Teldat Router

IP Tunnel Interface (TNIP)

Doc. *DM519-I* Rev. *8.30*
March, 2000

INDEX

Chapter 1
IP Tunnel Interface (TNIP)

1. Description

1.1. Introduction

This is known as Tunneling, the procedure through which diverse protocol packets are encapsulated in another protocol. This function is implemented through a virtual interface so its configuration is reduced to that of a simple interface: 'Tunnel Interface'. The Tunnel Interface is not previously linked to encapsulation, transport nor predetermined protocol. This is an architect providing the necessary services to implement any standard encapsulation scheme. As the tunnels are end to end links, you must configure independent tunnels for each link.

Tunneling possesses three components:

- Payload protocol or transport protocol: this is the protocol being encapsulated (IP or SRT).
- Carrier Protocol: This is the protocol which encapsulates.
  - ◊ Generic Routing Encapsulation (GRE).
- Transport protocol, delivery protocol: This is the protocol which transports the encapsulated payload protocol (Only IP).

1.2. Advantages of tunneling

There are various situations where encapsulating one protocol in another is useful:

- To interconnect multiprotocol local networks through a backbone with a single protocol.
- To resolve interconnection problems for networks containing protocols with a limited number of hopes and without this procedure cannot connect.
- To connect two non consecutive subnets.
- To permits Virtual Private Networks throughout the WAN networks.

1.3. Special considerations

The following points describe consideration and precautions that should be borne in mind when configuring tunnels:

- Encapsulation and desencapsulation produced at the tunnel ends are slow operations.
- You must take care when configuring and bear in mind the possible security and topology problems. E.g. you could configure a tunnel whose source and destination are not restricted by Firewalls.
- You must correctly select the methods through which the tunnel will go. Cases could arise such as crossing 'Fast FDDI' networks and slow links of 9600 bauds. Some payload protocols do not behave well in networks composed of mixed methods.
- Many end to end tunnels can saturate the link with routing information.

- Some routing protocols decide the best route solely based on the number of hops preferring the tunnel even if a better route exists. The tunnel always appears to be one hope even though the cost may be higher.
- A worse problem that can occur is if the routing information on the networks connected through the tunnel gets mixed up with the information on the networks transporting this information. In these cases, the best route towards the ‘tunnel destination’ is through the tunnel. This type of route is known as ‘recursive route’ and provokes a temporary fall in the tunnel. In order to avoid this problem, you must maintain the routing information independently;
  - ◊ Using an AS number or distinct Tag.
  - ◊ Using a different routing protocol.
  - ◊ Using static routes for the first hop (but taking care with the route loops).

2. Structure of the encapsulated frame

In the case of IP tunnel, the transport protocol or delivery is IP therefore the structure of the encapsulated frame is the following:

Delivery protocol header: IP
Encapsulation protocol header
Payload protocol packet

2.1. IP over IP with GRE

In this case the encapsulated controlling protocol is GRE. And the payload protocol or the protocol being encapsulated is IP. The encapsulated protocol GRE (Generic Routing Encapsulation) is described in the RFC1701 and this particular case of IP over IP with GRE in the RFC1702.

Delivery protocol header: IP
Encapsulation protocol header: GRE
Payload protocol: IP

The IP header is beyond the limits of this document although not the GRE header. The GRE header has the following form:

```
0                                       1                                       2                                       3 1
0   1   2   3   4   5   6   7   8   9   0   1   2   3   4   5   6   7   8   9   0   1   2   3   4   5   6   7   8   9   0 1
```

C	R	K	S	s	Recur	Flags	Ver	Protocol Type
Checksum (optional)								Offset (optional)
Key (optional)								
Sequence Number (optional)								
Routing (optional)								

Checksum Present (bit 0) (C)

If the Checksum Present bit is set to 1, then the checksum field is present and contains valid information. If either the Checksum Present bit or the Routing Present bit are set, both the Checksum and Offset fields are present in the GRE packet.

Routing Present (bit 1) (R)

Not used.

