry into the DNS server list. Name servers hold a cache, or database, of names mapped to IP addresses.

domain addserver *ip-addr*

Where:

- *ip-addr*—Specifies the IP address in four-part dotted decimal format.

cache clean If you specify an argument, this command sets whether expired records should be discarded or not. If you do not specify an argument, this command displays whether or not expired records are discarded.

domain cache clean [*boolean*]

Where:

- *boolean*—Turns the feature on or off. To turn the feature on, use **y**, **yes**, **true**, **on**, **1**, **set**, or **enable**. To turn the feature off, use **n**, **no**, **false**, **off**, **0**, **clear**, or **disable**.

cache list Displays the DNS cache.

domain cache list

cache load Restores the DNS cache from the specified file. This command will not overwrite current data.

domain cache load *filename*

Where:

- *filename*—Specifies the file from which the DNS cache is restored. The file includes the name and a three-character extension separated by a dot (.).

cache only

If you specify an argument, this command sets “ignore Domain.txt” on or off. If “ignore Domain.txt” is on, DNS name lookups will only go to the cache, and will not go to the DNS cache file *Domain.txt*. If you do not specify an argument, this command indicates the state of “ignore Domain.txt.”

domain cache only [*boolean*]

Where:

- *boolean*—Turns the feature on or off. To turn the feature on, use **y**, **yes**, **true**, **on**, **1**, **set**, or **enable**. To turn the feature off, use **n**, **no**, **false**, **off**, **0**, **clear**, or **disable**.

cache save

Saves the DNS cache to the specified file.

domain cache save *filename*

Where:

- *filename*—Specifies the file to which the DNS cache is saved. The file includes the name and a three-character extension separated by a dot (.).

cache size

If you specify an argument, this command changes the cache memory size to the number of entries provided. If you do not specify an argument, this command displays the memory size for the cache.

domain cache size [*count*]

Where:

- *count*—Specifies the memory size. The memory size can be between 10 and 100 entries. The default is 20 entries.

cache wait

If you specify an argument, this command changes the time between cache file updates to the number provided. If you do not specify an argument, this command displays the time in seconds between cache file updates.

domain cache wait [*update-time*]

Where:

- *update-time*—Specifies the amount of time in seconds between cache file updates. The amount of time can be between 20 seconds and 28800 seconds (8 hours). The default if you do not set this time is 20 seconds.

dropserver

Removes an entry from the DNS server list.

domain dropserver *hostid*

Where:

- *hostid*—Specifies the DNS name or IP address in four-part dotted decimal format.

list Displays the DNS server list.

domain list

query Sends out a query asking for the IP address associated with a DNS name or the DNS name associated with an IP address.

domain query {*ip-addr* | *DNS-name*}

Where:

- *ip-addr*—Specifies in four-part dotted decimal format the IP address for which you are requesting a DNS name.
- *DNS-name*—Specifies the DNS name for which you are requesting an IP address.

retry If you specify an argument, this command resets the number of retries to the number provided. If you do not specify an argument, this command displays the number of retries.

domain retry [*count*]

Where:

- *count*—Is an integer that specifies the number of retries. The retry count can be between 1 and 25. If you do not specify a retry count, the default is 2.

suffix If you specify an argument, this command defines the default domain name appended to hostnames. If you do not specify an argument, it displays the local suffix.

domain suffix [*suffix*]

Where:

- *suffix*—Is the domain name ending with a dot (.).

trace If you specify an argument, this command turns tracing for DNS operations on or off. If you do not specify an argument, this command displays whether or not domain is set to verbose mode.

domain trace [*boolean*]

Where:

- *boolean*—Turns the feature on or off. To turn the feature on, use **y**, **yes**, **true**, **on**, **1**, **set**, or **enable**. To turn the feature off, use **n**, **no**, **false**, **off**, **0**, **clear**, or **disable**.

EXAMPLES

```
TAL> domain list
Server address   srtt mdev   timeout  queries responses
128.105.2.10     1475 1495      7455        9         8
192.168.180.2    1785  429      7002        2         1
128.32.136.253   2000    0      8012        2         0
141.211.180.2    2000    0      4006        1         0
```

Where:

- `Server address`—The IP address of the DNS server
- `srtt`—Smoothed (rounded) round-trip time
- `mdev`—Mean round-trip time deviation
- `timeout`—Number of milliseconds before it last gave up
- `queries`—The number of DNS queries sent to that server
- `responses`—The number of replies to DNS queries from that server

DIAGNOSTICS

Resolver unknown

The DNS server specified could not be found.

<*Domain.Name*> <*number*> **IN {255}**

The domain name indicated by <*Domain.Name*> could not be resolved.

SEE ALSO

ping

A.10 ECHO

NAME

echo - Configures the Telnet client echo negotiation constraints.

SYNOPSIS

echo [**accept** | **refuse**]

DESCRIPTION

The **echo** command accepts or refuses a WILL ECHO offer from remote Telnet clients. In the absence of a negotiated offer, both sides default to no echoing. In this state, no information is sent until a carriage return (<CR>) is sent. When using an editor, however, this can be a problem, so many systems negotiate remote echoing and send each keystroke immediately. This is effective on Ethernet links, but can be inefficient across a radio link.

When you specify the **echo** command without an argument, it displays whether echoing is set to **accept** or **refuse**.

OPTIONS

accept Causes the console to send a DO ECHO if a WILL ECHO is received during Telnet negotiations.

echo accept

refuse Causes the console to send a DON’T ECHO if a WILL ECHO is received during Telnet negotiations.

echo refuse

A.11 EXIT

NAME

exit - Exits from the TALnet software.

SYNOPSIS

exit

DESCRIPTION

The **exit** command allows you to leave your TALnet session and enter the native operating system. After you issue the **exit** command, you are asked to confirm that you want to leave the TALnet session. If you enter **y**, the prompt (`C:\TAL>`) appears but you cannot route traffic through the Wireless Router, and all other Wireless Routers see the router as dead. If you enter **n**, you remain in TALnet. You typically use the **exit** command when you upgrade software or modify the configuration file.

If you exit the TALnet session, you must issue the command **start** to reload the configuration file and reexecute TALnet. (Note that the **start** command you issue to reload the configuration file and reexecute TALnet is not the TALnet **start** command; it is a native operating system command.)

OPTIONS

None.

EXAMPLES

The following example illustrates the process for exiting the TALnet session.

```
TAL> exit
Are you sure (y/n)? y
C:\TAL>
```

SEE ALSO

copy, **logout**, **reboot**

A.12 FTP

NAME

ftp - Makes an FTP connection from your Wireless Router.

SYNOPSIS

ftp *hostid* [*port*]

DESCRIPTION

The **ftp** command allows you to connect to a remote host from your Wireless Router and transfer files to and from that remote host. The *hostid* argument specifies the host to which you want to connect, and must be either the DNS name or the IP address in four-part dotted decimal format. The optional *port* argument is the number of the port through which you want to connect. If you do not specify a port number, the standard FTP control port of 21 is used.

Issuing the **ftp** command opens a new session on your router; this session uses an `ftp>` prompt. After you connect to a remote host using the **ftp** command, you must supply a username; you might also be required to specify a password. To switch from your FTP session to other sessions on the local router, use **Ctrl-w**.

After you open your FTP session, you can use the commands in the FTP SESSION COMMANDS section at the `ftp>` prompt to configure the FTP session or perform FTP operations.

FTP SESSION COMMANDS

ascii Sets the file transfer mode to ASCII. This command and the **type ascii** command behave the same.

batch Turns command batch mode on or off. Command batch mode attempts to optimize communication with the server by only sending when required, not each time a command is entered. For example, the transfer mode only needs to be sent if a file will actually be transferred. If you do not specify an argument, this command shows the current state of command batch mode.

batch [*boolean*]

Where:

- *boolean*—Turns the feature on or off. To turn the feature on, use **y**, **yes**, **true**, **on**, **1**, **set**, or **enable**. To turn the feature off, use **n**, **no**, **false**, **off**, **0**, **clear**, or **disable**.

binary Sets the file transfer mode to network image. This command, the **type binary** command, and the **type image** command behave the same.

cd Changes the current working directory on the remote host to the directory specified with the *remote-dir* argument.

cd *remote-dir*

Where:

- *remote-dir*—Is the name of the directory you want to use as your current working directory on the remote host.

compare Compares the contents of a file on the remote host with a file on the local Wireless Router.

compare *remote-file* [*local-file*]

Where:

- *remote-file*—Is the file on the remote host that you want to compare.
- *local-file*—Is a file on the local Wireless Router to which you want to compare the remote file. Do not specify a second filename if the filenames for both the remote file and the local file are the same.

dir Displays the contents of a directory on the remote host. This command and the **list** command behave the same.

dir [*remote-files* [*local-file*]]

Where:

- *remote-files*—Specifies which files you want to list, and can include the directory name, specific filenames, and wildcards. You can use a dot (.) to specify the current working directory. If you do not specify this argument, the contents of the current working directory on the remote host display.
- *local-file*—Specifies a file to which to write the directory listing. If you do not specify this argument, the contents are displayed on your console. If you specify a *local-file* argument, you must specify a *remote-file* argument.

get Retrieves a copy of the specified file from the remote host and stores it on the local Wireless Router.

get *remote-file* [*local-file*]

Where:

- *remote-file*—Is the name of the file on the remote host that you want to copy.
- *local-file*—Is the name you want to assign the file on the local Wireless Router. If you do not assign this argument, the filename stays the same.

hash Turns on hash mark printing. Hash marks (#) appear on the screen to indicate progress during file transfers; one hash mark appears for each 1000 bytes transferred. Hash mark printing is off by default.

lcd Changes the local current working directory. If you do not specify a directory, this command shows the local current working directory.

lcd [*dir*]

Where:

- *dir*—Is the name of the directory to which you want to switch.

list Displays the contents of a directory on the remote host. This command and the **dir** command behave the same.

list [*remote-files* [*local-file*]]

Where:

- *remote-files*—Specifies which files you want to list, and can include the directory name, specific filenames, and wildcards. You can use a dot (.) to specify the current working directory. If you do not specify this argument, the contents of the current working directory on the remote host displays.
- *local-file*—Specifies a file to which to write the directory listing. If you do not specify this argument, the contents are displayed on your console. If you specify a *local-file* argument, you must specify a *remote-file* argument.

lls Displays a list of files in your current working directory. If you specify an argument, only the files you specify display.

lls [*local-files*]

Where:

- *local-files*—Specifies which files you want to list, and can include the directory name, specific filenames, and wildcards. You can use a dot (.) to specify the current working directory. If you do not specify this argument, the contents of the current working directory on the local Wireless Router display.

ls Displays an abbreviated listing of remote files from the remote host. This command and the **nlst** command behave the same.

ls [*remote-files* [*local-file*]]

Where:

- *remote-files*—Specifies which files you want to list, and can include the directory name, specific filenames, and wildcards. You can use a dot (.) to specify the current working directory. If you do not specify this argument, the contents of the current working directory on the remote host display.
- *local-file*—Specifies a file to which to write the directory listing. If you do not specify this argument, the contents are displayed on your console. If you specify a *local-file* argument, you must specify a *remote-file* argument.

mcompare Compares the contents of multiple files on the remote host with files that have the same filename on the local Wireless Router.

mcompare *remote-file …*

Where:

- *remote-file*—Are the files on the remote host that you want to compare. You can use wildcards to compare multiple files, or specify more than one filename.

md5 Requests the MD5 signature of specified file on the remote host. Note that MD5 might not be available on all remote hosts.

md5 *remote-file*

Where:

- *remote-file*—Is the file on the remote host.

mget Retrieves copies of the specified files from the remote host and stores them on the local Wireless Router.

mget *remote-files ...*

Where:

- *remote-files*—Are the files on the remote host that you want to copy. You can use wildcards to get multiple files, or specify more than one filename.

mkdir Creates a new directory on the remote host.

mkdir *dir-name*

Where:

- *dir-name*—Is the name of the new directory.

mput Stores multiple files from the local Wireless Router on the remote host.

mput *local-files ...*

Where:

- *local-files*—Are the local files you want to move. You can use wildcards to store multiple files, or specify more than one filename.

nlst Displays an abbreviated list of remote files from the remote host. This command and the **ls** command behave the same.

nlst [*remote-files* [*local-file*]**]**

Where:

- *remote-files*—Specifies which files you want to list, and can include the directory name, specific filenames, and wildcards. You can use a dot (.) to specify the current working directory. If you do not specify this argument, the contents of the current working directory on the remote host display.
- *local-file*—Specifies a file to which to write the directory listing. If you do not specify this argument, the contents are displayed on your console. If you specify a *local-file* argument, you must specify a *remote-file* argument.

put Stores a file from the local Wireless Router on the remote host.

put *local-file* [*remote-file*]

Where:

- *local-file*—Is the name of the local file you want to copy.
- *remote-file*—Is the name you want to assign the file on the remote host. If you do not assign this argument, the local filename is used for the remote file.

quit Closes the FTP session on the remote host and exits FTP.

quit

read Displays the contents of the specified remote file.

read *remote-file*

Where:

- *remote-file*—Is the file on the remote host that you want to display.

rmdir Removes the specified remote directory.

rmdir *dir-name*

Where:

- *dir-name*—Is the directory on the remote host.

type Sets the specified file transfer mode. If you do not specify an argument, the current transfer mode displays.

type [**ascii** | **binary** | **image**]

Where:

- **ascii**—Indicates ASCII transfer mode. This command and the **ascii** command behave the same.
- **binary**—Indicates image transfer mode. This command, the **type image** command, and the **binary** command behave the same.
- **image**—Indicates image transfer mode. This command, the **type binary** command, and the **binary** command behave the same.

update Turns on or off MD5 updating. If MD5 updating is enabled, the **mget** and **mput** commands check each target file to see if it already exists and has the same MD5 signature as the source file. If the target file exists and has the same signature, it is not transferred. If you do not specify an argument, this command displays the current state of MD5 updating.

update [*boolean*]

Where:

- *boolean*—Turns the feature on or off. To turn the feature on, use **y**, **yes**, **true**, **on**, **1**, **set**, or **enable**. To turn the feature off, use **n**, **no**, **false**, **off**, **0**, **clear**, or **disable**.

verbose Changes the amount of information that is displayed during different FTP operations. For example, setting the level to 3 turns on hash mark printing. If you set the level to 1 or higher, the MD5 signature appears during an MD5 compare. If you do not specify an argument, the current level displays.

verbose [*level*]

Where:

- *level*—Indicates the verbose level. Level 2 is the default.

