Tunneling Async Protocols in BSTUN Configuration Example

Document ID: 41983

Contents

Introduction

Dedicated and native async protocols are not directly supported with any Cisco implementation. However, Block Serial Tunnel (BSTUN) async−generic tunneling can provide limited ability to tunnel this data.

Prerequisites

Requirements

There are no specific requirements for this document.

Components Used

The information in this document is based on the software and hardware versions:

- Use Feature Navigator II (registered customers only) , and use the **Search by Feature** option.
- Use Software Advisor (registered customers only) to search for the minimum supported software release needed for your hardware.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Conventions

Refer to Cisco Technical Tips Conventions for more information on document conventions.

Background Information

Async protocols such as Diebold's TC500 to communicate to money ATMs or tunneling HyperTerminal from a PC to another PC have no direct support or implementation in the Cisco IOS ®. As the name implies, this is a generic implementation that has some capability to carry this type of data. This is known as BSTUN async−generic, and requires the IBM or the Enterprise IOS feature set.

BSTUN async−generic was originally designed to carry unidirectional, small packets from security devices to a reporting device. BSTUN async−generic, however, can carry interactive traffic. In essence, this implementation attaches to native, async devices and receives the data into the serial interface and then into a memory buffer. Periodically, the buffered data is then encapsulated into a TCP packet and sent to the BSTUN peer where it is decapsulated and sent to the async device attached at the remote site.

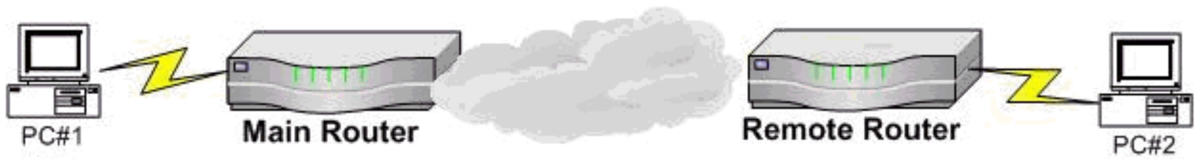
BSTUN async−generic is a simplistic operation. The router has no capability to be configured to have knowledge of the start of frame (SOF), the end of frame (EOF), or the addressing schema of the async protocol. If the address portion of the frame is in every frame, is one byte long, and is the same place in the frame, then the **asp address−offset** command can be issued to specify to the router where to find the address in the frame, as exampled later in this document. In many situations, however, there will not be a address portion contained into the protocol. Having no knowledge of the async protocol construction means that the router is unable to discern individual packets from others if they are not separated by a time period. Approximately 40 ms is required between frames at 9600 bits/sec to provide the router adequate amount of time to properly discern one packet from another. The router simply sees a data stream into it's serial interface and then wraps this data into TCP. There is no possibility the router is able make routing decisions based upon any individual aspect of the incoming frame. Thus, BSTUN async−generic must be physically designed so only one device attaches to the router serial interface. There is no local−acknowledgement feature. BSTUN supports local−ack for IBM3270 BISYNC protocol only.

Configure

In this section, you are presented with the information to configure the features described in this document.

Network Diagram

This document uses the network setup shown in this diagram.



Both PCs use Microsoft's HyperTerminal or in place of one of the PCs could be a connection into the console port of a Cisco router. These sample configurations represent configurations implemented from routers not previously configured in a lab scenario, and show the relevant portions of the configuration needed. These are configured assuming a 9600 bit/sec, 8N1 connection.

Configurations

This document uses the configurations shown in this section.

- Main router (Cisco 1700 Router)
- Remote Router (Cisco 3640 Router)
- Main router (Cisco 3600 Router)
- Remote #1 (Cisco 1700 Router)

- Remote #2 (Cisco 1700 Router)

Main router (Cisco 1700 Router)

```
main#show running-config
Building configuration...
.
.
.
ip subnet-zero
bstun peer-name 10.1.1.1
bstun protocol-group 1 async-generic
interface loopback0
   ip address 10.1.1.1 255.0.0.0
interface serial0
   physical-layer async
   encapsulation bstun
   asp role secondary
   bstun group 1
   bstun route all tcp 30.1.1.1
interface serial1
   ip address 20.1.1.1 255.0.0.0
ip route 0.0.0.0 0.0.0.0 20.1.1.2
line 1
   speed 9600
   databits 8
   parity none
   stopbits 1
.
.
.
!
end
```