Key Present (bit 2)

If the Key Present bit is set to 1, then the key field (or identifier) is present in the packet and contains a valid value.

Sequence Number Present (bit 3)

If the Sequence Number Present is set to 1, then the Sequence Number field is present and contains a valid value.

Strict Source Route Present (bit 4)

Not used.

Recursion Control (bits 5-7)

Recursion control contains a three bit unsigned integer which contains the number of additional encapsulations which are permissible. This is always zero.

Version Number (bits 13-15)

Always 0.

Protocol Type (2 octets)

Contains the payload protocol type.

Offset (2 octets)

The offset field indicates the octet offset from the start of the Routing field to the first route to be examined.

Checksum (2 octets)

Contains the IP checksum of the GRE header and the payload packet.

Key (4 octets)

Tunnel identifier.

Sequence Number (4 octets)

This is the Number used by the receiver to establish the correct order in which packets arrive.

Routing (variable length)

Does not exist.

When IP is encapsulated in IP using GRE, the TOS and the IP security options are copied from the payload protocol header to the delivery protocol header. The TTL however does not copy but establishes a default value used by the IP in order to prevent the RIP packets traveling through the tunnel timing out before reaching their destination.

2.2. IP adaptation over Internet

For further information on IP adaptation over Internet, see chapter 6.

2.3. IP over SRT with GRE

Again the encapsulation controlling protocol is GRE. In this case the payload protocol or protocol being encapsulated is SRT.

The GRE header fields are filled in and interpreted in the same way.

The TTL, TOS and the security options in the delivery protocol header are used by default in IP.

3. References

RFC-1701: Generic Routing Encapsulation (GRE), S. Hanks, October-1994

RFC-1702: Generic Routing Encapsulation over IPv4 networks, S. Hanks, October-1994

Chapter 2
IP tunnel interface configuration (TNIP)

1. Creating an IP tunnel Interface (TNIP)

In order to create an IP tunnel interface, you need to enter:

```
Config>ADD DEVICE TNIP
Added TNIP interface with num: 2

TNIP tunnel encapsulated type:[GRE]?
Config>
```

To subsequently access the configuration, simply enter:

```
Config>NETWORK 2
```

The protocol supported over the TNIP interface is IP. In order to activate the IP over the TNIP interface you need to assign an IP address to the selected interface or configure it as an interface without an assigned number. In order to carry this out you need to access the IP protocol configuration.

Example with a known IP address:

```
*P 4
Config>PROTOCOL IP
Internet protocol user configuration
Conf IP>ADD ADDRESS
Which net is this address for[0]? 2
New address [0.0.0.0]? 5.5.5.1
Address mask[255.0.0.0]? 255.255.255.0
Conf IP>LIST ADDRESS
IP addresses for each interface:
...
intf  2   5.5.5.1      255.255.255.0    NETWORK broadcast,    fill 0
...
Conf IP>EXIT
```

Example with an interface without an assigned number:

```
*P 4
Config>PROTOCOL IP
Internet protocol user configuration
Conf IP>ADD ADDRESS
Which net is this address for [0]? 2
New address[0.0.0.0]? 0.0.0.2
Conf IP>LIST ADDRESS
IP addresses for each interface:
   intf  0   ...          IP disabled on this interface
   intf  1   ...          IP disabled on this interface
   intf  2   0.0.0.2      255.255.255.0 NETWORK broadcast, fill 0
Conf IP>EXIT
Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Config>
```

2. IP Tunnel interface configuration (TNIP)

The TNIP interface configuration commands are described in this chapter. In order to access the TNIP configuration environment, you need to enter the following commands:

```
Config>NETWORK 2

-- IP Tunnel Net Config --
TNIP config>
```

Command	Function
? (HELP)	Lists the available commands or their options
SET	Configures the type of tunnel and its source and destination
ENABLE	Enables the tunnel or some of the options affecting operation
DISABLE	Disables the tunnel or some of its options
LIST	Displays the current configuration of the IP tunnel
EXIT	Exit the TNIP configuration process

The letters written in **bold** are the minimum number that you must enter in order to make the command effective.