EXAMPLES

The following is an example of a typical FTP session. In this session, you log on to the remote host with the IP address of `192.168.2.3`. After the FTP session opens, you set the transfer mode to `binary`, and turn hash mark printing on. You then retrieve a copy of the file `talnet.exe` from the remote host (`192.168.2.3`). Finally, you close the connection by entering `quit`, and are returned to your primary session on your local Wireless Router.

```
TAL> ftp 192.168.2.3
ftp> binary
ftp> hash
ftp> get talnet.exe
ftp> quit
TAL>
```

SEE ALSO

abort, kick, session

A.13 HELP

NAME

help - Displays the list of TALnet commands.

SYNOPSIS

help

DESCRIPTION

The **help** command displays the list of valid TALnet commands. You can also enter a question mark (**?**) to display the list.

OPTIONS

None.

EXAMPLES

The following example illustrates the **help** command and the resulting display of valid TALnet commands:

```
TAL> help
Command list:
                abort           arp             cd
close           copy            delete          device
dir             domain          echo            exit
ftp             help            hop             hostname
icmp            iface           ip              kick
logout          log             memory          mkdir
ping            process         pwd             reboot
rename          repeat          reset           rip
rmdir           route           session         snmp
start           stop            talk            tcp
telnet          time            tip             trace
udp             uptime          users           version
view            who             ?
```

A.14 HOP

NAME

hop - Invokes the traceroute tool.

SYNOPSIS

hop *option*

DESCRIPTION

The **hop** command invokes the traceroute tool, which allows you to see the routes that IP datagrams follow to a given location.

For sample output, see the EXAMPLES section.

OPTIONS

check

Traces the route of an IP datagram sent to a specified IP address or DNS name.

hop check *hostid*

Where:

- *hostid*—Can be either a DNS name or an IP address specified in four-part dotted decimal format.

maxttl

If you specify an argument, this command sets the maximum number of routers to pass through by altering the time-to-live (TTL) field. If you do not specify an argument, this command displays the current value.

hop maxttl [*number*]

Where:

- *number*—Is an integer that specifies the maximum TTL. The maximum TTL can be between 1 and 255. If you do not specify a maximum TTL, the default is 30.

maxwait

If you specify an argument, this command sets the maximum wait for a timeout. If you do not specify an argument, this command displays the current value.

hop maxwait [*number*]

Where:

- *number*—Specifies the maximum amount of time in seconds that the router will wait before timing out. The maximum wait can be between 1 and 3600 seconds. If you do not specify a maximum wait, the default is 6 seconds.

queries If you specify an argument, this command sets the number of queries sent to each router. If you do not specify an argument, this command displays the current value.

hop queries [*number*]

Where:

- *number*—Is an integer that specifies the number of queries. This number can be between 1 and 5. If you do not specify the number, the default is 3.

trace If you specify an argument, this command sets tracing of hop checks on or off. In other words, it logs every message that comes back as it comes back. If the general TALnet log is on and hop-check tracing is on, hop checks are recorded to a file. If you do not specify an argument, this command displays the status of hop-check tracing.

hop trace [*boolean*]

Where:

- *boolean*—Turns the feature on or off. To turn the feature on, use **y**, **yes**, **true**, **on**, **1**, **set**, or **enable**. To turn the feature off, use **n**, **no**, **false**, **off**, **0**, **clear**, or **disable**.

EXAMPLES

```
TAL> hop check agate.berkeley.edu
Resolving agate.berkeley.edu... traceroute to 128.32.155.1:33434
 1: 192.168.180.152 wayne.pa.tetherless.com. (1 ms) (1 ms) (2 ms)
 2: 192.168.180.1   gatekeeper.pa.tertherless.com. (3 ms) (3 ms) (5 ms)
 3: 192.168.181.1   gateway.pa.tetherless.net. (6 ms) (4 ms) (4 ms)
 4: 104.174.168.6   ln1-tlg-tetherless.tlg.net. (8 ms) (7 ms) (7 ms)
 5: 104.174.4.10    gw1-mv-tlg.tlg.net. (48 ms) (21 ms) (17 ms)
 6: 192.16832.45    border1-serial13-0.SanFrancisco.mci.net. (19 ms)
                    (19 ms) (15 ms)
 7: 192.168.32.6    barrnet.SanFrancisco.mci.net. (16 ms) (22 ms)(20 ms)
 8: 192.31.48.202   su-c.barrnet.net. (21 ms) (20 ms) (19 ms)
 9: 131.119.2.10    ucbnex.barrnet.net. (17 ms) (17 ms) (17 ms)
10: 198.128.16.21   inr-666-dmz.berkeley.edu (33 ms) (30 ms) (19 ms)
11: 128.32.1.2      inr-108-styx.Berkeley.EDU (24 ms) (19 ms) (18 ms)
12: 128.32.155.1    agate.berkely.edu (23 ms) (21 ms) (21 ms)
traceroute done: normal (Unreachable Port)
```

Where:

- The first line resolves the IP address for the DNS name.
- Subsequent lines list the TTL, the IP address, and DNS name for the host or router, then three round-trip times from the sending host to that host or router. Note that this is not a per-hop round-trip time.

SEE ALSO

icmp, **log**

A.15 HOSTNAME

NAME

hostname - Sets the hostname of the TALnet router.

SYNOPSIS

hostname {*name* | *iface*}

DESCRIPTION

Sets the hostname of the TALnet router. The hostname is used in system prompts, and should be the same as the DNS name for that router. If you specify an interface name, a reverse DNS lookup is executed on the interface IP address.

OPTIONS

name Sets the system name and changes the prompt to reflect that name.

hostname *name*

Where:

- *name*—Is the hostname, and can contain uppercase and lowercase letters, numbers, and hyphens, but cannot contain spaces. The hostname is case sensitive and should not begin with a number.

iface Sets the system name based on the reverse DNS lookup on the specified interface's IP address. You can discover and modify the interface IP address using the **iface show** *iface* command, or you can discover possible interfaces using the **iface show all** command. The **hostname** *iface* command sets the prompt based on the results of the DNS hostname lookup.

hostname *iface*

Where:

- *iface*—Is the symbolic name for the interface on which you want to do the reverse DNS lookup.

EXAMPLES

In the following example, the hostname is set to `linda`.

```
TAL> hostname linda
linda>
```

In the following example, the hostname is the DNS name for Ethernet interface 0 (`ether0`). The DNS name for that interface is erin.pa.tetherless.com, so the new hostname is `erin`.

```
linda> hostname ether0
Hostname set to erin
erin>
```

SEE ALSO

domain, **iface**

A.16 ICMP

NAME

icmp - Manages the tables and parameters associated with the Internet Control Message Protocol (ICMP).

SYNOPSIS

icmp *option*

DESCRIPTION

The **icmp** command displays or manipulates parameters associated with ICMP. ICMP is a network-layer protocol that sends reports of problems about datagrams back to the source that sent the datagram. This command allows you to set parameters associated with ICMP or to view the status of messages received.

OPTIONS

echo

If you specify an argument, this command sets echoing of ICMP messages to the console on or off. The default is to respond to ICMP echo requests. If you do not specify an argument, this command displays the current setting.

icmp echo [*boolean*]

Where:

- *boolean*—Turns the feature on or off. To turn the feature on, use **y**, **yes**, **true**, **on**, **1**, **set**, or **enable**. To turn the feature off, use **n**, **no**, **false**, **off**, **0**, **clear**, or **disable**.

status

Displays the status of ICMP. For sample output, see the EXAMPLES section.

icmp status

trace

If you specify an argument, this command turns the echoing of *Port Unreachable*, *Host Unreachable*, *Source Quench*, or *Time-to-Live Exceeded* messages on or off. If you do not specify an argument, this command displays the current setting.

icmp trace [*boolean*]

Where:

- *boolean*—Turns the feature on or off. To turn the feature on, use **y**, **yes**, **true**, **on**, **1**, **set**, or **enable**. To turn the feature off, use **n**, **no**, **false**, **off**, **0**, **clear**, or **disable**.

EXAMPLES

```
TAL> icmp status
ICMP Statistics:
General:           25 in, 25 out, 0 errors
Echos:              0 in, 25 out, 25 replies to us, 0 replies from us
Timestamps:         0 in, 0 out, 0 replies to us, 0 replies from us
Address Masks:      0 in, 0 out, 0 replies to us, 0 replies from us
Time Exceeded:      0 in, 0 out
Parameter Problems: 0 in, 0 out
Source Quench:      0 in, 0 out
Redirects:          0 in, 0 out
```

SEE ALSO

hop, **ping**

A.17 IFACE

NAME

iface - Configures network interface parameters.

SYNOPSIS

iface config *iface-name* [*global-config...*] [*encap-specific...*]

iface create *iface-name* [*global-config...*] [*encap-specific...*]

iface delete *iface-name*

iface kill *iface-name*

iface make *iface-name* [*global-config...*] [*encap-specific...*]

iface show {*iface-name* | **all**} [**verbose**]

DESCRIPTION

Use the **iface** command to create, configure, and remove network interfaces and parameters on the system. You must use the **iface make** command to create all wire-based interfaces and to assign the interfaces to devices. If the default address for the system is not suitable for a specific interface, you must specify an interface-specific IP address. You must also set the subnet mask for all interfaces that are not point-to-point.

OPTIONS

config Use the **iface config** command to configure various parameters for a named interface.

iface config *iface-name* [*global-config...*] [*encap-specific...*]

Where:

- *iface-name*—Specifies the interface you are configuring, and should match the symbolic name assigned with the **iface create** or **iface make** command.
- *global-config*—Indicates globally configured parameters, as follows. You can issue more than one global parameter for each command.

address *ip-addr* Specifies the IP address to be used for the named interface. The address should be given in four-part dotted decimal format; for example, 192.168.185.66.

broadcast *ip-addr* Specifies the broadcast address for the interface. For point-to-point links, this should be undefined. For other media such as Ethernet, this should be set to the local network address with the host portion as all ones; for example, 192.168.185.79.

demand

Marks the interface for demand use. Normally used with dial-up connections running SLIP or PPP. If you configure a demand interface, make sure that the interface is associated with a device.

device *device-name*

Associates the interface with an existing named device. You must associate an Ethernet interface with a packet driver device before you enable the interface. If you are configuring a dedicated PPP interface, you must associate that interface with a communications device. The *device-name* argument is the symbolic name assigned with the **device make** command.

down

Marks the interface as being down and stops routing traffic on the interface.

encapsulation *encap-name*

Sets up what type of encapsulation the interface is to be used with. You must specify this argument pair with the **iface make** or **iface create** subcommands. The *encap-name* argument can be any of the following:

- **arcnet—**Specifies the standard ARCnet encapsulation method for IP datagrams on ARCnet devices.
- **ethernet—**Specifies standard IP-over-Ethernet encapsulation.
- **ppp**—Specifies Point-to-Point Protocol (PPP) encapsulation on asynchronous or synchronous serial lines. This is the preferred encapsulation method for serial lines.
- **raw ip**—For internal use on loopback and null interfaces only.
- **slfp**—Specifies Serial Line Framing Protocol (SLFP) encapsulation, an outdated method used on some asynchronous links. In general, you should use PPP encapsulation instead of SLFP encapsulation.
- **slip**—Specifies Serial Line Internet Protocol (SLIP) encapsulation on asynchronous serial lines. In general, you should use PPP encapsulation instead of SLIP encapsulation.

forward *iface-name* — Sets up the interface on which to forward all outgoing traffic to another specified interface. The *iface-name* argument specifies the interface on which you are forwarding traffic, and should match the symbolic name assigned with the **iface create** or **iface make** command.

idle — Forces a demand interface to close down the link-layer protocol.

kick — Forces a demand interface to start up the link-layer protocol.

mru *size* — Sets the maximum size of datagrams that can be received on the interface. This is the size without any link-layer encapsulation.

mtu *size* — Sets the maximum size of datagrams that can be transmitted on the interface. This is the size without any link-layer encapsulation.

mxu *size* — Sets the maximum size of datagrams that can be received and transmitted on the interface. This is the size without any link-layer encapsulation.

netmask *ip-addr* — Specifies the subnet mask in decimal address format for the interface. This is the mask specified by the address class as modified by any subnet mask. For point-to-point links, this should be left undefined, but for other media such as Ethernet, this should be set. A sample value is 255.255.255.240.

timeout *timeout-value* — Sets the idle timeout for the interface. Used by demand devices to close down the connection when the interface is idle.

unlink — Removes the binding from an interface to a device. The interface must be marked as down.

up — Marks the interface as up and starts the link-layer protocol.

- *encap-specific*—Indicates parameters that you can configure only if you use SLIP encapsulation, as follows:

vj {**on** | **off** | **auto**} — Indicates whether Van Jacobsen header compression should be on, off, or automatically enabled. If Van Jacobsen header compression is automatically enabled, it will start out disabled, but switch to enabled if a compressed datagram is received. You can only configure this parameter if you have specified SLIP encapsulation.

csid — Compresses the connection ID. You can only specify **csid** if you have specified SLIP encapsulation and Van Jacobsen header compression is enabled.

create — Adds a new interface to the system. This command and the **iface make** command behave the same.

iface create *iface-name* [*global-config...*] [*encap-specific...*]

Where:

- *iface-name*—Assigns a symbolic name to the interface. This name can include uppercase and lowercase letters, hyphens, and numbers, and should indicate the type of interface and the unit number; for example, *ether0*.
- *global-config*—All options listed for the **iface config** command apply.
- *encap-specific*—All options listed for the **iface config** command apply.

delete — Removes the named interface from the configuration. The interface must be marked as down before this can be done. This command and the **iface kill** command behave the same.

iface delete *iface-name*

Where:

- *iface-name*—Specifies the interface you are deleting, and should match the symbolic name assigned with the **iface create** or **iface make** command.

kill — Removes the named interface from the configuration. The interface must be marked as down before this can be done. This command and the **iface delete** command behave the same.

iface kill *iface-name*

Where:

- *iface-name*—Specifies the interface you are deleting, and should match the symbolic name assigned with the **iface create** or **iface make** command.

make — Adds a new interface to the system. This command and the **iface create** command behave the same.

iface make *iface-name* [*global-config...*] [*encap-specific...*]

Where:

- *iface-name*—Assigns a symbolic name to the interface. This name can include uppercase and lowercase letters, hyphens, and numbers, and should indicate the type of interface and the unit number; for example, *ether0*.
- *global-config*—All options listed for the **iface config** command apply.
- *encap-specific*—All options listed for the **iface config** command apply.

show Shows the configuration and some statistics of the specified interfaces.

iface show {*iface-name* | **all**} [**verbose**]

Where:

- *iface-name*—Is the symbolic name for a specific interface.
- **all**—Requests status on all configured interfaces.
- **verbose**—Requests more-detailed information and statistics.

