

Chapter 16

Synchronous Tunnelling

Introduction	16-2
Synchronous Tunnelling on the Router	16-2
Configuration Example	16-3
Command Reference	16-4
add stt	16-5
delete stt	16-6
reset stt	16-6
set stt	16-7
show stt	16-8

Introduction

This chapter describes the synchronous tunnelling service provided by the router, and how to set up and use synchronous tunnelling on the router. The synchronous tunnelling module is called STT, which is an acronym for Synchronous Tunnelling over TCP.

In a typical network, routers carry routable protocols from one local area network to another, either directly or over wide area links. In some situations there is a requirement to carry protocols that are non-routable and that are carried over synchronous lines. Synchronous tunnelling is a mechanism for carrying these protocols through a network of routers over another, routable, protocol. The word *tunnelling* refers to the fact that the traffic is passed from one point in the network to another, through a *tunnel* between two routers.

An example of a situation where synchronous tunnelling might be used is a network that consists of terminal servers communicating over wide area links. The management of the network decides to upgrade the network so that LAN-to-LAN routing can take place. However, the terminal servers must remain in operation until the entire network has been converted. The ability of the router to perform synchronous tunnelling means that the network conversion can take place in stages. As a router is placed in a remote site, the wide area link that served the site is used for the router. The terminal server can still be used if an extra synchronous port is available at both the remote and central sites. These ports are set to tunnel the synchronous traffic and as far as the terminal server is concerned, the link still exists.

Synchronous tunnelling is normally a proprietary option in routers. No standards exist for tunnelling generalised synchronous data through a network.

Synchronous Tunnelling on the Router

The router supports a form of synchronous tunnelling. Since synchronous tunnelling is proprietary, the router cannot tunnel to any other brand of router. The router can only tunnel synchronous traffic between devices that communicate using HDLC. Synchronous tunnel traffic is carried over a TCP connection. This provides a reliable path for the tunnel but adds a considerable amount of overhead for the data being carried.

The STT module allows traffic on a spare synchronous port on one router to be carried to a spare synchronous port on another router. The two routers must be capable of forming a TCP connection, so must be attached to the same internet. Since synchronous tunnelling is normally a function internal to a particular organisation, this prerequisite should be readily obtainable.

Configuration Example

The following example shows how to configure a simple synchronous tunnel. A synchronous tunnel must be configured on both routers at the ends of the synchronous tunnel. A synchronous tunnel, called "Testing", is to be created between routers AC1 and AC2. The IP address of AC1 is 172.16.9.51 and that of AC2 is 172.16.9.52. The synchronous port is used for the tunnel on AC1 is Syn 1 and on AC2, Syn 2. The maximum frame length supported by the tunnel is 300 octets (or bytes). Figure 16-1 shows a basic synchronous tunnel and Table 16-1 lists the required parameters and values.