2.1. ?(HELP)

You use the **?(HELP)** command to list the available commands in the prompt where you are working. You can also subsequently use this command after a specific command to list the available options.

Syntax:

```
TNIP config>?
```

Example:

```
TNIP config>?
SET
ENABLE
DISABLE
LIST
EXIT
TNIP config>
```

2.2. SET

This command is used to configure the tunnel source and destination and to access the used encapsulated protocol configuration.

Syntax:

```
TNIP config>SET ?
PROTOCOL
SOURCE
DESTINATION
```

a) SET PROTOCOL

This is used to access the encapsulated protocol configuration. This is GRE by default and is currently the only one supported.

Example:

```
TNIP config>SET PROTOCOL

-- GRE Config --
GRE config>
```

The TNIP interface supports various types of encapsulation. Through this command you specify the type of encapsulation going to be used.

b) SET SOURCE

Through this command the IP tunnel source IP address is configured. This must coincide with the IP address of one of the router's configured interfaces except that of the tunnel itself (Ethernet, FRAME RELAY, internal etc.) and with the IP address configured as destination in the device at the other end of the tunnel.

If the source IP address does not coincide with any of the router's interfaces, the packets destined to this IP address will not be received by the router as own and it will try to route them towards another device.

If the configured source IP address does not coincide with the destination address configured at the router's other end, the link will never exist.

Example:

```
TNIP config>SET SOURCE
Tunnel Source address  [0.0.0.0]?1.1.6.1
TNIP config>
```

c) SET DESTINATION

Through this command you configure the IP tunnel destination IP address. This must coincide with the IP address configured as source in the router at the other end.

If the destination IP address does not coincide with the source configured address at the other end, the packets routed to this router will be discarded as not pertaining to the tunnel.

This route must exist towards this destination or else the tunnel packets cannot be rerouted. As a precaution, this route must be a static route to avoid recursive problems in the routing tables as explained in chapter 1.

Example:

```
TNIP config>SET DESTINATION
Tunnel destination address  [0.0.0.0]?2.2.6.1
TNIP config>
```

2.3. ENABLE

This is used to enable the tunnel

Syntax:

```
TNIP config>ENABLE ?
TUNNEL
```

a) ENABLE TUNNEL

The tunnel is not active by default. In order to activate it in the configuration, you use this command.

Example:

```
TNIP config>ENABLE TUNNEL
TNIP config>
```

2.4. DISABLE

This command is used to disable the tunnel.

Syntax:

```
TNIP config>DISABLE ?
TUNNEL
```

a) DISABLE TUNNEL

The tunnel is not active by default. In order to deactivate it in the configuration, you use this command.

Example:

```
TNIP config>DISABLE TUNNEL
TNIP config>
```

2.5. LIST

This command is used to list the tunnel configuration.

Syntax:

```
TNIP config>LIST ?
STATE
TUNNEL_MODE
ADDRESSES
ALL
```

a) LIST STATE

This displays the tunnel status. The possibilities are enabled or disabled.

Example:

```
TNIP config>LIST STATE
Tunneling IP:   enable
TNIP config>
```

b) LIST TUNNEL MODE

This command shows the type of encapsulation used. Only GRE is currently supported.

Example:

```
TNIP config>LIST TUNNEL_MODE
Tunnel Mode: GRE
TNIP config>
```

c) LIST ADDRESSES

The tunnel source and destination IP addresses are displayed through this command.

Example:

```
TNIP config>LIST ADDRESSES
Tunnel Addresses
Source:  1.1.6.1
Destination: 2.2.6.1
TNIP config>
```

d) LIST ALL

The tunnel configuration status, type of encapsulation used and the tunnel source and destination IP addresses are displayed through this command.

Example:

```
TNIP config>LIST ALL
Tunnel Mode: GRE
Tunnel Addresses
Source:  1.1.6.1
Destination: 2.2.6.1
Tunneling IP:   enabled
TNIP config>
```

2.6. EXIT

Use the **EXIT** command to return to the previous prompt.