EXAMPLES

The following example creates the dedicated PPP interface `ppp0` with an IP address of `192.168.179.241`. The interface is associated with the device `com2`, the maximum size of datagrams that can cross this interface is `1500`, and the interface uses PPP encapsulation.

```
TAL> iface make ppp0 device com2 address 192.168.179.241 mxu 1500
  encapsulation ppp up
```

The following example creates an Ethernet interface (`ether0`). The first line creates the interface, associates it with the device `pkt0`, specifies an IP address of `192.168.179.241`, and defines Ethernet encapsulation. The second line creates a subnet mask address of `255.255.255.248`, a broadcast address of `192.168.179.24`, and specifies that datagrams no larger than `1500` bytes can cross the interface. The up keyword enables the interface.

```
TAL> iface make ether0 device pkt0 address 192.168.179.241
  encapsulation ether

TAL> iface config ether0 netmask 255.255.255.248 mxu 1500
  broadcast 192.168.179.24 up
```

The following example shows typical output on all configured interfaces on your Wireless Router:

```
TAL> iface show all
ether0 Attached to ether0, address: 192.168.180.249 192.168.179.201
       MXU 1500, link encapsulation: Ethernet
       flags 810A trace 0 netmask 0xFFFFFF00 broadcast 192.168.180.255
loopback  Not attached to any interface, address: 127.0.0.1
       MXU 1500, link encapsulation: Raw IP
       flags 8002 trace 0 netmask 0xFF000000 broadcast 255.255.255.255
null   Not attached to any interface, address: 0.0.0.0
       MXU 1500, link encapsulation: Raw IP
       flags 8002 trace 0 netmask 0xFFFFFFFF broadcast 255.255.255.255
radio0 Attached to radio0, address: 192.168.179.201
       MXU 1500, link encapsulation: TALK
       flags 800A trace 0 netmask 0xFFFFFFFF broadcast 255.255.255.255
tunnel Not attached to any interface, address: 0.0.0.0
       MXU 1500, link encapsulation: Raw IP
       flags 8002 trace 0 netmask 0xFFFFFFFF broadcast 255.255.255.255
```

DIAGNOSTICS

Interface *iface* **unknown**

An attempt was made to configure a nonexistent interface.

Interface *iface* **does not exist**

An attempt was made to remove a nonexistent interface.

Interface *iface* **already exists**

An attempt was made to create a new interface with the same name as an already-existing interface.

Iface *iface* **unknown**

An attempt was made to show the configuration of a nonexistent interface.

Unknown option *option***, command aborted**

An unknown configuration command was given.

Failed to add IP address onto *iface*

The router was unable to assign the specified IP address to the named interface.

Interface *iface* **already marked demand**

An attempt was made to mark an interface as demand when this had already been done.

Interface *iface* **already linked to** *device*

An attempt was made to link an interface to a device when the interface has already been linked to a device.

Device *device* **does not exist**

An attempt was made to link an interface to a nonexistent device.

Device *device* **is busy**

An attempt was made to link an interface to a device already in use by another interface.

Interface *iface* **already marked down**

An attempt was made to shut down an interface when it was already shut down.

Interface *iface* **must be down to change encapsulation**

An attempt was made to change the encapsulation of an interface when the interface might be in use. Use the **iface config** *iface-name* **down** command to mark the devices as down.

Unknown encapsulation *encap*

An attempt was made to use an unknown encapsulation type.

Interface *iface* **is already up and ready**

An attempt was made to use the **kick** subcommand on an interface that is already up and running.

Interface *iface* **is already idle**

An attempt was made to use the **idle** subcommand on a demand interface that is already idle.

Interface *iface* **is not configured for demand operation**

An attempt was made to apply the **kick** or **idle** subcommand to an interface that is not configured for demand mode.

Interface *iface* **must be down to unlink**

An attempt was made to unlink an interface from its device while the interface was running. Use the **iface config** *iface-name* **down** command to mark the devices as down.

Interface *iface* **is already marked up**

An attempt to start an already-running interface was made.

Interface *iface* **has no device**

An attempt to start an interface not bound to any device was made.

Interface *iface* **has no encapsulation**

An attempt to start an interface without any link-layer encapsulation was made.

SEE ALSO

device, **talk**

A.18 IP

NAME

ip - Manages the parameters associated with the Internet Protocol (IP).

SYNOPSIS

ip *option*

DESCRIPTION

The **ip** command allows you to configure IP options. IP is a network-layer protocol that breaks data from higher-level protocols such as TCP and UDP into datagrams. IP provides an addressing space and other controls to allow the datagrams to be routed. If IP datagrams are too large, they are fragmented into multiple smaller datagrams so that they can be delivered. Fragmented datagrams can arrive in any order and IP will reassemble them. If some of the fragments are lost, IP throws the rest of the fragments away and requests a retransmission. Because lost datagrams are lost without notification, the only way for the router to know when to ask for a retransmission is through a timer, called a retransmission timer.

OPTIONS

address

If you specify an argument, this command sets the global node address. If you do not specify an argument, this command displays the current value.

ip address [*ip-addr*]

Where:

- *ip-addr*—Specifies the IP address, expressed in four-part dotted decimal format.

filter

Allows you to configure a packet filter, also known as a firewall. A packet filter is a sequential set of permit and deny conditions that apply to addresses. Packet filtering includes an implicit deny condition; if a packet does not match any of the conditions specified, it is implicitly denied. You can use variants of the **ip filter** command to do the following:

To delete the filters associated with the listed interface, use the following command syntax:

ip filter *iface-name* **delete**

To list the filters for a particular interface, use the following command syntax:

ip filter *iface-name* **list**

To set up the filter, use the following command syntax. Note that, once you apply a filter to an interface, the router assumes an implicit deny condition for any undefined condition. You must specify whether to deny or permit traffic on the specified interface, and whether you are restricting incoming or outgoing traffic. Acceptable values for *type*, *src*, and *dst* follow the command.

ip filter *iface-name* {**deny** | **permit**} {**in** | **out**} *type src dst*

The *type* argument sets the kind of traffic the filter is regulating. You can use any of the following:

- *****—All data packets
- **icmp**—All ICMP datagrams
- **icmprd**—ICMP redirect datagrams
- **icmpxrd**—All ICMP datagrams except redirect datagrams
- **tcp**—All TCP packets
- **tcpsyn**—TCP SYNs
- **tcpxsyn**—All TCP packets except TCP SYNs
- **udp**—All UDP datagrams

The *src* argument filters packets based on their source, or where they originated from.The *dst* argument filters packets based on their destination. You can use any of the following arguments to filter based on source or destination:

- *—Applies the filter to all addresses.
- *ip-addr*—Applies the filter to the specified address.
- **!** *ip-addr*—Applies the filter to all but the specified address. If this is used with a deny condition, all addresses but the specified address are denied. If this is used with a permit condition, all addresses but the specified address are permitted.
- *ip-addr*/*bits*—Applies the filter to all addresses in which the specified number of bits matches the bits in the specified address.
- **:** *loport* {+|– *hiport*}—Applies the filter to the specified TCP or UDP port or port range. The plus sign (+) specifies any port number greater than or equal to the one entered in the previous field. The minus sign (–) specifies a range of port numbers, and must be followed by the high port number. This is useful for denying only certain connections; for example, Telnet, or FTP sessions. You can also specify this with an IP address.

pool Sets up a pool of IP addresses for dynamic assignment. If you specify the **pool** command without an argument, it displays current settings. For sample output, see the EXAMPLES section.

ip pool [*pool-base pool-count*]

Where:

- *pool-base*—Specifies the lowest IP address in the pool.
- *pool-count*—Specifies the number of IP addresses to be pooled.

rtimer This command allows you to set the IP retransmission timer in seconds. If you do not specify an argument, this command displays the current value.

rtimer [*secs*]

Where:

- *secs*—Indicates the number of seconds before data is retransmitted. If you do not set the retransmission timer, the default is 30 seconds.

status Outputs statistics about IP.

ip status

ttl If you specify an argument, this command sets the maximum time to live for locally created IP datagrams. If you do not specify an argument, this command displays the current value.

ip ttl [*count*]

Where:

- *count*—Indicates the maximum number of hops. The default is set to 255, the Internet-recommended value.

EXAMPLES

```
TAL> ip pool
None

TAL> ip pool 192.168.184.11 25
TAL> ip pool
192.168.184.11 thru 192.168.184.35
```

SEE ALSO

icmp, **iface**, **tcp**, **udp**

A.19 KICK

NAME

kick - Restarts a stalled Telnet or File Transfer Protocol (FTP) session on your Wireless Router.

SYNOPSIS

kick [*session-index*]

DESCRIPTION

The **kick** command allows you to restart a Telnet or FTP session if the session stalls after encountering a congested or failing network. The **kick** command forces the TCP/IP protocol to retransmit data that might have been lost in the network. This command only applies to Telnet or FTP sessions created using the TALnet **telnet** or **ftp** commands.

To use this command, you must switch to the main TALnet session by entering **Ctrl-w**. Identify the index number of the stalled Telnet or FTP session by entering the **session** command. Then, issue the **kick** command with the appropriate *session-index* argument. If you do not provide a *session-index* argument, any network activity in the last used session will be restarted. To return to the restarted session, enter **Ctrl-w** again.

When a network is congested or failing, the TCP/IP protocol slows down attempts to retransmit lost data. After many failures, the retransmission attempts can be spaced widely, with several minutes between each. Use the **kick** command to wake up any Telnet or FTP sessions when network connectivity returns or becomes less congested.

OPTIONS

None.

EXAMPLES

In the following example, the **session** command displays current sessions. The **kick 1** command restarts session 1, which is an FTP session.

```
TAL> session
 #  S#  Type      Rcv-Q Snd-Q State        Ed Ec Remote socket
 0  133 Command       0     2 Established   1 1  Telnet CLI (2.3.3.4)
*1  132 FTP           0     0 Established   1 1  ftp (2.3.3.4)

TAL> kick 1
TAL>
```

DIAGNOSTICS

Invalid session.

The session you provided does not exist. Use the **session** command to obtain the proper session index.

SEE ALSO

ftp, **session**, **telnet**

A.20 LOG

NAME

log - Manages output of system debugging and logging information.

SYNOPSIS

log [[**-f**] **console** | [**-f**] **session** | **stop** | *filename*]

DESCRIPTION

The **log** command allows you to specify the destination for debugging and logging information. You can display this information directly on the service console or in a separate console session. You can also write the information to a file for later analysis.

When you enter the **log** command without arguments, it displays the current destination of all system debugging and logging information.

OPTIONS

-f Copies all debugging and logging information to a separate file (*console.log* when used with the **console** subcommand or *session.log* when used with the **session** subcommand). The information also displays on your console or in a separate console session.

console Displays all debugging and logging information on the service console. The information is displayed as soon as possible, usually interspersed with the output of user commands already in progress.

session Displays all debugging and logging information in a separate console session. To view the information, switch to the debugging session by typing **Ctrl-w** at the service console.

stop Stops all logging activity and closes the log file and any open logging sessions.

filename Sends all debugging and logging information to the specified file.

EXAMPLES

The following example opens a separate session on the service console that contains just debugging and logging information. The data is also copied to a file named *session.log*.

```
TAL> log -f session
```

The following example shows that you have configured the Wireless Router to display system debugging and logging information in a separate session and to write the information to a file called *session.log*.

```
TAL> log
Logging to session
Writing to session.log
```

The following example stops all logging activity and closes the log file and any open log sessions:

```
TAL> log stop
```

SEE ALSO

session

A.21 LOGOUT

NAME

logout - Logs you off of the Wireless Router.

SYNOPSIS

logout

DESCRIPTION

The **logout** command allows you to log off the TAL Wireless Router. The **logout** command behaves differently depending on the type of connection you have:

- If you are directly connected to the Wireless Router through a service console, a login prompt and login banner appear on the service console.
- If you are connected through a Telnet connection, the connection closes.

Log out of the Wireless Router whenever you have completed your activities on that router. If you are logged in through a service console and remain logged in, other users can access TALnet over a modem without logging in. Likewise, if you are logged in through a Telnet connection and that connection is dropped before you log out, other users can access TALnet without logging in.

OPTIONS

None.

EXAMPLES

In the following example, the user logs out of a service console connection:

```
TAL> logout

TALnet Release 1.2
Copyright 1992-1996 Tetherless Access Limited, All Rights Reserved.
Compiled at TAL: Feb 23 1996, 12:23:05
Router platform: SubSpace 2001 (rev1)
Memory Usage:    Library=41K, Code=389K, Data=36K

user:
```

SEE ALSO

exit

A.22 MEMORY

NAME

memory - Configures and controls memory allocation.

SYNOPSIS

memory [*option*]

DESCRIPTION

The **memory** command displays or configures parameters and tables associated with memory allocation.   Memory allocation is used to spawn processes and make connections as well as keep the TALnet software alive and running. This command allows you to control memory usage and buffer numbers and sizes.

When you enter the **memory** command without arguments, it displays memory statistics. For sample output, see the EXAMPLES section.

OPTIONS

freelist

Displays the storage allocator freelist. Each entry consists of a statting segment in hexadecimal and a size in decimal. For sample output, see the EXAMPLES section.

memory freelist

ibufsize

Sets the interrupt buffer size to the number of bytes specified. The size should be set to the size of the largest type of buffer plus a header size of eight. If you do not specify an argument, this command displays the size of the interrupt buffer.

memory ibufsize [*bytes*]

Where:

- *bytes*—Specifies the size of the buffer. If you do not set the interrupt buffer size, the default is 1792.

nibufs

Sets the number of buffers available for the interrupt buffer pool. If you do not specify an argument, this command displays the current value. A minimum of two free buffers should be maintained. This information can be found by outputting memory status.

memory nibufs *count*

Where:

- *count*—Specifies the number of available buffers. If you do not specify a number of buffers, the default is 5.

sizes Displays a history of requested storage allocator sizes.

memory sizes

thresh Sets the memory threshold size in bytes. If the amount of available memory passes below this number, no more connections can be opened or accepted. If you do not specify an argument, this command displays the current value.

memory thresh [*bytes*]

Where:

- *bytes*—Specifies the memory threshold size. If you do not set the memory threshold size, the default is 8192.