Figure 16-1: Basic synchronous tunnel

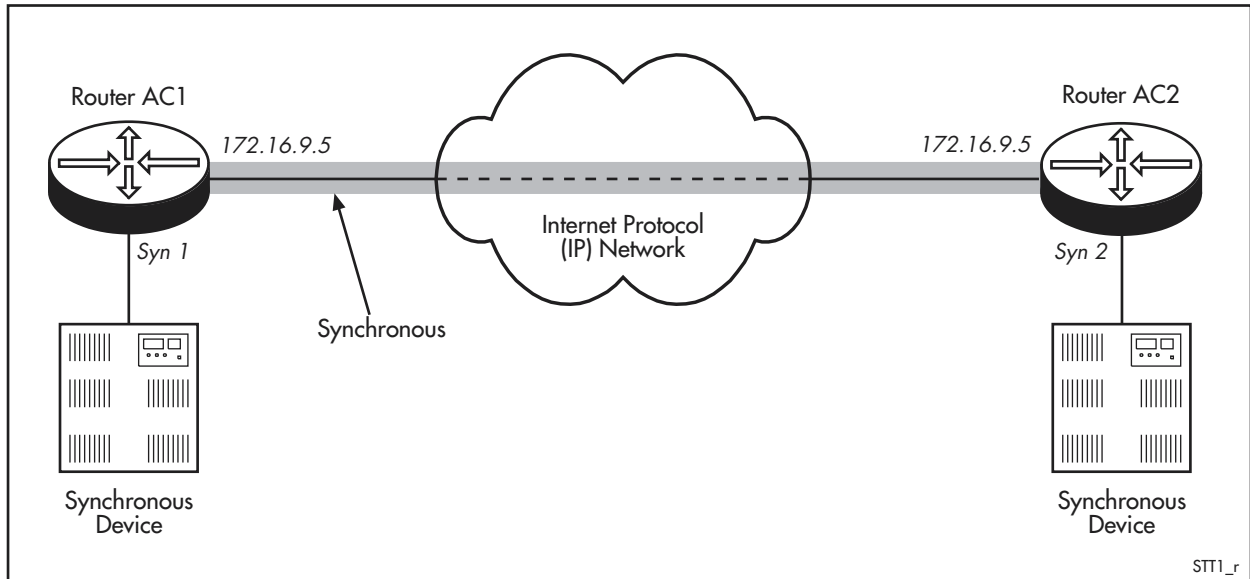


Table 16-1: Example configuration parameters for a basic synchronous tunnel

Parameter	Router AC1	Router AC2
Synchronous device connected to port	Syn 1	Syn2
Name of synchronous tunnel	Testing	Testing
Maximum frame length (octets)	300	300
IP Address of TCP/IP link	172.16.9.51	172.16.9.52

To configure the synchronous tunnel

1. **Create the synchronous tunnel on the routers at each end of an existing TCP/IP link.**

The synchronous tunnel must be added to both routers. On router AC1 the command is:

```
add stt=test lsyn=1 rsyn=2 ip=172.16.9.52 datalen=300
```

On router AC2 the command is:

```
add stt=testing lsyn=2 rsyn=1 ip=172.16.9.51 datalen=300
```

2. Verify that the synchronous tunnel is operational.

To check that the synchronous tunnel is operational, use the command:

```
show tcp
```

Verify that a TCP connection exists between routers AC1 and AC2, and that one of the ports has the value 5026. Port 5026 is the port that the router uses to listen for attempts to establish the synchronous tunnel. This port is closed by default until the **add stt** command is executed. The port then remains open until a router restart.

3. Connect the synchronous devices.

The devices that are to use the tunnel can now be connected to the correct synchronous port on the local router. Test the end-to-end connectivity by using the synchronous devices to send test data across the synchronous tunnel.

Command Reference

This section describes the commands available on the router to configure and manage synchronous tunnelling. Synchronous tunnelling requires the IP module to be enabled and configured correctly. See [Chapter 21, Internet Protocol \(IP\)](#) for detailed descriptions of the commands required to enable and configure IP.

The shortest valid command is denoted by capital letters in the Syntax section. See [“Conventions” on page lxiv of About this Software Reference](#) in the front of this manual for details of the conventions used to describe command syntax. Note that the = sign is required in the command syntax.

See [Appendix A, Messages](#) for a list of messages and their meanings.

add stt

Syntax `ADD STT=sttname LSYN=localsyn RSYN=0..7 IP=ipadd
[DATALEN=maxlength]`

where:

- *sttname* is the name of the synchronous tunnel, and may be 1 to 15 characters long. Valid characters are all alphabetic and numeric characters. The name is not case-sensitive, meaning that “tunnelX” and “TUNNELx” are equivalent.
- *localsyn* is the number of a synchronous port in the range 0 to the highest available synchronous port on the router, expressed as a decimal number.
- *ipadd* is an IP address in dotted decimal notation.
- *maxlength* is the maximum frame length supported by the synchronous devices using the tunnel from 6 to 1650 expressed as a decimal number.

Description This command adds a synchronous tunnel to the router.