Syntax:

```
TNIP config>EXIT
```

Example:

```
TNIP config>EXIT
Config>
```

3. Configuration of encapsulated GRE protocol

The encapsulated protocol configuration commands are described in this chapter. As previously said, only GRE is currently supported consequently this section corresponds to the GRE protocol configuration. In order to access the GRE configuration environment and as seen in section 2.2 a 'SET PROTOCOL', you need to enter the following commands:

```
Config>NETWORK 2

-- IP Tunnel Net Config --
TNIP config>SET PROTOCOL

-- GRE Config --
GRE config>
```

Command	Function
? (HELP)	Lists the available commands or the options
SET	Configure encryption
ENABLE	Enables some of the options which permit encapsulated protocol and which affect the tunnel operation
DISABLE	Disables some of the options
LIST	Displays current configuration
EXIT	Exits the GRE configuration process

The letters written in **bold** are the minimum number that you must enter in order to make the command effective.

3.1. ?(HELP)

You use the **?(HELP)** command to list the available commands in the prompt where you are working. You can also subsequently use this command after a specific command to list the available options.

Syntax:

```
GRE config>?
```

Example:

```
GRE config>?
SET
ENABLE
DISABLE
LIST
EXIT
GRE config>
```

3.2. SET

This command is used to configure the GRE encapsulated protocol.

Syntax:

```
GRE config>SET ?
CIPHER KEY
```

a) SET CIPHER KEY

The GRE protocol supports data cipher through the RC 4 protocol. Through this command you can configure the session key of the tunnel interface. This key admits a maximum of 32 alphanumerical characters. In cases of enabling cipher and not specifically configuring the key, the default key will be used.

Example:

```
GRE config>SET  CIPHER KEY
Cipher key :        ****************
Rewrite key:        ****************
New cipher key set
GRE config>
```

3.3. ENABLE

This command is used to enable the options permitting GRE protocol and controlling the tunnel operation.

Syntax:

```
TNIP config>ENABLE ?
CHECKSUM
CIPHER
KEY
PROP_SEQ
SEQUENCE
```

a) ENABLE CHECKSUM

This command is used to send the checksum. By default the tunnel does not guarantee the integrity of the packets. By enabling this option, the router sends the packets with a checksum field. If a packet is received with checksum, the device always checks this discarding those packets whose checksum is invalid even if the device has this particular option disabled.

Example:

```
GRE config>ENABLE CHECKSUM
GRE config>
```

b) ENABLE CIPHER

The GRE protocol supports the possibility of ciphering data through the RC4 protocol. This command activates the tunnel interface ciphering.

Example:

```
GRE config>ENABLE CIPHER
 GRE config>
```

c) ENABLE KEY

This command is used to enable the tunnel identifier check. On enabling this option, the device requests an identifier for the tunnel in question. This tunnel identifier must be the same at both ends of the tunnel. This option is disabled by default.

By enabling this option, the router discards those packets containing an different identifier to that configured.

Example:

```
GRE config>ENABLE KEY
Tunnel key : [0]?12345
GRE config>
```

d) ENABLE PROPRIETARY SEQUENCE

This is used to enable the proprietary sequence number. On enabling this option, in cases of de-synchronization at the ends of the tunnel (packets received with the sequence number being less than expected), the return to normal operating mode is much quicker as the packets now include the sequence number expected by the other end. By default this option is disabled in the tunnel. We recommend the use of this option when there are Teldat Routers at both ends of the tunnel.

Example:

```
GRE config>ENABLE PROP_SEQ
GRE config>
```

e) ENABLE SEQUENCE

This is used to enable the option to ensure order in the incoming datagrams. By default this option is disabled in the tunnel.

On enabling this option, the router discards those packets which do not arrive in order.

Example:

```
GRE config>ENABLE SEQUENCE
GRE config>
```

3.4. DISABLE

This command is used to disable the options permitting GRE protocol and controlling the tunnel operation.