EXAMPLES

The following sample output shows the storage allocator freelist:

```
TAL> memory freelist
86920000      16 | 87260000      16 | 87ba0000      16 | 888f0000      16
89130000      16 | 89370000      16 | 98270008      16 | 8a180000      80
8b1f0000      16 | 8b350000      52 | 8bcc0008    1360 | 8c5d0008     264
8d200008    1064 | 8f640000      68 | 91650000    1960 | 6fed96f8    8208
```

The following sample output shows memory statistics for your Wireless Router:

```
TAL> memory
Memory: 111312 bytes free (of 235456 total), 124144 used,
        24636 requests, 24357 frees, 279 outstanding, 0 failures,
        0 severe shortages, 0 minor shortages,
        0 invalid requests.

Mbufs:  21910 requests, 21869 frees, 41 outstanding, 0 failures,
        15458 grow requests (4523 required additional memory.)

Ibufs:  115699 requests, 115698 frees, 1 outstanding, 0 failures,
        1980 recovered, 0 destroyed,
        39 preallocated: 5 to host, 34 to interfaces.
```

A.23 MKDIR

NAME

mkdir - Creates a directory.

SYNOPSIS

mkdir *directory-name*

DESCRIPTION

The **mkdir** command creates, or makes, a new subdirectory within the current working directory. You can also create a directory within another directory by specifying the path name as part of the directory name. To remove a directory, use the **rmdir** command. To list the contents of a directory, use the **dir** command.

OPTIONS

None.

EXAMPLES

The following example creates the directory `temp`:

```
TAL> mkdir temp
```

DIAGNOSTICS

Usage: mkdir <directory>

You issued the **mkdir** command, but did not specify a directory name.

SEE ALSO

dir, **rmdir**

A.24 PING

NAME

ping - Determines if a node is running or reachable.

SYNOPSIS

ping *hostid* [*length* [*interval* [*incflag*]]]

DESCRIPTION

The **ping** command uses the Packet Internet Groper (ping) facility to determine if a computer is alive or reachable by sending ICMP *Echo Request* datagrams. If the remote computer is reachable, it returns an ICMP *Echo Response* datagram. When using **ping** for problem isolation, first **ping** the local host to make sure the local network interface is responding.

All variables are reset after **ping** is executed. That is, if you enter in any values for a **ping**, the values will be erased before you execute the next **ping**.

OPTIONS

hostid

When entered with only the host ID, the **ping** command returns an IP address and delay time. For sample output, see the EXAMPLES section.

ping *hostid*

Where:

- *hostid*—Indicates the system you are querying, and can be either a DNS name or an IP address expressed in four-part dotted decimal format.

length

An ICMP *Echo Request* datagram (a ping) contains IP and ICMP headers and an 8-byte time stamp. It also contains an arbitrary number of pad bytes to fill out the datagram. This command allows you to control the number of pad bytes by specifying the number of bytes in the data packet being built.

ping *hostid length*

Where:

- *hostid*—Indicates the system you are querying, and can be either a DNS name or an IP address expressed in four-part dotted decimal format.
- *length*—Specifies the size of the data packet in bytes. The default is 16 bytes.

interval

If you set an interval, multiple pings are transmitted and statistics are reported. This argument specifies the time to pause between transmissions.   If the interval is too small, ICMP *Source Quench* errors might be generated or the datagrams might be silently discarded. This command allows you to have control over the transmission rate by specifying the pause (in milliseconds) between transmissions. Use the **reset** command to stop issuing pings. For sample output, see the EXAMPLES section.

ping *hostid length interval*

Where:

- *hostid*—Indicates the system you are querying, and can be either a DNS name or an IP address expressed in four-part dotted decimal format.
- *length*—Specifies the size of the data packet in bytes. The default is 16 bytes.
- *interval*—Indicates the pause in milliseconds between transmissions.

incflag

This version of the command allows you to set the increment flag, which increments the IP number with each transmission. Use this command sparingly. It is useful for finding all the computers within a range of IP numbers. You can stop this process using the **reset** command. For sample output, see the EXAMPLES section.

ping *hostid length interval incflag*

Where:

- *hostid*—Indicates the system you are querying, and can be either a DNS name or an IP address expressed in four-part dotted decimal format.
- *length*—Specifies the size of the data packet in bytes. The default is 16 bytes.
- *interval*—Indicates the pause in milliseconds between transmissions.
- *incflag*—Indicates the amount to increment the IP address with each transmission.

EXAMPLES

The following example sends a ping to the host `barbar`. The information that displays as a result of the ping indicates that host barbar is at IP address `192.168.180.2`, and that the round-trip time was `19` milliseconds.

```
TAL> ping barbar
192.168.180.2 (19 ms)
```

The following example sends repeated pings to host nuhoa.aloha.net. The IP address for the host is 192.168.112.34. The pings are 100 milliseconds apart and contain 100 bytes of data. In the resulting display, rtt indicates the round-trip time, and mdev indicates the mean round-trip time deviation.

```
TAL> ping nuhou.aloha.net 100 100
Resolving nuhou.aloha.net... Pinging nuhou.aloha.net (192.168.112.34);
data 100 interval 100 ms:
sent   rcvd    %       rtt   avg rtt      mdev
1         1  100       110       110         0
2         2  100        55       103        14
3         3  100        55        97        23
4         4  100       110        99        21
5         5  100        55        94        27
6         6  100       110        96        24
8         7   88       110        98        22
```

The following example sends an initial ping to host theb.org. The IP address for this host is 192.168.129.130. The command specifies to send subsequent pings to incremental IP addresses. Because the flag is set at 1, the IP addresses increment by 1. The resulting output lists the IP addresses you ping, the corresponding DNS names, and the round-trip time for the ping.

```
TAL> ping theb.org 100 100 1
Resolving theb.org... Pinging theb.org (192.168.129.130); data 100
interval 100 ms:
192.168.129.130  theb.org. (110 ms)
192.168.129.131  taz.live105.com. (55 ms)
192.168.129.132  taz.apache.org. (330 ms)
192.168.129.133  cinagro.organic.com. (55 ms)
192.168.129.134  (55 ms)
```

SEE ALSO

icmp, **reset**

A.25 PROCESS

NAME

process - Determines the status of TALnet operating system software tasks, called processes.

SYNOPSIS

```
process [option]
```

DESCRIPTION

The **process** command allows you to view the state of processes (or tasks) running as part of the TALnet operating system software. The TALnet software switches between processes so quickly that all processes appear to run simultaneously.

If you do not specify an argument, the **process** command lists all currently active processes within the TALnet operating system software.

OPTIONS

check Allows the TALnet software to ensure that enough system memory is set aside to perform each process. Note, however, that enabling process checking might adversely affect system performance.

```
process check [on | off]
```

Where:

- **on**—Enables process checking.
- **off**—Disables process checking.

status Shows the current status of all processes within the TALnet software.

```
process status
```

trace Allows you to monitor changes in the state of all process activity within the TALnet software. When enabled, the state of all processes (including task creation, sleep, wakeup, suspension, and deletion) is sent to the system log. See the **log** command description for more information on setting up and viewing the system log.

```
process trace level
```

Where:

- *level*—Is a number from 0 to 16 that indicates the state you want to trace:

  0—Turns off process tracing.
  1—Traces process creations, deletions, suspensions and restarts.
  2—Traces alert messages to processes.
  4—Traces each time the system switches to another process.
  8—Traces each time a process is woken up by another process.

  You can trace multiple state changes by adding the numbers together. For example, to trace process alert messages (2) and process switches (4), use a *level* code of **6**.

EXAMPLES

The following example shows sample output from the **process status** command:

```
TAL> process status
System up 5 days, 5 hours, 45 minutes, 2 seconds
Interrupt stack: 562 (of 2048) bytes used.
Sleeps:  68414363 waits, 36926970 nops.
Wakeups: 19574643 signals, 15809563 woken, 3765080 nops, 2114 duplicates
         12093253 queued, 7 max queued, 0 lost.

PID       SP        stksize  maxstk  event     fl  in out name
9a7c0010  eb9707b2  1024     253     992d002c  IW         telnet display
9a9e0010  ec990418  576      208     8dd10010  IW         Discard server
9f3e0010  9efb03d8  512      80      80b82a9a  IW         iface idle
9c720010  9bef07d2  1024     474     80b8895a  IW         timer
9ee70010  89ba1e4e  0        0       9ecf002c  IW         console cmd
8dfd0010  8dda01aa  256      133     8be40010  IW         Telnet port
9dbb0010  9d7803c4  512      311     9dc40010  IW         console dsply.
8e610010  8e3e01ae  256      131     8c150010  IW         Echo listener
90160010  8feb0254  320      46      98120054  IW         scc0a out
99240010  98e203c6  500      194     80b88fa0  IW         TALK manager
9be60010  9ae30fd0  2048     563     80b88930  IW         network
9cea0010  9ca703d8  512      272     80b880a2  IW         killer
9ada0010  9aa702ba  384      122     80b836b4  IW         keyboard
8e930010  8e7001ae  256      131     8c2d0010  IW         Discard port
9c9e0010  9c7b01da  256      43      80b80096  IW         gcollect
9a850010  ec180690  1024     397     8e0e0010  IW         tn keyboard
8ef20010  8eaf03cc  512      94                I          ether0 out
9a730010  eb16042a  1024     573               I          tn command
8ef20010  8eaf03cc  512      94                I          ether0 out
```

The first section of the output displays operating system statistics, as follows:

- `Interrupt stack`—Shows the amount of memory used by the operating system when handling special time-oriented tasks that interrupt the currently active process. The maximum value shows the maximum amount of memory available for this purpose.
- `Sleeps`—Shows the number of times a process has been put to sleep because it is waiting for other processes to finish (`waits`) or has no operations to perform (`nops`).

- `Wakeups`—Shows the number of times a process has been wakened from sleep, and includes the following fields:
  - `Signals`—Shows how many times a process sent a signal to another process, trying to get its attention.
  - `Woken`—Shows how many times a signal succeeded.
  - `Nops`—Shows how many times a signal was ignored by a process.
  - `Queued`—Shows how many signals had to be set aside for later processing because the system was too busy.
  - `Duplicates`—Shows how many times a process that had already been wakened received a wakeup signal.
  - `Max queued`—Shows the maximum number of signal that were queued simultaneously.
  - `Lost`—Shows how many signals were lost because too many were queued or the system was too busy to respond.

The process list that follows the operating system statistics shows information about each process in the system:

- `PID`—Stands for *process ID*, which is a unique hexadecimal number assigned to each process that identifies the process to the TALnet operating system and other processes.
- `SP`—Stands for *stack pointer*, and shows the hexadecimal starting point in memory where the process stores temporary information needed to perform its tasks.
- `Stksize`—Shows how much memory is set aside for temporary information.
- `Maxstk`—Shows how much of the temporary memory has ever been used. (If this number approaches or exceeds the `Stksize` number, the system might become unstable. You can use the **process check** subcommand to automatically detect this condition.)
- `Event`—Shows the hexadecimal location in memory that other processes can reference to wake up the process.
- `Fl`—Stands for *flags*, and shows the current state of the process.This column can include any of the following:
  - `I`—Stands for *interrupts*, and is visible when the process can be interrupted by higher-priority tasks in the system.
  - `W`—Stands for *waiting*, and is visible when the process is asleep and waiting for the system or another process to wake it when system resources are available.
  - `S`—Stands for *suspended*, and shows that the process has been temporarily stopped.
- `In` and `out`—Show if any network traffic is currently waiting for the process.
- `Name`—Describes the process and its function.

SEE ALSO

log

A.26 PWD

NAME

pwd - Displays the path of your current working directory.

SYNOPSIS

pwd

DESCRIPTION

The **pwd** command displays the full path of your current working directory. The **pwd** command behaves the same as the **cd** command without arguments.

OPTIONS

None.

EXAMPLES

In the following example, the current working directory is `D:/TAL`:

```
TAL> pwd
D:/TAL
```

SEE ALSO

cd

A.27 REBOOT

NAME

reboot - Reinitializes the router.

SYNOPSIS

reboot

DESCRIPTION

The **reboot** command allows you to reboot, or fully reinitialize, the Wireless Router. You can issue the **reboot** command from a service console or through a Telnet session.

When you issue the **reboot** command, you are asked to confirm that you want to reinitialize the Wireless Router. If you are connected to the router directly through a service console and you answer **y**, you receive the message “Reboot initiated,” the configuration file reloads, and the TALnet prompt (defined by the **hostname** command) displays on the console screen. When the reinitialization completes, you receive a startup banner; do not enter any commands until you receive this banner. If you are connected to the router from a directly attached service console, and system administrator privileges have been configured, you will be prompted for a username and password. See the **user** command description. If you are connected to the router through a Telnet session and you answer **y**, the configuration file is reloaded and the Telnet session ends.

If you enter **n** when asked to confirm that you want to reinitialize the router, you receive the message “Reboot aborted” and return to the TALnet prompt (defined by the **hostname** command) and no reinitialization occurs.

OPTIONS

None.

EXAMPLES

The following example illustrates the **reboot** command from a directly attached console session. Do not enter any commands until you receive the startup banner.

```
TAL> reboot
Reboot (y/n)? y
Reboot initiated
TAL>
TALnet Release 1.2
Copyright 1992-1996 Tetherless Access Limited, All Rights Reserved.
Compiled at TAL: Feb 23 1996, 12:23:05
Router platform: SubSpace 2001 (rev1)
Memory Usage:    Library=41K, Code=389K, Data=36K
```

The following example illustrates what happens when you choose not to reinitialize the Wireless Router:

```
TAL> reboot
Reboot (y/n)? n
Reboot aborted.
TAL>
```

SEE ALSO

exit

A.28 RENAME

NAME

rename - Changes the name of a file.

SYNOPSIS

rename *old-file new-file*

DESCRIPTION

The **rename** command changes the name of a file. The argument *old-file* is the original filename. The argument *new-file* is the new name. You can also use the **rename** command to move a file from one directory to another. To move a file from another directory into your current working directory, include the directory path in the *old-file* argument. To move a file from your current working directory to another directory, include the directory path in the *new-file* argument. Note that you cannot rename files between logical drives.

Unlike the **copy** command, the original file no longer exists when you use the **rename** command.

OPTIONS

None.

EXAMPLES

The following example renames the existing `talnet.exe` file to a new file called `talold.exe`:

```
TAL> rename talnet.exe talold.exe
```

The following example moves the file `sample.cfg` from the current working directory to the directory `D:\TAL`:

```
TAL> rename sample.cfg D:\tal\sample.cfg
```

DIAGNOSTICS

Usage: rename <oldfile> <newfile>

You issued the **rename** command, but did not use the correct syntax. Check that you included both the original filename (oldfile) and a new filename (newfile).