The **lsyn** parameter specifies the synchronous port on the local router where a synchronous device is connected.

The **rsyn** parameter specifies the synchronous port on the remote router where a synchronous device is connected.

The **ip** parameter specifies the IP address, in dotted decimal notation, of the router at the remote (other) end of the synchronous tunnel.

The **dataLEN** parameter specifies the maximum frame length supported by the synchronous devices using the tunnel. The default is 256.

If the command is successful, the synchronous tunnel is started. Details of the synchronous tunnel just defined are shown in the same format as in the [show stt command on page 16-8](#).

Examples To create a synchronous tunnel called “MUX” to connect a synchronous device on synchronous port 1 of the local router to a synchronous device on synchronous port 4 of the remote router with an IP address of 172.16.9.35, use the command:

```
add stt=mux lsyn=1 rsyn=4 ip=172.16.9.35
```

Related Commands [show stt](#)

delete stt

Syntax DELEte STT=*sttname*

where *sttname* is the name of the synchronous tunnel, and may be 1 to 15 characters long. Valid characters are all alphabetic and numeric characters. The name is not case-sensitive, meaning that “tunnelX” and “TUNNELx” are equivalent.

Description This command deletes a synchronous tunnel that has been previously defined with the [add stt command on page 16-5](#). If the command is successful, the synchronous tunnel is stopped and then deleted.

Examples To delete a synchronous tunnel called “MUX”, use the command:

```
del stt=mux
```

Related Commands [reset stt](#)
[show stt](#)

reset stt

Syntax RESET STT=*sttname*

where *sttname* is the name of the synchronous tunnel, and may be 1 to 15 characters long. Valid characters are all alphabetic and numeric characters. The name is not case-sensitive, meaning that “tunnelX” and “TUNNELx” are equivalent.

Description This command resets the synchronous tunnel.

Examples To reset the synchronous tunnel called “MUX”, use the command:

```
reset stt=mux
```

Related Commands [delete stt](#)
[show stt](#)

set stt

Syntax SET STT=*sttname* [LSYN=*localsyn*] [RSYN=0..7] [IP=*ipadd*]
[DATALEN=*maxlength*]

where:

- *sttname* is the name of the synchronous tunnel, and may be 1 to 15 characters long. Valid characters are all alphabetic and numeric characters. The name is not case-sensitive, meaning that “tunnelX” and “TUNNELx” are equivalent.
- *localsyn* is the number of a synchronous port, in the range 0 to the highest available synchronous port on the router, expressed as a decimal number.
- *ipadd* is an IP address, in dotted decimal notation.
- *maxlength* is the maximum frame length supported by the synchronous devices using the tunnel from 6 to 1650 expressed as a decimal number.

Description This command modifies the characteristics of a synchronous tunnel previously defined with the [add stt command on page 16-5](#).

The **lsyn** parameter specifies the synchronous port on the local router where a synchronous device is connected. The **rsyn** parameter specifies the synchronous port on the remote router where a synchronous device is connected.

The **ip** parameter specifies the IP address, in dotted decimal notation, of the router at the remote (other) end of the synchronous tunnel.

The **dataLEN** parameter specifies the maximum frame length supported by the synchronous devices using the tunnel. The default is 256.

The new characteristics of the synchronous tunnel just modified are shown, in the same format as in the [show stt command on page 16-8](#).

Examples To create a synchronous tunnel called “MUX” to connect a synchronous device on synchronous port 1 of the local router to a synchronous device on synchronous port 4 of the remote router with an IP address of 172.16.9.35, use the command:

```
add stt=mux lsyn=1 rsyn=4 ip=172.16.9.35
```

Related Commands [add stt](#)
[show stt](#)

show stt

Syntax `SHOW STT[=sttname]`

where *sttname* is the name of the synchronous tunnel, and may be 1 to 15 characters long. Valid characters are all alphabetic and numeric characters. The name is not case-sensitive, meaning that “tunnelX” and “TUNNELx” are equivalent.