Syntax:

```
TNIP config>DISABLE ?
CHECKSUM
CIPHER
KEY
PROP_SEQ
SEQUENCE
```

a) DISABLE CHECKSUM

This command is used to disable the send checksum. By default the tunnel does not guarantee the integrity of the packets. By disabling this option, the router sends the packets without a checksum field. If a packet is received with checksum, the device always checks this discarding those packets whose checksum is invalid.

Example:

```
GRE config>DISABLE CHECKSUM
GRE config>
```

b) DISABLE CIPHER

The GRE protocol supports the possibility of ciphering data through the RC4 protocol. This command deactivates the tunnel interface ciphering.

Example:

```
GRE config>DISABLE CIPHER
GRE config>
```

c) DISABLE KEY

This command is used to disable the tunnel identifier check. This option is disabled by default.

On disabling this option, the device discards those packets containing an identifier.

Example:

```
GRE config>DISABLE  KEY
GRE config>
```

d) DISABLE PROPIETARY SEQUENCE

This is used to disable the proprietary sequence number use. By default this option is disabled in the tunnel.

Example:

```
GRE config>DISABLE PROP_SEQ
GRE config>
```

e) DISABLE SEQUENCE

This is used to disable the option to ensure the order of the incoming datagrams. By default, this option is disabled in the tunnel.

By disabling this option, the router does not discard those packets arriving out of order.

Example:

```
GRE config>DISABLE SEQUENCE
GRE config>
```

3.5. LIST

This is used to list the GRE protocol configuration for the TNIP interface.

Syntax:

```
TNIP config>LIST ?
ALL
STATE
OPTIONS
```

a) LIST STATE

Displays the cipher configuration status. The possibilities are enabled and disabled.

Example:

```
GRE config>LIST STATE
Cipher:         enable
GRE config>
```

b) LIST OPTIONS

Displays the status (enabled/disabled) of the options controlling the tunnel operation.

Example:

```
GRE config>LIST OPTIONS
GRE Options GRE
End-to-End Checksumming:     disabled
Tunnel identification key:   enabled
--------------------> key:   12345
Drop Out-of-Order Datagrams: disabled
Proprietary sequence number: disabled
GRE config>
```

c) LIST ALL

Displays the configured cipher status and the status of the options controlling the tunnel operation.

Example:

```
GRE config>LIST ALL
Cipher:         enable
GRE Options GRE
End-to-End Checksumming:     disabled
Tunnel identification key:   enabled
--------------------> key:   12345
Drop Out-of-Order Datagrams: disabled
Proprietary sequence number: disabled
GRE config>
```

3.6. EXIT

The **EXIT** command is used to return to the previous prompt.

Syntax:

```
GRE config>EXIT ?
```

Example:

```
GRE config>EXIT
TNP Config>
```

Chapter 3
IP tunnel Interface Monitoring (TNIP)

1. IP Tunnel interface statistics (TNIP)

1.1. Introduction

- Viewing of the IP Tunnel interface monitoring prompt: this corresponds to the part of the dynamic tunnels that will be seen in section 6.1 in chapter 6, Dynamic Tunnels (Internet).
- IP Tunnel interface monitoring commands: this corresponds to the part of the dynamic tunnels that will be seen in section 6.2 in chapter 6, Dynamic Tunnels (Internet).

1.2. TNIP Interfaces and the MONITORs process DEVICE command

On executing the DEVICE command from the MONITOR process (+) prompt, all the TNIP interface statistics are displayed:

```
+DEVICE 2
Auto-test    Auto-test     Maintenance
Ifc  Interface     CSR     Vect      valids    failures        failures
2    TNIP/0          0       0           2           0               0
Input Stats
-----------
  Frames ok    0
  Frames error 0
  ---> Invalid encapsulated     0
  ---> Out-of-Order frames      0
  ---> Checsksum errors         0
  ---> Key errors               0
  ---> Unknown payload protocol 0
  ---> Error in cipher          0
  ---> Internal errors          0
Output Stats
------------
  Frames ok    0
  Frames error 0
  ---> Invalid encapsulation    0
  ---> Unknow internal protocol 0
+
```