Can't rename: No such file or directory

The original file that you specified does not exist. Use the **dir** command to ensure that the file is in the current directory and that you are using the correct filename.

SEE ALSO

copy, **dir**

A.29 REPEAT

NAME

repeat - Repeats a command on a specified interval.

SYNOPSIS

repeat [*interval*]*command*

DESCRIPTION

The **repeat** command reissues a command on a regular interval. The *interval* argument specifies the pause in milliseconds between each repeated command execution. In general, do not use a number lower than 1000 milliseconds (1 second). If you do not specify an interval, the default is 55 seconds. The *command* argument is the actual command line you want to repeat.

When you issue the **repeat** command, a new session labeled "Repeat" is created to display the output of the command you are repeating. In general, only use the **repeat** command with monitoring commands when you want to see how a state changes. Do not use the command with other commands that start their own session; for example, **hop check** or **tip**.

To stop the repeated execution of a command, use the **session** command to identify the session index for the "Repeat" session. Then issue the **reset** command to remove the repeat session.

OPTIONS

None.

EXAMPLES

The following example configures the Wireless Router to reissue the **talk show neighbor** command every 3000 milliseconds (3 seconds). A new session appears, displaying the output from the repeated **talk show neighbor** commands.

```
TAL> repeat 3000 talk show neighbors
```

SEE ALSO

reset, **session**

A.30 RESET

NAME

reset - Removes console sessions.

SYNOPSIS

reset [*session-index*]

DESCRIPTION

Use the **reset** command to remove console sessions. A console session is a set of commands and output that are grouped together to form a logical service function. Sessions keep related data together on one screen, separate from other service functions. The service console itself is a session. New sessions are created automatically when you issue certain commands. The following commands create a new session separate from the main console session:

- **domain cache list**
- **domain query**
- **ftp**
- **hop check**
- **log session**
- **ping** (when used with an *interval* argument)
- **repeat**
- **telnet**
- **tip**
- **trace**

Use the **reset** command without any arguments to remove the most recently opened session. To remove a specific session, identify the session index with the **session** command, then use that number as the *session-index* argument to the **reset** command.

Some of the commands that create new sessions have means for removing them directly, without using the **reset** command. See the command reference pages associated with those commands.

OPTIONS

None.

SEE ALSO

session

A.31 RIP

NAME

rip - Manages the parameters and tables associated with the Routing Information Protocol (RIP).

SYNOPSIS

rip *option*

DESCRIPTION

The **rip** command displays or sets up the system parameters and tables associated with RIP Versions 1 and 2. RIP is an interior gateway protocol; in other words, it is a routing protocol that updates all routes within a single autonomous system, thus allowing routers within that autonomous system to communicate with other routers in that autonomous system. This command allows you to set up entries in RIP neighbor tables.

OPTIONS

accept Accepts a previously refused RIP neighbor.

rip accept *hostid*

Where:

- *hostid*—Indicates the neighbor, and can be either a DNS name or an IP address expressed in four-part dotted decimal format.

add Adds a new entry to the RIP neighbor table.

rip add *hostid* [**rip1** | **rip1c** | **rip2** | **split** | **us**] [*port*]

Where:

- *hostid*—Indicates the neighbor, and can be either a DNS name or an IP address expressed in four-part dotted decimal format.
- **rip1**—Specifies a RIP Version 1 (RIP-1) neighbor.
- **rip1c**—Specifies a neighbor that is compatible with RIP-1.
- **rip2**—Specifies a RIP Version 2 (RIP-2) neighbor. This is the default if you do not specify the type of neighbor.
- **split**—Specifies a RIP neighbor with split-horizon processing.
- **us**—Includes the local interface’s addresses into the response.
- *port*—Is an integer UDP port number for the RIP daemon. The default is 520 if you do not specify a port number for the RIP daemon.

drop

Removes an entry from the RIP neighbor table.

rip drop *hostid*

Where:

- *hostid*—Indicates the neighbor, and can be either a DNS name or an IP address expressed in four-part dotted decimal format.

merge

If you specify an argument, this command merges redundant routing information by removing more-specific entries to the same destination. If you do not specify an argument, this command displays whether the feature is on or off.

rip merge [*boolean*]

Where:

- *boolean*—Turns the feature on or off. To turn the feature on, use **y**, **yes**, **true**, **on**, **1**, **set**, or **enable**. To turn the feature off, use **n**, **no**, **false**, **off**, **0**, **clear**, or **disable**.

refuse

Rejects RIP exchanges to or from a node.

rip refuse *hostid*

Where:

- *hostid*—Indicates the neighbor, and can be either a DNS name or an IP address expressed in four-part dotted decimal format.

request

Sends out a RIP request datagram to a RIP node.

rip request *hostid* [*version*] [*port*]

Where:

- *hostid*—Indicates the neighbor, and can be either a DNS name or an IP address expressed in four-part dotted decimal format.
- *version*—Indicates either RIP-1 or RIP-2, expressed as an integer. If you do not specify a version number, the default is RIP-2.
- *port*—Is a UDP port number for the RIP daemon, expressed as an integer. If you do not specify a port number, the default is 520.

status

Displays the RIP status counters and the neighbor table.

rip status

trace

If you specify an argument, this command changes the trace level for RIP activities. If you do not specify an argument, this command displays the RIP trace level.

rip trace [*mask*]

Where:

- *mask*—Is a trace number expressed as an integer indicating the debugging level of RIP processes.

EXAMPLES

```
TAL> rip status
RIP: sent 1 rcvd 1 reqst 0 resp 0 unk 0 refused 0
Active RIP output interfaces:
Dest Addr       Iface     Split     Us   Version
192.168.180.254 le0       FALSE     FALSE     RIP2
192.168.180.8   le0       FALSE     FALSE     RIP1c
```

DIAGNOSTICS

Unknown host

The IP address could not be resolved.

RIP unknown option???

Unknown option for the **rip** subcommand.

SEE ALSO

route

A.32 RMDIR

NAME

rmdir - Deletes a directory.

SYNOPSIS

rmdir *directory-name*

DESCRIPTION

The **rmdir** command allows you to delete, or remove, a directory specified by the *directory-name* argument. You can only delete one directory at a time, and the directory you attempt to delete must not contain any files or directories. To remove files within a directory, use the **delete** command. You can delete a directory in another location by including the path in the directory name.

OPTIONS

None.

EXAMPLES

The following example deletes the directory `sample`:

```
TAL> rmdir sample
```

DIAGNOSTICS

Usage: rmdir <directory>

You issued the **rmdir** command, but either did not include a directory name, or the directory you specified does not exist in your current working directory. Use the **dir** command to list the contents of the current directory.

Can't rmdir: Permission denied

The directory you tried to delete includes files. You must delete the contents of a directory before you can delete the directory itself.

Can’t rmdir: No such file or directory

The directory you tried to delete does not exist. Verify that you used the correct directory name.

SEE ALSO

delete, **dir**, **mkdir**, **user**

A.33 ROUTE

NAME

route - Manages the IP routing table.

SYNOPSIS

route [*option*]

DESCRIPTION

The **route** command displays or manipulates the IP routing table. You can create a routing table yourself, or it can be built dynamically using the RIP daemon. This command allows you to set up or update the IP routing entries for both network and host routes.

If you do not specify an argument, this command displays the routing table.

OPTIONS

add

Adds a new entry into the IP routing table.

route add {*dest-ip-addr*[/*bits*] | **default**} *iface* [*gateway* [*metric*]]

Where:

- *dest-ip-addr*—Indicates the target IP address expressed in four-part dotted decimal format.
- */bits*—Is a slash followed by an integer indicating the number of effective subnet mask bits.
- **default**—Defines a default route.
- *iface*—Is the symbolic interface name that indicates the interface through which the datagram is routed.
- *gateway*—Indicates the target router, and is an IP address expressed in four-part dotted decimal format, or **none**. Use the **none** keyword when you do not want to specify a target router but do want to specify a hop count.
- *metric*—Is an integer indicating the hop count to the destination.

addprivate

Adds a private entry into the IP routing table. Private entries are not exchanged by RIP daemons, so they are not propagated to other routing tables in the network.

route addprivate {*dest-ip-addr*[/*bits*] | **default**} *iface* [*gateway metric*]

See the description of the **route add** command for argument definitions.

drop Removes an entry from the IP routing table.

route drop *dest-ip-addr*[/*bits*]

See the description of the **route add** command for argument definitions.

flush Flushes the whole routing table except local network routes.

route flush

lookup Finds a route entry to a specific destination.

route lookup *dest-ip-addr*

Where:

- *dest-ip-addr*—Indicates the target IP address, and can be a DNS name or an IP address expressed in four-part dotted decimal format.

EXAMPLES

```
TAL> route
Dest              Len  Interface  Gateway     metric   P   Timer   Use
255.255.255.255    32   le0                      1     P    0       0
192.168.180.0      24   le0                      1          0       2
default             0   le0       192.168.180.1  1     P    0       3
```

DIAGNOSTICS

Unknown host

The IP address could not be resolved.

Host ??? unreachable

The IP address could not be reached.

Interface ??? unknown

Unknown interface name.

Can't add route

An attempt to add a new route failed.

SEE ALSO

rip

A.34 SESSION

NAME

session - Displays or manipulates console sessions.

SYNOPSIS

session [*session-index*]

DESCRIPTION

Use the **session** command to view information about current console sessions or to switch to a different session. A console session is a set of commands and output that are grouped together to form a logical service function. Sessions keep related data together on one screen, separate from other service functions. The service console itself is a session. New sessions are created automatically when you type certain commands.The following commands create a new session separate from the main console session:

- **domain cache list**
- **domain query**
- **ftp**
- **hop check**
- **log session**
- **ping** (when used with an *interval* option)
- **repeat**
- **telnet**
- **tip**
- **trace**

Note Do not create more than eight sessions at one time.

Use the **session** command without any arguments to view information about current sessions. Then you can use the session index (the first column of data displayed) to switch to that session, using the **session** [*session-index*] form of the command. You can also switch between sessions by entering **Ctrl-w**. To close a session, use the **reset** command.

OPTIONS

None.

SEE ALSO

reset

A.35 SNMP

NAME

snmp - Configures support for the Simple Network Management Protocol (SNMP).

SYNOPSIS

snmp community add *community access*

snmp community delete *community*

snmp community show {*community* | **all**}

snmp syscontact [*name*]

snmp syslocation [*description*]

DESCRIPTION

SNMP is an application-layer network management protocol that allows you to send queries to or set variables on network devices such as routers. SNMP consists of three elements:

- SNMP manager—A network management client program that requests or sets information. SNMP managers run on Network Management Stations (NMSs).
- SNMP agent—A software module on a managed network device, such as a router. The agent contains information the manager can monitor or change, and sends traps to the manager that alert the manager when network conditions change.
- Management Information Base (MIB)—A repository on the managed network device that contains performance and administrative information about that device. The MIB defines variables maintained by the agent that the manager can request or set. (*MIB* also refers to the document that describes the information that can be read or set on a managed device.)

SNMP supports the following operations:

- Get-request—The SNMP manager sends a request to the agent, asking for a value from a specific variable.
- Get-next-request—The SNMP manager sends a request to the agent, asking for a value of the variable following the named variable. Get-next requests are useful for traversing entire branches of a MIB database.
- Set-request—The SNMP manager sends a value for a specific variable to the agent to store.
- Get-response—The SNMP agent replies to the Get or Get-next request.
- Trap—The SNMP agent sends an unsolicited message to the manager, alerting the manager that a condition on the network has changed.

Note The current version of the TALnet software does not support trap operations. Set operations are only valid for the current session. In addition, the software does not support SNMPv2 get-bulk-request operations.

OPTIONS

community add

Defines a community string and the associated access to the SNMP agent. When network managers send SNMP requests to an agent, they specify the community they belong to. You must define at least one community before you can start the SNMP agent process.

snmp community add *community access*

Where:

- *community*—Specifies the community name, and can be a single word using uppercase and lowercase letters and numbers.
- *access*—Indicates the level of access members of that community have, and can be either **read-write** or **read-only**. Communities with read-only access can only perform get-request and get-next-request operations. Communities with read-write access can also perform set-request operations.

community delete

Removes the specified community from the list of accepted communities. You cannot remove all communities if SNMP is currently running. To remove all communities, first stop SNMP with the **stop snmp** command.

snmp community delete *community*

Where:

- *community*—Specifies the community you want to delete, and is the community name defined with the **snmp community add** subcommand.

community show

Shows the specified community or all defined communities and their associated access.

snmp community show {*community* | **all**}

Where:

- *community*—Specifies a single community, and is a community name defined with the **snmp community add** subcommand.
- **all**—Specifies all defined communities.

syscontact

Configures the SNMP agent to return the system contact information on queries from management stations. If you do not specify an argument, this command displays the currently configured system contact variable.

snmp syscontact [*name*]

Where:

- *name*—Sets the value of the SNMP variable *system.sysContact.0* to the specified string, and can include uppercase and lowercase letters, numbers, and spaces.

syslocation

Configures the SNMP agent to return the system location information to queries from management stations. If you do not specify an argument, this command displays the currently configured location variable.

snmp syslocation [*description*]

Where:

- *description*—Sets the value of the SNMP variable *system.sysLocation.0* to the specified string, and can include uppercase and lowercase letters, numbers, and spaces.

EXAMPLES

The following example defines a community named `public` that has `read-only` access. Network managers who belong to this community can only perform get-request or get-next-request operations.

```
TAL> snmp community add public read-only
```

In the following example, the first line defines the system contact as `John Doe`. The second line requests the currently defined system contact.

```
TAL> snmp syscontact John Doe

TAL> snmp syscontact
John Doe
```

In the following example, the first line defines the system location as in the Machine Hall, in rack 2B. The second line requests the currently defined system location.

```
TAL> snmp syslocation Machine Hall, rack 2B

TAL> snmp syslocation
Machine Hall, rack 2B
```

SEE ALSO

start, **stop**

A.36 START

NAME

start - Enables server processes that provide various services within the TALnet software.

SYNOPSIS

start *option*

DESCRIPTION

The **start** command allows you to enable optional servers that process service requests. To disable these servers, use the **stop** command.