Remote Router (Cisco 3640 Router)

```
REMOTE#show running-config
Building configuration...
bstun peer-name 30.1.1.1.
bstun protocol-group 1 async-generic
interface loopback 0
   ip address 30.1.1.1
interface ethernet1/0
   shutdown
interface serial 2/0
   physical-layer async
   encapsulation bstun
   asp role primary
   bstun group 1
   bstun route all tcp 10.1.1.1

interface serial 2/1
   ip address 20.1.1.2 255.0.0.0
ip route 0.0.0.0 0.0.0.0 20.1.1.1
line 65
   speed 9600
   parity none
   databits 8
   stopbits 1
.
.
!
end
```

Note: When you issue the **physical−layer async** command on the serial interface, a TTY line is assigned to the serial interface. This TTY line definition is where the databits, stopbits, parity, and speed are configured. This is the formula to determine which line corresponds with which serial interface.

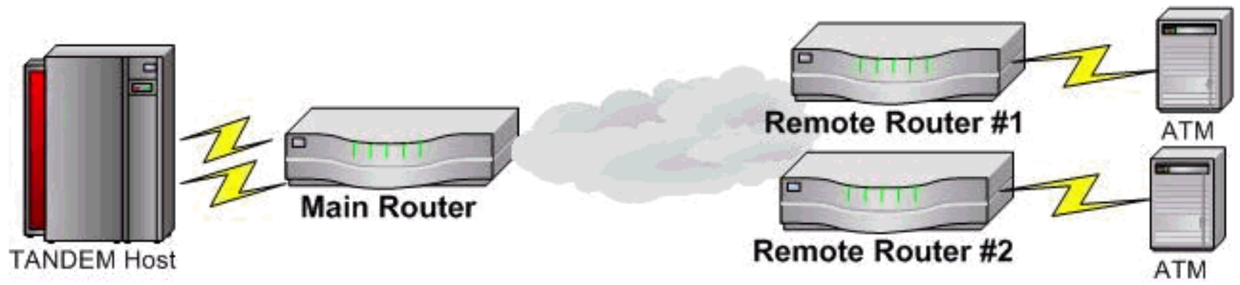
line#=(slot# x 32) + interface# + 1

The show line in the Remote router configuration output indicates in the far right column the corresponding line number. Serial2/0 is represented by line 65 and the physical definitions for this link are configured under line 65

```
REMOTE#sh line
   Tty Typ     Tx/Rx        A Modem  Roty AccO AccI   Uses   Noise  Overruns   Int
*    0 CTY                       −        −          −        −        −          0
    65 TTY   9600/9600  −         −          −        −        −      0          0
   129 AUX 9600/9600  −        −          −        −        −       0          0
   130 VTY                     −        −          −        −        −          0
   131 VTY                     −        −          −        −        −          0
   132 VTY                     −        −          −        −        −          0
   133 VTY                     −        −          −        −        −          0
   134 VTY                     −        −          −        −        −          0

Line(s) not in async mode −or− with no hardware support:
1−64, 66−128
```

In this scenario, a Tandem communicates with remote ATM devices. In this sample configuration, the async protocol runs a 4800 7E2 protocol and the Main router connected to the TANDEM is a 3600 series router to remote 1700 series routers. See this network diagram.



Main router (Cisco 3600 Router)

```
main#show running−config
Building configuration...
bstun peer−name 10.1.1.1.
bstun protocol−group 1 async−generic
bstun protocol−group 2 async−generic
interface loopback 0
   ip address 10.1.1.1
interface serial1/0
   encapsulation frame−relay
interface serial 1/0.1 point−to−point
   ip address 20.1.1.1 255.255.255.0
   frame−relay interface−dlci 100
interface serial 1/0.2 point−to−point
   ip address 20.2.1.1 255.255.255.0
   frame−relay interface−dlci 200
interface serial 2/0
   physical−layer async
   encapsulation bstun
   asp role secondary
   bstun group 1
   bstun route all tcp 30.1.1.1

interface serial 2/1
 physical−layer async
   encapsulation bstun
   asp role secondary
   bstun group 2
   bstun route all tcp 30.2.1.1

ip route 30.2.1.0 255.255.0.0 20.2.1.2
ip route 0.0.0.0 0.0.0.0 20.1.1.2
line 65
   speed 4800
   parity even
   databits 7
   stopbits 1
.
line 66
   speed 4800
   parity even
   databits 7
   stopbits 1
.
!
end
```