Chapter 4
IP tunnel configuration examples

1. Steps to follow. IP tunnel over IP

1.1. Steps to follow at each end of the tunnel

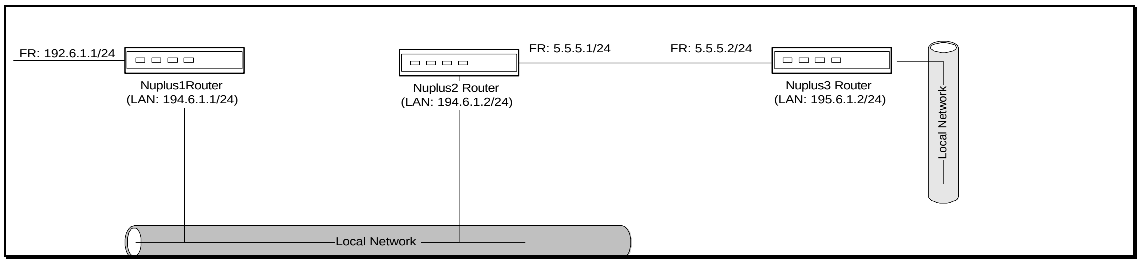
- Create the IP tunnel interface, save it and restart.
- Assign an IP address to the tunnel interface or configure it without a number.
- Configure the tunnel source. This must be the IP address of an interface existing in the device but not the tunnel itself.
- Configure the tunnel destination. Aggregate the necessary IP route in order to reach this destination.
- Configure the encapsulated protocol which travels in the tunnel (or type of tunnel).
- Enable the desired options (checksum, sequence number and/or identifier).
- Aggregate the IP routes for those networks that need to be accessed through the IP tunnel giving the other end tunnel interface address as the next hop (should this be configured) or the tunnel interface (should this be configured without a number).
- Enable the tunnel, save and restart.

1.2. Steps to follow for those devices which use the tunnel

- Aggregate the necessary routes so the tunnel source and destination are accessible.

1.3. Example 1: IP over IP with GRE

Nuplus1 source and Nuplus3 destination tunnel configuration so the networks 193.6.1.0/24 and 195.6.1.0/24 can communicate.



a) Nuplus1 Configuration

Aggregating the Frame Relay interface and the IP tunnel.

```
*P 4
Config>SET HOSTNAME
What is the new router name?[]?Nuplus1
Config>LIST DEVICE

Con    Ifc Type of interface                   CSR     CSR2  int
---      1 Router->Node                          0             0
---      2 Node->Router                          0             0
LAN      0 Ethernet                        9000000            1C
WAN1     3 X25                             F001600  F000C00   9E
WAN2     4 X25                             F001620  F000D00   9D
ISDN 1   5 channel D: X.25                 A000000            1B
ISDN 1   7 channel B: X.25                 F001640  F000E00   9C
ISDN 2   6 channel D: X.25                 A200000            1B
ISDN 2   8 channel B: X.25                 F001660  F000F00   9B
Config>SET DATA FR
which port will be changed[1]?
Config>ADD DEVICE TNIP
Added TNIP interface with num: 2

TNIP tunnel encapsulated type:[GRE]?
Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

Configuration of the interface addresses.

```
Nuplus1 *P 4
Nuplus1 Config>PROTOCOL IP
Internet protocol user configuration
Nuplus1 Conf IP>LIST ADDRESS
IP addresses for each interface:
   intf  0                                  IP disabled on this interface
   intf  1                                  IP disabled on this interface
   intf  2                                  IP disabled on this interface
   intf  3                                  IP disabled on this interface
Nuplus1 Conf IP>ADD ADDRESS
Which net is this address for[0]?
New address [0.0.0.0]? 194.6.1.1
Address mask [255.255.255.0]? 255.255.255.0
Nuplus1 Conf IP>ADD ADDRESS
Which net is this address for[0]? 1
New address[0.0.0.0]? 193.6.1.1
Address mask [255.255.255.0]? 255.255.255.0
Nuplus1 Conf IP>ADD ADDRESS
Which net is this address for[0]? 2
New address [0.0.0.0]? 0.0.0.2
Nuplus1 Conf IP>LIST ADDRESS
IP addresses for each interface:
   intf  0   194.6.1.1       255.255.255.0    NETWORK broadcast,    fill 0
   intf  1   193.6.1.1       255.255.255.0    NETWORK broadcast,    fill 0
   intf  2   0.0.0.2         0.0.0.0          NETWORK broadcast,    fill 0
   intf  3                                    IP disabled on this interface
Nuplus1 Conf IP>EXIT
```

After that, configure the tunnel.