OPTIONS

discard Starts the discard server. This server receives packets but does nothing with them. This is useful for testing network connections. A remote host can connect to the discard server and send packets; the packets themselves are discarded, but the effects of their transmission and routing can be observed using other monitoring commands.

start discard

echo Starts the echo server. This server responds to Internet Control Message Protocol (ICMP) echo datagrams, which are usually generated by the **ping** facility. This is useful for testing network connections, as well as testing whether hosts and routers are able to respond to simple messages.

start echo

ftp Starts the File Transfer Protocol (FTP) server, which implements part of FTP. This server responds to incoming connections from a remote FTP client. The FTP client and server pair are used to transfer files between hosts. This server must be enabled before FTP connections can be made to the router.

start ftp

rip Starts the Routing Information Protocol (RIP) process. This dynamic routing protocol allows the router to exchange information with other routers regarding the state of routes through the network.

start rip [*secs*]

Where:

- *secs*—Specifies the interval in seconds between transmissions of the routing table to other routers. The default is 30.

snmp Starts the Simple Network Management Protocol (SNMP) agent process, which processes incoming SNMP requests. SNMP requests are operations on important configuration, performance, and error data.

```
start snmp
```

telnet Starts the Telnet server, which implements a remote access login protocol. This server responds to incoming connections from a remote Telnet client. The Telnet client and server pair provide remote control for service and support functions. This server must be enabled before Telnet connections can be made to the router.

```
start telnet
```

SEE ALSO

stop

A.37 STOP

NAME

stop - Disables server processes that provide various services within the TALnet software.

SYNOPSIS

stop *option*

DESCRIPTION

The **stop** command allows you to disable optional servers that process various service requests. To enable these servers, use the **start** command.

OPTIONS

discard Stops the discard server. This server receives packets but does nothing with them. This is useful for testing network connections. A remote host can connect to the discard server and send packets; the packets themselves are discarded, but the effects of their transmission and routing can be observed using other monitoring commands.

```
stop discard
```

echo Stops the echo server. This server responds to Internet Control Message Protocol (ICMP) echo datagrams, which are usually generated by the ping facility. This is useful for testing network connections, as well as testing whether hosts and routers are able to respond to simple messages.

```
stop echo
```

ftp Stops the File Transfer Protocol (FTP) server, which implements part of FTP. This server responds to incoming connections from a remote FTP client. The FTP client and server pair are used to transfer files between hosts.

```
stop ftp
```

rip Stops the Routing Information Protocol (RIP) process. This dynamic routing protocol allows the router to exchange information with other routers regarding the state of routes through the network.

```
stop rip
```

snmp Stops the Simple Network Management Protocol (SNMP) agent process, which processes incoming SNMP requests. SNMP requests are operations on important configuration, performance, and error data.

```
stop snmp
```

telnet Stops the Telnet server, which implements a remote access login protocol. This server responds to incoming connections from a remote Telnet client. The Telnet client and server pair provide remote control for service and support functions.

```
stop telnet
```

SEE ALSO

start

A.38 TALK DEBUG

NAME

talk debug - Manages TALtalk protocol debugging.

SYNOPSIS

talk debug {**all** | **errors** | **events** | **neighbors** | **packets** | **states**} [**off**]

DESCRIPTION

The **talk debug** command allows you to view information that might help optimize or provide insight into the operation of your wireless network.

DEBUG OPTIONS

all Enables or disables all possible debugging information.

talk debug all [**off**]

errors Enables or disables debugging of all serious protocol errors and violations.

talk debug errors [**off**]

events Enables or disables debugging of major TALtalk protocol events.

talk debug events [**off**]

neighbors Enables or disables logging and debugging of adjacent radio events such as a radio being discovered or abandoned.

talk debug neighbors [**off**]

packets Enables or disables logging of all packets sent and received by the TALtalk protocol. Use this option with caution; on a busy network, many console messages might be generated.

talk debug packets [**off**]

states Enables or disables logging of all TALtalk protocol software state transitions. This might be helpful to TAL personnel to verify proper software operation.

talk debug states [**off**]

SEE ALSO

talk radio, **talk show**, **talk tune**

A.39 TALK RADIO

NAME

talk radio - Initializes, configures, and manages wireless radio interfaces.

SYNOPSIS

talk radio *iface-name option*

DESCRIPTION

The **talk radio** command allows you to initialize, configure, and modify wireless radio interfaces that use the TALtalk protocol.

Before a wireless interface can function, you must provide the following basic configuration information:

- Interface name—The interface name lets you refer to wireless interfaces with familiar names that are appropriate to your network environment.
- TALtalk address—The TALtalk link-layer address is an 8-digit hexadecimal number beginning with the characters *0x*. Hexadecimal numbers use digits between 0 and 9 and uppercase or lowercase letters between A and F. For example, *0x41544D31* and *0x00000001* are valid TALtalk addresses. TALtalk addresses are usually assigned by a site coordinator, TAL, or a central authority serving your area.
- Transmission frequency—Wireless interfaces must have specific radio frequencies assigned to them. This must be the same frequency used by other radios in your wireless network.
- Transmission power level—The power level controls the strength of the radio transmission. This value should be set to the lowest possible setting that provides good data service.
- Pseudorandom noise (PN) code—TALtalk wireless interfaces provide eight PN codes (also called spreading codes) to encode network data for transmission. All radios in your wireless network must use the same PN code.

OPTIONS

address Assigns a name and TALtalk address to a wireless interface. This command also initializes the serial communications controller used by the router to communicate with the attached radio. You must issue this command before you issue any other **talk radio** commands.

talk radio *iface-name* **address** *talk-address* [**model** {**L** | **S**}] **config scc**

Where:

- *iface-name*—Assigns a symbolic name to the interface. The name can include hyphens, numbers, and uppercase and lowercase letters, but must be one word. Use a meaningful name; we recommend *radio0* or *radio1*.

- **address** *talk-address*—Associates the symbolic interface name with the synchronous serial device. The address is the TALtalk link-layer address, which is usually assigned by your site coordinator, TAL, or a central authority. The address is an 8-digit hexadecimal number preceded by the characters *0x*. For example, *0x41544D31* and *0x00000001* are valid TALtalk addresses.
- **model** {**L** | **S**}—Specifies the band in which the radio operates. L-band radios operate in the 902- to 928-MHz frequency range. S-band radios operate in the 2400- to 2483.5-MHz frequency range. If you do not specify the radio model, the router assumes you are configuring an L-band model.
- **config scc**—Initializes the synchronous serial device used by the router to communicate with the wireless interface.

analyze

Analyzes the received signal strength (RSS) of packets from transmitters. The receiving Wireless Router determines the average RSS for incoming data packets. If you do not specify the **on** or **off** keyword, the current state of RSS sampling displays. To view the averages, use the **talk show rss** command.

talk radio *iface-name* **analyze rss** [**on** | **off**]

Where:

- *iface-name*—Represents the symbolic name of the wireless interface you are analyzing. This name is the one you assigned with the **talk radio address** command.
- **on**—Turns on the RSS analysis feature.
- **off**—Turns off the RSS analysis feature.

auto-neighbor

Sets up dynamic routing so that new neighbors are automatically added to or removed from the RIP neighbor table. The neighbor table does *not* automatically update by default. If you do not specify the **on** or **off** keyword, the status of this feature displays.

talk radio *iface-name* **auto-neighbor** [**on** | **off**]

Where:

- **on**—Enables automatic neighbor updates.
- **off**—Disables automatic neighbor updates.

channel

Assigns the radio frequency, output power, and PN code to a wireless interface. Use the same interface name specified in the **address** subcommand. The *channel-name* can be any familiar name to help you refer to the radio channel when using other TALtalk commands.

Available frequencies and power settings are outlined in the AVAILABLE FREQUENCIES, CHANNELS, AND POWER SETTINGS section.

talk radio *iface-name* **channel** *channel-name* **config** {**MHZ** *frequency* | **number** *channel-num*} **MW** *power* **PN** *code*

Where:

- *iface-name*—Represents the symbolic name of the wireless interface you are configuring. This name is the one you assigned with the **talk radio address** command.
- **channel** *channel-name*—Assigns a symbolic name to the channel.
- **MHZ** *frequency*—Configures the frequency on which the radio will operate. The *frequency* value is specified in megahertz (MHz), and must be the same for all radios within your wireless network. This keyword and argument and the **number** *channel-num* keyword and argument both set the frequency. Do not use both keyword-argument pairs. See Table A-2 for available frequencies and associated channel numbers.
- **number** *channel-num*—Is the number of the channel associated with the frequency. The *channel-num* value can be an integer between 1 and 9 on L-band radios and between 1 and 15 on S-band radios. The channel number must be the same for all radios within your wireless network. This keyword and argument and the **MHZ** *frequency* keyword and argument both set the frequency. Do not use both keyword-argument pairs. See Table A-2 for channel numbers and frequencies.
- **MW** *power*—Configures the power level at which the radio will transmit. The *power* value should be the lowest possible setting that provides reliable data transmission, and should be determined during your link analysis. Specify the power in milliwatts (mW). See Table A-3 for available power settings.
- **PN** *code*—Specifies one of eight PN codes the radio will use to encode data for network transmission. The *code* value can be an integer between 1 and 8; all radios in your network must use the same code.

neighbor {**ignore** | **hear**}

Configures the Wireless Router to ignore or hear neighboring transmitters. For example, if a neighboring transmitter is powerful enough for your local router to hear it, but not powerful enough to provide quality data service, you might want to ignore the neighboring transmitter. This is an advanced command, and should be used with caution.

talk radio *iface-name* **neighbor** {**ignore** | **hear**} *talk-address*

Where:

- *iface-name*—Is the symbolic name assigned by the **talk radio address** command.
- **ignore**—Indicates that the router you are configuring should ignore data from the specified neighbor.
- **hear**—Indicates that the router you are configuring should listen to data from the specified neighbor. Use this option to reverse a previously ignored neighbor.
- *talk-address*—Is the TALtalk link-layer address of the neighboring transmitter.

neighbor MW

Specifies the transmit power on a per-link basis; in other words, you can increase or decrease the transmit power to a specific neighbor. Modifying the power on a per-link basis is usually done as a result of an RSS analysis on the receiving Wireless Router.

talk radio *iface-name* **neighbor** *talk-address* **MW** {*power* | **default**}

Where:

- *iface-name*—Represents the symbolic name of the wireless interface you are configuring. This name is the one you assigned with the **talk radio address** command.
- **neighbor** *talk-address*—Identifies the link on which you are changing the power. The *talk-address* argument is the TALtalk link-layer address of the neighboring transmitter.
- **MW** *power*—Configures the power level at which the radio will transmit. Specify the power in milliwatts (mW). See Table A-3 for available power settings.
- **default**—Returns the transmission power to the specified neighbor to the default level for that radio. The power set with the command **talk radio channel** is the default transmission power.

EXAMPLES

```
TAL> talk radio radio1 address 0x54532123 config scc
TAL> talk radio radio1 channel one config MHZ 925.397 PN 7 MW 10
TAL> talk radio radio1 neighbor ignore 0x23124323
TAL> talk radio scc0a neighbor 0x11223344 MW 200
```

AVAILABLE FREQUENCIES, CHANNELS, AND POWER SETTINGS

Table A-2 lists available transmission frequencies for L-band and S-band radios.

Table A-2 Transmission Frequency Channels

Channel	L-Band	S-Band
1	904.601	2407.067
2	907.201	2412.265
3	909.801	2417.465
4	912.401	2422.663
5	915.000	2427.863
6	917.599	2433.060
7	920.199	2438.259
8	922.798	2443.457
9	925.397	2448.659
10	—	2453.857
11	—	2459.056
12	—	2464.254
13	—	2469.454
14	—	2474.653
15	—	2479.851

Table A-3 lists power settings for L-band and S-band radios.

Table A-3 Power Settings

L-Band		S-Band	
mW	**dBm**	**mW**	**dBm**
1	0	1	0
3	4.8	3	4.8
7	8.5	7	8.5
10	10	10	10
17	12.3	17	12.3
23	13.6	23	13.6
32	15	32	15
45	16.5	45	16.5
66	18.2	66	18.2
77	18.9	77	18.9
88	19.4	88	19.4

Table A-3 Power Settings

L-Band		S-Band	
mW	**dBm**	**mW**	**dBm**
100	20	100	20
111	20.5	111	20.5
144	21.6	144	21.6
166	22.2	166	22.2
188	22.7	188	22.7
202	23.1	202	23.1
222	23.5	222	23.5
250	24	250	24
277	24.4	271	24.3
303	24.8	293	24.7
333	25.2	314	25
377	25.8	336	25.3
411	26.1	357	25.6
444	26.5	378	25.8
500	27	400	26
555	27.4	442	26.5
599	27.8	483	26.8
655	28.2	525	27.2
699	28.4	566	27.5
755	28.8	608	27.8
800	29	650	28.1

DIAGNOSTICS

Radio ‘*name*’ already exists.

You attempted to assign a TALtalk address to a radio that already exists.

Radio ‘*name*’ has not been defined yet.
Please configure it, using the ‘address’ command.

You attempted to assign channel information to a radio that has not been named yet. Use the **address** subcommand to assign the radio a name.

Expecting talk radio <interface> <subcommand>.

You entered a **talk radio** command, but did not include either an **address** keyword (to specify the device driver) or a **channel** keyword.

Missing device driver or channel specification.

You entered the **talk radio address** subcommand, but did not specify **config scc**.

Missing radio model specification.

When you entered a **talk radio address** subcommand with the **model** keyword, you did not specify the actual radio model. You must enter either **L** or **S**. The configuration will assume that you are configuring an L-band radio until you revise the command.

WARNING: Unsupported radio model: '*string***', defaulting to L**

When you entered the **talk radio address** subcommand, you entered an invalid string following the **model** keyword. You must enter either **L** or **S**. The configuration will assume that you are configuring an L-band radio until you revise the command.

Expecting radio subcommand '*string***'.**

You entered an invalid keyword in the **talk radio address** or **talk radio channel** subcommand.

Unknown radio subcommand. Expecting 'config' or 'delete'

You entered a **talk radio address** or **talk radio channel** subcommand, but did not enter the keyword **config** or **delete**.

Missing channel config commands.

You entered a **talk radio channel** subcommand but did not enter any keywords to configure the channel.

String **is an invalid frequency.**

The string of text you supplied for the frequency is not a valid argument. Refer to Table A-2 for valid frequencies.

Malformed decimal number *string*

You supplied an invalid value after the decimal point when setting the frequency. Refer to Table A-2 for valid frequencies.

*Number***.***number* **is not a valid frequency.**

The value you supplied as an argument to the **MHZ** keyword is not a valid frequency. Refer to Table A-2 for valid frequencies.