Remote #1 (Cisco 1700 Router)

```
REMOTE1#show running−config
Building configuration...
bstun peer−name 30.1.1.1
bstun protocol−group 1 async−generic
interface loopback0
   ip address 30.1.1.1 255.255.0.0
interface serial0
   physical−layer async
   encapsulation bstun
   asp role primary
   bstun group 1
   bstun route all tcp 10.1.1.1
interface serial1
   encapsulation frame−relay
interface serial1.1 point−to−point
   ip address 20.1.1.2 255.255.255.0
   frame−relay interface−dlci 100
ip route 0.0.0.0 0.0.0.0 20.1.1.1
line 1
   speed 4800
   databits 7
   parity even
   stopbits 2
.
.
.
!
end
```

Remote #2 (Cisco 1700 Router)

```
REMOTE2#show running−config
Building configuration...
bstun peer−name 30.2.1.1
bstun protocol−group 2 async−generic
interface loopback0
   ip address 30.2.1.1 255.255.0.0
```

```
interface serial0
   physical−layer async
   encapsulation bstun
   asp role primary
   bstun group 2
   bstun route all tcp 10.1.1.1
interface serial1
   encapsulation frame−relay
interface serial1.1 point−to−point
   ip address 20.2.1.2 255.255.255.0
   frame−relay interface−dlci 100
ip route 0.0.0.0 0.0.0.0 20.2.1.1
line 1
   speed 4800
   databits 7
   parity even
   stopbits 2
.
.
.
!
end
```

Verify

There is currently no verification procedure available for this configuration.

Troubleshoot

BSTUN receives a packet into the serial interface, encapsulates it, and sends this TCP packet to the remote router when the **bstun route all tcp** command is issued. The TCP packet is received at the remote router and is decapsulated. The data is sent out on the serial interface. If this connection does not work, the incoming data must first be verified with the **debug asp** packet. You see the data received by the router on the serial interface. Since the router has no protocol construction and varies according to the async protocol, sample debug is not provided. The data stream seen by the router must match what is sent by the device. If it does not match, more than likely, the speed, databits, parity, or stopbits are not configured to match the device. This can be the case as well if no data is received.

If data is received on the serial interface, issue the **show bstun** command in order to show if the connection is open or closed. Open state with only packets trasmitted indicates the TCP is sent to the remote BSTUN peer. At this point, a ping test from the IP address of the local BSTUN peer−name to the remote BSTUN peer−name IP address verifies if IP is configured and working properly. If ping testing is successful, then at the remote, issue the **debug asp packet** command in order to determine if the packet is received and sent onto the serial interface to the async device.

Complete these steps in order to troubleshoot.

1. Verify data is received into the host router with the **debug asp packet** command.
2. Ensure IP connectivity with ping test sourcing pings from the bstun peer−name IP address to the remote IP address of the remote BSTUN peer−name.
3. At the remote, verify that packets are sent to the remote device with the **debug asp packet** command.
4. If the async protocol does have an address contained in the packets sent to the router, it can be beneficial to issue the **asp offset−address** command under the interface with the appropriate byte number corresponding to where the address is contained in the packet. The default value for this is 0. For example, if the packet is 01C1ABCDEF, where C1 is the address, the serial interface can be configured with the **asp offset−address 01** command. In some cases, this allows the router to identify a packet and increases the probability that the router handles the data as a framed packet and not just

as a data stream.

Related Information

- **STUN (Serial Tunnel) & BSTUN (Block Serial Tunnel) Technical Support**
- **Technical Support − Cisco Systems**

Updated: Sep 09, 2005 Document ID: 41983