```
Nuplus1 Config>NETWORK 2

-- IP Tunnel Net Config --
Nuplus1 TNIP config>LIST ALL
Tunnel Mode: GRE
Tunnel Addresses
Source:      0.0.0.0
Destination: 0.0.0.0
Tunneling IP:   disabled
TNIP config>
Nuplus1 TNIP config>SET SOURCE

Tunnel Source address [0.0.0.0]? 194.6.1.1
Nuplus1 TNIP config>SET DESTINATION

Tunnel Destination address [0.0.0.0]? 5.5.5.2
Nuplus1 TNIP config>ENABLE TUNNEL
Nuplus1 TNIP config>SET PROTOCOL

-- GRE Config --
Nuplus1 GRE config>LIST ALL
Cipher:         disabled
GRE Options GRE
End-to-End Checksumming:      disabled
Tunnel identification key:    disabled
Drop Out-of-Order Datagrams:  disabled
Proprietary sequence number:  disabled
Nuplus1 GRE config>
Nuplus1 GRE config>ENABLE CHECKSUM
Nuplus1 GRE config>ENABLE KEY

Tunnel key: [0]? 1234
Nuplus1 GRE config>ENABLE SEQUENCE
Nuplus1 GRE config>EXIT
Nuplus1 TNIP config>EXIT
```

Aggregate the necessary routes.

```
Nuplus1 Config>PROTOCOL IP
Internet protocol user configuration
Nuplus1 Conf IP>LIST ROUTE

Nuplus1 Conf IP>ADD ROUTE
IP destination [0.0.0.0]? 5.5.5.2
Address mask [0.0.0.0]? 255.255.255.255
Via gateway at [0.0.0.0]? 194.6.1.2
Cost[1]?
Nuplus1 Conf IP>ADD ROUTE
IP destination [0.0.0.0]? 195.6.1.0
Address mask [0.0.0.0]? 255.255.255.0
Via gateway at [0.0.0.0]? 0.0.0.2
Cost[1]?
Nuplus1 Conf IP>EXIT
Nuplus1 Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Nuplus1 Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

```
Nuplus1 *P 3
Console Operator
Nuplus1 +PROTOCOL IP
Nuplus1 IP>INTERFACE
Interface  IP Address(es)        Mask(s)
   Eth/0     194.6.1.1        255.255.255.0
    FR/0     193.6.1.1        255.255.255.0
  TNIP/0     0.0.0.0          0.0.0.0
Nuplus1 IP>DUMP
Type   Dest net           Mask  Cost  Age  Next hop(s)

 Est*  5.5.5.2          FFFFFFFF  1    0    194.6.1.2
 Dir*  193.6.1.0        FFFFFF00  1    0    FR/0
 Dir*  194.6.1.0        FFFFFF00  1    0    Eth/0
 Est*  195.6.1.0        FFFFFF00  1    0    TNIP/0

Routing table size: 768 nets (49152 bytes), 4 nets known
Nuplus1 IP>
```

b) Nuplus2 Configuration

Aggregate the Frame Relay interface.

```
*P 4
Config>SET HOSTNAME
What is the new router name?[]?Nuplus2
Config>SET DATA FR
which port will be changed[1]?
Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

Configure the Frame Relay interface.