No more than one 'MHZ' or 'number' command per channel

You attempted to supply more than one frequency setting or channel number setting by issuing both the **MHz** and **number** keywords or repeating these keyword and argument pairs.

Must specify a frequency for this channel.

You did not assign a frequency with the **MHz** keyword or a frequency channel with the **number** keyword. You must supply one or the other.

Valid channel numbers for this radio are from 1 thru 9.

When configuring an L-band radio, the argument you supplied to the **number** keyword was not a number from 1 to 9.

Valid channel numbers for this radio are from 1 thru 15.

When configuring an S-band radio, the argument you supplied to the **number** keyword was not a number from 1 to 15.

No more than one 'PN' selection per channel

You attempted to assign more than one PN code to a channel.

PN Codes are values from 1 thru 8.

The argument you supplied to the **PN** keyword is not a number from 1 to 8.

Must specify a PN code for this channel.

You did not assign a PN code with the **PN** keyword.

Expecting a whole number, but got "*string*" instead.

You entered an argument to the **MW** keyword that includes characters that are not numbers. If the configuration file defines the power to be 1 milliwatt, the router will reset the power to 45 milliwatts. If the configuration file sets any other power level, that power level is used.

Power cannot be given as a negative number.

You entered an invalid number as an argument to the **MW** keyword. Revise the command with one of the values in Table A-3.

Please use a number less than or equal to 800 milliwatts. Power has been set to the maximum of 800 milliwatts.

On an L-band radio, you attempted to specify a power that is higher than 800 mW. Refer to Table A-3 for valid values. Until you revise the command, the maximum power setting of 800 mW will be used.

Please use a number less than or equal to 650 milliwatts. Power has been set to the maximum of 650 milliwatts.

On an S-band radio, you attempted to specify a power that is higher than 650 mW. Refer to Table A-3 for valid values. Until you revise the command, the maximum power setting of 650 mW will be used.

No more than one 'MW' power selection per channel

You entered the keyword **MW** more than once in a **talk radio channel** subcommand.

Must specify power output for this channel.

When you entered the **talk radio channel** subcommand, you did not enter the keyword-and-argument pair **MW** *power.*

Cannot add neighbor to 'ignore' table. No memory.

You attempted to configure the Wireless Router to ignore a specific neighbor, but did not have enough memory.

SEE ALSO

talk debug, **talk show, talk tune**

A.40 TALK SHOW

NAME

talk show - Manages TALtalk protocol statistics.

SYNOPSIS

talk show neighbors [**verbose**]

talk show radio

talk show rss [*talk-address*]

talk show timers

talk show debug

DESCRIPTION

The **talk show** command allows you to view information that might help optimize or provide insight into the operation of your wireless network.

OPTIONS

neighbors

Shows statistics about the remote radios that this router is capable of transmitting to or receiving from. The **verbose** option displays even more information about these neighbors.

talk show neighbors [*talk-address*] [**verbose**]

Where:

- *talk-address*—Is the TALtalk link-layer address of the neighboring transmitter. If you do not specify a neighbor, this command shows the average RSS for all neighbors that you analyzed. If you do not specify this argument, all neighbors display.

radio

Displays currently configured radio parameters (radio model, channel number, frequency, default and per-link power settings, and PN code):

talk show radio

rss

Shows the accumulated results of the RSS analysis performed with the **talk radio analyze** command.

talk show rss [*talk-address*]

Where:

- *talk-address*—Is the TALtalk link-layer address of the neighboring transmitter. If you do not specify a neighbor, this command shows the average RSS for all neighbors that you analyzed.

timers Shows information about the various timing values the TALtalk protocol is currently using.

talk show timers

debug Shows which parts of the TALtalk protocol are presently generating debugging information.

talk show debug

RECEIVED SIGNAL STRENGTH TABLE

Table A-4 shows the digital RSS value that appears in the output of the **talk show rss** command, and relates that value to the signal quality. (Note that exact values might vary.) Ideally, the signal quality should be high enough to allow an average BER of 10^{-6} or lower.

Table A-4 Received Signal Strength (RSS)

DC Voltage at the RSS Test Point	Digital RSS Value in Decimal	Signal Quality (Average BER)	RSS in dBm (Interference Free)
7.5	192	$<10^{-10}$	>-70
7.0	179	$<10^{-10}$	>-70
6.5	166	$<10^{-10}$	>-70
6.0	154	$<10^{-10}$	-70
5.5	141	$<10^{-10}$	-88
5.0	128	10^{-7}	-94
4.5	115	10^{-6}	-95
4.0	102	10^{-5}	-98
3.5	90	10^{-4}	-100
3.0	77	10^{-3}	-101
2.5[1]	64	10^{-3}	-102
2.0[1]	51	10^{-2}	-103
1.5[1]	38	10^{-2}	-105

[1] Occasional sync loss

EXAMPLES

The following example shows the currently configured radio parameters:

```
TAL> talk show radio
Radio radio0:
    Model L, operating on channel 9 (925.397 MHz) with PN code 7.
    Default power output is 1 milliwatt

    Custom power settings are used to talk to the following neighbors:
         11 22 33 44 at 800 milliwatts     55 66 77 88 at 500 milliwatts
```

The following example lists all neighbors:

```
TAL> talk show neighbors
TALK address: PRT1
  Idle state, 0 in send queue, 0 active neighbors
  sent last hello 10 seconds ago.
  Outgoing transaction ID: 0. Incoming ID: 0.

                            OCTETS          OUTPUT DIALOGS
Neighbor   State       Input     Output   Attempts    Failed  Last Data
---------- -------- ---------- --------- --------- --------- -----------
00 11 52 13 Good            196       124        2         0   0+00:01:07
```

The following example displays all Wireless Router neighbors, and provides more-detailed information about the radio links to these neighbors.

```
TAL> talk show neighbors verbose
TALK address: 00 11 22 33
  Idle state, 0 in send queue, 1 active neighbor,
  sent last hello 10 seconds ago.
  Outgoing transaction ID: 0. Incoming ID: 0.

  Neighbors:
    Neighbor at 00 11 52 13 is Good (as of 2 seconds ago)
    0 transitions to 'suspect' state.
    Found IP peer.  (In 2 tries)

    Output: 2 requests, 0 dialog failures, 0 system errs
            124 bytes, 0 cfm timeouts, 0 ack timeouts
            0 bad control ids, 0 attempts while inhibited
     Input: 2 requests, 2 completions
            376 bytes, bad control ids, 0 data timeouts
```

The following example shows the RSS for neighboring Wireless Routers `PRT3` and `PRT2`. The signal strength for PRT3 is `192`. Note that, according to Table A-4, this is equivalent to a BER of less than 10^{-10}.

```
TAL> talk show rss
Neighbor      Strength   Notes
------------  --------   -------------------------------------
PRT3            192
PRT2              0      Not enough samples-- need more traffic.
```

SEE ALSO

talk debug, **talk radio, talk tune**

A.41 TALK TUNE

NAME

talk tune - Allows you to manipulate link-layer acknowledgments.

SYNOPSIS

talk tune acks [*iface-name* {**on** | **off**}]

DESCRIPTION

The **talk tune** command allows you to turn link-layer acknowledgments on or off on a per-radio basis. If you enable link-layer acknowledgments, the receiving node must acknowledge every frame it receives. If the sending node does not receive an acknowledgment, it resends the frame. The *iface-name* argument is the symbolic name for the radio interface you are configuring. If you specify the **talk tune acks** command without any arguments, the status of link-layer acknowledgments on all radios displays.

Caution If you turn link-layer acknowledgments on, you must turn them on for all Wireless Routers that communicate within your internetwork.

SEE ALSO

talk debug, **talk radio**, **talk show**

A.42 TCP

NAME

tcp - Configures and manages Transmission Control Protocol (TCP) parameters.

SYNOPSIS

tcp *option*

DESCRIPTION

The **tcp** command allows you to configure and manage parameters associated with TCP. TCP is a connection-oriented transport-layer protocol that provides reliable transport of data from one application to another. TCP breaks messages into packets and provides timeout services, but relies on IP to transmit packets across internetwork. TCP uses standard IP addresses and also a per-host collection of port addresses. Therefore, each address contains an IP number and a port number. TCP assumes IP to be unreliable. To combat this, it uses a windowing flow-control device. It uses positive acknowledgments (ACKs), sequence numbers (SYNs), and retransmission to enable TCP to recover when bad data occurs.

OPTIONS

ceiling

Sets the maximum round-trip time (rtt) in milliseconds. This command is useful for specifying when TCP SYNs time out and data must be retransmitted. If you do not specify an argument, this command displays the current value.

tcp ceiling [*ms*]

Where:

- *ms*—Is the maximum round-trip time in milliseconds (ms). If you do not set this round-trip time, the default is 120,000 ms (120 seconds).

floor

Sets the minimum round-trip time in milliseconds. This command allows you to keep the system from flooding the other end of the TCP connection with data or retransmissions. If you do not specify an argument, this command displays the current value.

tcp floor [*ms*]

Where:

- *ms*—Is the minimum round-trip time in milliseconds. If you do not set this round-trip time, the default is 500 ms (0.5 seconds).

irtt

Sets the initial round-trip time estimate, in milliseconds, to be used for new TCP connections until they can measure and adapt to the actual value. Increasing the initial round-trip time when operating over slow channels will avoid the flurry of retransmissions that would otherwise occur as the smoothed estimate settles down at the correct value. To affect incoming connections, issue this command before servers are started. If you do not specify an argument, this command displays the current value.

tcp irtt [*ms*]

Where:

- *ms*—Is the initial round-trip time in milliseconds. If you do not set this round-trip time, the default is 120,000 ms (120 seconds).

kick

If there is unacknowledged data on the send queue of the specified transmission control block, this command forces an immediate retransmission.

kick *tcb*

Where:

- *tcb*—Is the identification number of the transmission control block.

limit

Sets the outgoing window limit in bytes. Windowing allows a certain amount of data to continue being transmitted when no acknowledgments for previous data have been received. This removes overhead on slow links. If you do not specify an argument, this command displays the current window limit.

tcp limit [*bytes*]

Where:

- *bytes*—Is the size of the window limit.

mss

Sets the number of bytes for the maximum segment size that will be sent on all outgoing TCP connection requests (SYN segments). This tells the remote end the size of the largest segment (packet) it may send. Changing the maximum segment size affects only future connections; existing connections are unaffected. If you do not specify an argument, this command displays the current value.

tcp mss [*bytes*]

Where:

- *bytes*—Is the maximum segment size. If you do not set a maximum segment size, the default is 1460.

reset

Deletes the specified transmission control block, which forces the connection to abort. You can discover the current transmission control block using **tcp status**.

tcp reset *tcb*

Where:

- *tcb*—Is the identification number of the transmission control block.

rtt

Replaces the automatically computed round-trip time in the specified transmission control block with the specified round-trip time in milliseconds. This command speeds recovery from a series of lost packets because it provides a manual bypass around the normal backoff retransmission timing mechanisms.

tcp rtt *tcb ms*

Where:

- *tcb*—Is the identification number of the transmission control block.
- *ms*—Is the round-trip time in milliseconds.

status

Displays the status of TCP data sent and received. For sample output, see the EXAMPLES section.

tcp status

syndata

Determines whether or not sequence numbers (SYN) and data can travel together in the same packet. Some systems cannot handle SYN and data together. If you do not specify an argument, this command displays the current setting.

tcp syndata [*boolean*]

Where:

- *boolean*—Turns the feature on or off. To turn the feature on, use **y**, **yes**, **true**, **on**, **1**, **set**, or **enable**. To turn the feature off, use **n**, **no**, **false**, **off**, **0**, **clear**, or **disable**.

trace

Sets tracing on or off. When tracing is enabled, it echoes TCP connection status, including *SYN sent*, *Established*, and *Last ACK*, to the console. If you do not specify an argument, this command displays the current setting.

tcp trace [*boolean*]

Where:

- *boolean*—Turns the feature on or off. To turn the feature on, use **y**, **yes**, **true**, **on**, **1**, **set**, or **enable**. To turn the feature off, use **n**, **no**, **false**, **off**, **0**, **clear**, or **disable**.

window Changes the window size by setting the number of bytes of a TCP packet that TALnet will accept. If you do not specify an argument, this command displays the current window size.

tcp window [*bytes*]

Where:

- *bytes*—Is the window size in bytes. If you do not set the window size, the default is 2920.

EXAMPLES

```
TAL> tcp status
TCP Statistics:
Segments: 1298 in, 1292 out, 0 errors, 0 retransmitted.
Connects: 6 attempts (2 active, 3 passive) 1 failed.
          2 streams are currently open.
          0 were forcibly closed on us.
          We forcibly closed 0 streams.

ID       Rcv-Q Snd-Q Local socket        Remote socket       State
7aa80000    0     0  192.168.180.249:23  192.168.180.2:294   Established
82f20008    0     0  0.0.0.0:23          0.0.0.0:0           Listen (S)
7ea30008    0     0  192.168.180.249:1024 192.168.180.2:21   Established
830c0000    0     0  0.0.0.0:21           0.0.0.0:0          Listen (S)
831d0000    0     0  0.0.0.0:7            0.0.0.0:0          Listen (S)
83320000    0     0  0.0.0.0:9            0.0.0.0:0          Listen (S)
```

Where:

- `Segments`—Provides the following statistics about segments (packets):
  - `in`—Shows the number of TCP segments (packets) received.
  - `out`—Shows the total number of TCP segments transmitted.
  - `errors`—Shows the number of received segments with errors.
  - `retransmitted`—Shows the number of segments that were retransmitted.
- Connects—Provides the following statistics about connections:
  - `attempts`—Shows the number of TCP connections initiated locally.
  - `active`—Shows the number of TCP connections initiated to a remote host.
  - `passive`—Shows the number of TCP connection requests from a remote host.
  - `failed`—Shows the number of TCP connections attempted but failed.
  - `streams are currently open`—Shows the current number of established TCP connections.
  - `forcibly closed on us`—Shows the number of connection resets received from remote hosts.
  - `we forcible closed`—Shows the number of resets sent to remote hosts. Resets can be sent manually using the **reset** subcommand.

- `ID`—Transfer Control Block (TCB) identification number for each active or passive connection.
- `Rcv-Q`—Shows the number of items in the receive queue.
- `Send-Q`—Shows the number of items in the send queue.
- `Local socket`—Shows the local IP address and port number.
- `Remote socket`—Shows the remote IP address and port number.
- `State`—Indicates whether TCP sockets are active or passive. Active sockets initiate connections with passive sockets. Passive sockets are denoted in the status table by listen (S) listed as their state. The table also denotes whether a TCP connection has been established by listing the state as *Established* or whether a connection is being attempted by listing *SYN sent*. Sockets can also be listed with *FIN wait* or *Time wait*.