Description This command displays the status of one or all synchronous tunnels. If a synchronous tunnel is specified, information is displayed about that synchronous tunnel. Otherwise, information about all synchronous tunnels is displayed (Figure 16-2, Table 16-2).

Figure 16-2: Example output from the **show stt** command

Name	Port		IP address	DataLen	State
	Local	Remote			
ALBERT	01	01	172.16.9.56	1650	OPEN
LINCOLN	02	03	172.16.9.45	1650	REQ SENT
toSYNDirectQLen	00000		fromSYNQLen	00000	toTCPQLen
					00000
STT Counters:					
goodOpen	00002		goodTCPUp	00003	goodSYNup
badInstance	00000		noTCB	00000	inTCPbadSCB
localListenOpen	00006		goodTCPlisten	00002	badTCPlisten
TCPcalledIn	00003		TCPnotOpen	00000	TCPnotOpen2
connStrSend	00012				00000
TCPstateFree	00000		TCPstateConFree	00000	TCPstateEstabR
TCPstateCWait	00000		TCPstateClosed	00001	confirmCalled
					00002
synBusy	00000		toSYNDQfull	00000	inSynOverflow
goodSYNin	00007		goodTCPout	00007	noSYNtoOut
sttHead:Length	00000		sttHead:Version	00000	00014

Table 16-2: Parameters in the output of the **show stt** command

Parameter	Meaning
Name	Name of the synchronous tunnel.
Local/Remote Port	Local and remote synchronous ports.
IP Address	IP address of the router at the remote end of the tunnel.
DataLen	Maximum frame length supported by the tunnel.
State	Current state of the TCP connection used by the tunnel.
toSYNDirectQLen	Length of the queue holding packets to be sent to the synchronous device.
fromSYNQLen	Length of the queue of packets received from the synchronous device.
toTCPQLen	Length of the queue of packets to be sent to the TCP connection.
goodOpen	Number of times an STT has entered the OPEN state.
goodTCPUp	Number of times an STT's TCP connection has come up.

Table 16-2: Parameters in the output of the **show stt** command (cont.)

Parameter	Meaning
goodSYNup	Number of times an STT's connection with the synchronous device has come up.
badInstance	Number of times a packet was received from the synchronous device for a nonexistent STT.
noTCB	Number of times an STT tried to use a nonexistent TCP connection.
inTCPbadSCB	Number of times a packet was received from TCP for a nonexistent STT.
localListenOpen	Number of successful opens of the TCP listen port for all STTs.
goodTCPlisten	Number of successful opens of TCP listen connections.
badTCPlisten	Number of failed opens of TCP listen connections.
TCPcalledIn	Number of connect strings received from the remote router.
TCPnotOpen	Number of packets sent to a TCP connection not open.
TCPnotOpen2	Number of TCP packets received while a TCP connection is not open.
connStrSend	Number of times a connect string has been sent to the remote router.
TCPstateFree	Number of times an STT's TCP connection has entered the FREE state.
TCPstateConFree	Number of times an STT's TCP connection has entered the CONNECT FREE state.
TCPstateEstabR	Number of times an STT's TCP connection has entered the ESTABLISHED state.
TCPstateCWait	Number of times an STT's TCP connection has entered the CLOSE WAIT state.
TCPstateClosed	Number of times an STT's TCP connection has entered the CLOSED state.
confirmCalled	Number of packets confirmed by the SYN module.
synBusy	Number of times the synchronous port was busy when requested.
toSYNDQfull	Number of TCP packets discarded.
inSynOverflow	Number of packets from the synchronous device that were discarded.
goodSYNin	Number of packets successfully received from the synchronous device.
goodTCPout	Number of packets successfully sent to the remote router.
noSYNtoOut	Number of times there was no packet to send to a synchronous device.
sttHead:Length	Number of packets received with corrupted header length fields.
sttHead:Version	Number of packets received with corrupted header version fields.

Related Commands [add stt](#)