SEE ALSO

ip

A.43 TELNET

NAME

telnet - Connects to another node using the Telnet protocol.

SYNOPSIS

telnet *hostid* [*port*]

DESCRIPTION

The **telnet** command allows you to connect from your local Wireless Router to a remote host using the Telnet protocol.The *hostid* argument specifies the host to which you want to connect, and must be either the DNS name or the IP address in four-part dotted decimal format. The optional *port* argument specifies the port number you want to connect through. If you do not specify a port number, the standard Telnet port of 23 is used.

The Telnet connection appears in a new session on your Wireless Router. To switch between sessions, enter **Ctrl-w**. To close a Telnet connection and end the session, use the **close** command from the main session on your Wireless Router.

OPTIONS

None.

EXAMPLES

The following example opens a Telnet connection to the remote host with the IP address `192.168.2.3`:

```
TAL> telnet 192.168.2.3
```

SEE ALSO

abort, **close**, **kick**, **session**

A.44 TIME

NAME

time - Sets or displays the time on the Wireless Router.

SYNOPSYS

time [*hh***:***mm***:***ss*]

DESCRIPTION

The **time** command allows you to reset the time on your Wireless Router. Specify the time in 24-hour format, separating the hour, minutes, and seconds with colons. For example, 5:08 in the afternoon would be **17:08**.

To display the current time, enter the **time** command without any arguments.

OPTIONS

None.

EXAMPLES

The following example resets the time on the Wireless Router to `9:23` in the morning:

```
TAL> time 9:23
```

The following example shows that the currently configured time is 5:28 and 23 seconds in the afternoon (expressed in 24-hour time as `17:28:47`).

```
radio> time
Time is: 17:28:47
```

SEE ALSO

version

A.45 TIP

NAME

tip - Allows you to configure a modem directly from the Wireless Router.

SYNOPSYS

tip *device-name*

DESCRIPTION

Use the **tip** command to connect to an asynchronous device and create a terminal session (TIP session) on that device. The argument *device-name* specifies the device, and is the symbolic name assigned to the device with the **device make** command. By opening a TIP session on the COM1 or COM2 port, you can issue commands to anything connected to that port. For example, you can use the **tip** command to configure a modem directly from the Wireless Router.

Before you open a TIP session, you must have created the device using the **device create** command. Also, the device must not currently be in use (for example, for PPP connections on the COM2 port or for the service console on the COM1 port). Check for the following lines in the configuration file:

device config *device-name* [*global-config*] **up**

iface config *iface-name* **device** *device-name* [*global-config*] **up**

If either of these lines exist, take down the device with the corresponding **device** or **iface** command:

device config *device-name* **down**

iface config *iface-name* **down unlink**

During the TIP session, type the following string to initialize the modem and turn on auto-answer mode. When you send this string, most modems also automatically set the speed, flow control, and other parameters to match the parameters configured on the device. Make sure these parameters are set on your router before you begin the TIP session.

ATS0=1

During the TIP session, you can send any standard modem configuration commands to the modem. To end a TIP session, type **Ctrl-C**.

DIAGNOSTICS

Device <device> unknown

The device specified does not exist on the system.

Device <device> no async port

The device specified is not an asynchronous device.

Device already in use

The device is currently being used; for example, for a PPP connection or as a login session.

SEE ALSO

device, **iface**, **reset**

A.46 TRACE

NAME

trace - Controls intelligent decoding of packet contents as they enter or leave your Wireless Router.

SYNOPSIS

trace [*iface*]

trace *iface trace-flag* [*filename*]

DESCRIPTION

The **trace** command allows you to decode data packets as they enter or leave an interface on your Wireless Router. The output of **trace** commands appears in a separate session on your service console; use **Ctrl-w** to switch between sessions. Trace information varies depending on the type of packet, but might include the type, source, destination, and size of the packet. Use the **trace** command to debug errant network devices.

The *iface* argument specifies a particular interface on which you want information.

The *trace-flag* argument must be preceded by an *iface* argument. This argument pair starts or stops decoding packets for that particular interface, and further defines the information to include. To issue more than one trace flag, you must repeat the **trace** command. Placing a hyphen (-) before a flag causes the opposite action for that flag to happen. See the section TRACE-FLAG OPTIONS for valid trace flags.

The *filename* argument specifies that the trace information should be written to the specified file as well as appear in a separate console session.

If you do not specify an argument, the **trace** command shows tracing status on all interfaces.

TRACE-FLAG OPTIONS

off	Turns off all packet decoding for this interface.
input	Decodes incoming data packets (packets entering the router via the specified interface).
-input	Stops decoding incoming data packets on the specified interface.
output	Decodes outgoing data packets (packets leaving the router via the specified interface).
-output	Stops decoding outgoing data packets on the specified interface.

broadcast Decodes data packets containing a broadcast address. To use this flag, you must already have specified whether decoding occurs on incoming or outgoing packets.

-broadcast Stops decoding data packets containing a broadcast address. To use this flag, you must already have specified whether decoding occurs on incoming or outgoing packets.

raw Turns off all intelligent packet decoding and dumps raw data as hexadecimal bytes. To use this flag, you must already have specified whether decoding occurs on incoming or outgoing packets.

-raw Turns on intelligent packet decoding (the default state for all **trace** commands). To use this flag, you must already have specified whether decoding occurs on incoming or outgoing packets.

ascii In addition to decoding the data packet, display its contents (if printable) as ASCII characters. To use this flag, you must already have specified whether decoding occurs on incoming or outgoing packets.

-ascii Turns off ASCII packet decoding. To use this flag, you must already have specified whether decoding occurs on incoming or outgoing packets.

hex In addition to decoding the data packet, display its contents (if printable) as hexadecimal bytes. To use this flag, you must already have specified whether tracing occurs on incoming or outgoing packets.

-hex Turns off hexadecimal packet decoding. To use this flag, you must already have specified whether tracing occurs on incoming or outgoing packets.

EXAMPLES

The following example specifies that you want to decode all packets entering the Wireless Router on interface `ether0`. In addition to displaying this information in a service console session, you want the information to appear in a file called `trace.dat`. Finally, you want the contents to display as ASCII characters.

```
TAL> trace ether0 input trace.dat
TAL> trace ether0 ascii
```

The following example shows the current trace status on your Wireless Router. All incoming packets on interface `ether0` are being decoded. The information will appear in ASCII format in a separate console session and in a file called `trace.dat`. Tracing is turned `off` on interface `radio0`.

```
TAL> trace
ether0: input (ASCII dump) trace file: trace.dat
radio0: tracing off.
```

DIAGNOSTICS

Interface <*interface name***> unknown.**

The interface name provided could not be located on the router.

Can't write to <*erroneous file name***>.**

The specified file could not be opened. The filename might be invalid, or there might not be enough room available to store data.

SEE ALSO

session

A.47 UDP

NAME

udp - Shows statistics relating to the User Datagram Protocol (UDP).

SYNOPSIS

udp status

DESCRIPTION

The **udp** command displays statistics relating to UDP. UDP is an unreliable connectionless transport-layer protocol that transports data from one application to another. UDP relies on IP to transmit datagrams across the network but includes its own checksums and port numbers. For sample output, see the EXAMPLES section.

EXAMPLES

```
TAL> udp status
UDP Statistics:
Packets: 5 in, 6 out, 0 errors, 51 sent to unknown socket.

    &UCB Rcv-Q  Local socket
912f0010     0  0.0.0.0:161
```

Where:

- `&UCB`—Shows UDP Control Block (UCB) addresses.
- `Rcv-Q`—Shows the number of items in the receive queue.
- `Local socket`—Shows the local IP address and port number.

SEE ALSO

ip

A.48 UPTIME

NAME

uptime - Determines the length of time since the Wireless Router last booted.

SYNOPSYS

uptime

DESCRIPTION

Use the **uptime** command to determine the length of time since your Wireless Router was last booted. A very short uptime (for example, a few seconds) might indicate that the system is rebooting unexpectedly.

EXAMPLE

In the following example, the Wireless Router has been running for 24 days, 5 hours, and 51 seconds:

```
TAL> uptime
System up 24 days, 5 hours, 51 seconds
```

A.49 USER

NAME

user - Manages the user table.

SYNOPSIS

user *option*

DESCRIPTION

The **user** command allows you to establish or modify a list of users who have access to the router. You can define the following levels of authorization:

- FTP—To allow a user to read, write, and overwrite files from an FTP connection, use an authorization flag of 0x07. You must also specify the root directory.
- Operator—To allow an operator to interactively modify the configuration without changing the configuration file and rebooting the router, use an authorization flag of 0x10. If a user with operator privileges modifies the configuration interactively, changes are not saved when the router reboots.
- PPP—To allow the user to connect to the Wireless Router using the Point-to-Point Protocol (PPP), use an authorization flag of 0x20. You must also specify the interface on which the user can make PPP connections.
- System administrator—To provide FTP and operator privileges, and also allow the user to log in through the service console, use an authorization flag of 0x47. You must also specify the root directory. If you define one user with administrator privileges, all users will be prompted for a username and password when they attempt to access the Wireless Router through the service console. If you do not define any system administrator users, any user can access the Wireless Router through the service console.

OPTIONS

config Changes options for a particular user including authorization, interface, password, or root directory.

user config *name* [**authorization** *flag*] [**iface** *iface*] [**password** *pswd*] [**root** *dir*]

For a description of keywords and arguments, see the **user create** command.

create Adds a user to the user table and assigns user options, including authorization, interface, password, and root directory. For sample output, see the EXAMPLES section.

user create *name* [**authorization** *flag*] [**iface** *iface*] [**password** *pswd*] [**root** *dir*]

Where:

- *name*—Is the username of the user you are adding.
- **authorization** *flag*—Establishes the level of access privileges. If you do not specify an authorization level, the user has all levels of access when logging in through the console, but no level of access when logging in remotely.
- **iface** *iface*—Specifies a particular interface. Specify the interface if you are going to allow the user to make PPP connections.
- **password** *pswd*—Specifies a user password. Specify the root directory if you are going to allow the user to connect to the router using FTP.
- **root** *dir*—Indicates the root directory used for FTP transfers.

delete

Removes a user and that user’s options from the user table.

user delete *name*

Where:

- *name*—Is the username of the user you are removing.

kill

Removes a user and that user’s options from the user table. Identical to the **user delete** command.

user kill *name*

make

Adds a user to the user table and assigns user options, including authorization, interface, password, and root directory. Identical to the **user create** command.

user make *name* [**authorization** *flag*] [**iface** *iface*] [**password** *pswd*] [**root** *dir*]

show

Displays statistics about the user or user table to a standard output device, such as the console. For sample output, see the EXAMPLES section.

user show {*name* | **all**} [**verbose**]

Where:

- *name*—Is the name of the user about whom you want to display statistics.
- **all**—Requests statistics about all users in the user table.
- **verbose**—Requests more detailed information and statistics.

EXAMPLES

The following example shows the current table of authorized users. The column labeled `Username` displays the username, and the column labeled `Password` displays the password of each user. The column labeled `Iface` displays specific interfaces that user has privileges on. The column labeled `Auth` displays the level of authorization. For example, user `Betsy` is authorized to make PPP connections (`0x20`) on the `ppp0` interface.

```
TAL> user show all
Username  Password  Iface     Auth Root
Jim       MrNull              0x10
Betsy     casco     ppp0      0x20
```

In the following example, the first command assigns system administrator privileges (`0x47`) to the user `Silva`. The second command displays the new table. Because a user with system administrator privileges can also access the router using FTP, the command specifies a root directory.

```
TAL> user create silva authorization 0x47 password nikki

TAL> user show all
Username  Password  Iface     Auth Root
Jim       MrNull              0x10
Betsy     casco     ppp0      0x20
Silva     nikki               0x47 /
```

A.50 VERSION

NAME

version - Lists the current version of TALnet software.

SYNOPSIS

version

DESCRIPTION

The **version** command displays the current version of TALnet software and statistics about that software. If you experience problems with your Wireless Router, identify the version of software before contacting your technical support representative.

OPTIONS

None.

EXAMPLES

The following example lists the current version of TALnet software as Release 1.2, compiled on February 23, 1996:

```
TAL> version
TALnet Release 1.2
Copyright 1992-1996 Tetherless Access Limited, All Rights Reserved.
Compiled at TAL: Feb 23 1996, 12:23:05
Router platform: SubSpace 2001 (rev1)
Memory Usage:    Library=41K, Code=389K, Data=36K
```

SEE ALSO

time

A.51 VIEW

NAME

view - Displays the contents of a file.

SYNOPSIS

view *filename*

DESCRIPTION

The **view** command allows you to view the contents of a file and scroll back and forth as necessary through the file. The *filename* argument is the file you want to view. After you issue the **view** command, you can use the key sequences in Table A-5 to navigate through the file.

Table A-5 Navigating through a File View

Key	Function
h or **H** or **<**	Move to the beginning of the file.
e or **>**	Move to the end of the file.
d or **Ctrl-N** or **j**	Move down one line.
u or **k**	Move up one line.
Ctrl-D	Move down half a screen.
Ctrl-U	Move up a half screen.
D or **Ctrl-V**	Move down one screen.
U or **v**	Move up one screen.
Ctrl-C or **q** or **Q**	Exit the file.

OPTIONS

None.

EXAMPLES

In the following example, you request to view the contents of the file `talnet.cfg`:

```
TAL> view talnet.cfg
```

A.52 WHO

NAME

who - Lists users who are currently logged on to the Wireless Router.

SYNOPSIS

who

DESCRIPTION

The **who** command lists users who are currently logged on to the Wireless Router, including users who are logged on directly through the service console and users who are logged on through Telnet connections. The output includes the username, the type of connection, and the length of time the user has been logged on. If the user is connected through Telnet, the output also lists the IP address and the TCP port number of the user's host machine.

OPTIONS

None.

EXAMPLES

In the following example, two users are connected to the Wireless Router: `admin` and `service`. User `admin` is connected directly through a service console, and has been connected for `1` day, `9` hours, `26` minutes, and `5` seconds. User `service` is connected remotely using Telnet, and has been connected for `4` hours, `59` minutes, and `17` seconds. The IP address of the host machine from which user `service` is connected is `204.94.180.135`. The TCP port number is `1537`.

```
TAL> who
#  user        time-on     type    source
0  admin       1+09:26:05  console
1  service     04:59:17    telnet  204.94.180.135:1537
```