```
nuplus2 *P 4
nuplus2 Config>NETWORK 1

-- Frame Relay user configuration --
nuplus2 FR config>DISABLE LMI
nuplus2 FR config>SET FRAME-SIZE
Interface MTU in bytes [2048]? 1500
nuplus2 FR config>ADD PVC
Circuit number[16]?
Committed Information Rate (CIR) in bps[16000]? 64000
Committed Burst Size (Bc) in bits[16000]?
Excess Burst Size (Be) in bits[0]?
Encrypt information? [No]:(Yes/No)?
Assign circuit name[]?
nuplus2 FR config>ADD PROTOCOL-ADDRESS
IP Address [0.0.0.0]? 5.5.5.2
Circuit number[16]?
nuplus2 FR config>EXIT
```

Configure the interface addresses.

```
nuplus2 Config>PROTOCOL IP
Internet protocol user configuration
nuplus2 Conf IP>ADD ADDRESS
Which net is this address for[0]?
New address [0.0.0.0]? 194.6.1.2
Address mask [255.255.255.0]? 255.255.255.0
nuplus2 Conf IP>ADD ADDRESS
Which net is this address for[0]? 1
New address [0.0.0.0]? 5.5.5.1
Address mask [255.255.255.0]? 255.255.255.0
nuplus2 Conf IP>LIST ADDRESS
IP addresses for each interface:
   intf  0   194.6.1.2       255.255.255.0    NETWORK broadcast,    fill 0
   intf  1   5.5.5.1         255.255.255.0    NETWORK broadcast,    fill 0
   intf  2                                    IP disabled on this interface
nuplus2 Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

```
nuplus2 *p 3
Console Operator
nuplus2 +PROTOCOL IP
nuplus2 IP>INTERFACE
Interface  IP Address(es)        Mask(s)
  Eth/0     194.6.1.2        255.255.255.0
   FR/0     5.5.5.1          255.255.255.0
nuplus2 IP>
```

c) Nuplus3 Configuration

Aggregate the Frame Relay interface and the IP tunnel.

```
*P 4
Config>SET HOSTNAME
What is the new router name?[]?Nuplus3
Config>SET DATA FR
which port will be changed[1]?
Config>ADD DEVICE TNIP
Added TNIP interface with num: 2

TNIP tunnel encapsulated type:[GRE]?
Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

Configure the interface addresses.

```
Nuplus3 *P 4
Nuplus3 Config>PROTOCOL IP
Internet protocol user configuration
Nuplus1 Conf IP>LIST ADDRESS
IP addresses for each interface:
   intf  0                                 IP disabled on this interface
   intf  1                                 IP disabled on this interface
   intf  2                                 IP disabled on this interface
   intf  3                                 IP disabled on this interface
Nuplus3 Conf IP>ADD ADDRESS
Which net is this address for[0]?
New address [0.0.0.0]? 195.6.1.1
Address mask [255.255.255.0]? 255.255.255.0
Nuplus3 Conf IP>ADD ADDRESS
Which net is this address for[0]? 1
New address[0.0.0.0]? 5.5.5.2
Address mask [255.255.255.0]? 255.255.255.0
Nuplus3 Conf IP>ADD ADDRESS
Which net is this address for[0]? 2
New address [0.0.0.0]? 0.0.0.2
Nuplus3 Conf IP>LIST ADDRESS
IP addresses for each interface:
   intf  0   195.6.1.1        255.255.255.0    NETWORK broadcast,    fill 0
   intf  1   5.5.5.2          255.255.255.0    NETWORK broadcast,    fill 0
   intf  2   0.0.0.2          0.0.0.0          NETWORK broadcast,    fill 0
   intf  3                                     IP disabled on this interface
Nuplus3 Conf IP>EXIT
```

Then configure the Frame Relay interface

```
Nuplus3 *P 4
Nuplus3 Config>NETWORK 1

-- Frame Relay user configuration --
Nuplus3 FR config>DISABLE LMI
Nuplus3 FR config>SET FRAME-SIZE
Interface MTU in bytes [2048]? 1500
Nuplus3 FR config>ADD PVC
Circuit number[16]?
Committed Information Rate (CIR) in bps[16000]? 64000
Committed Burst Size (Bc) in bits[16000]?
Excess Burst Size (Be) in bits[0]?
Encrypt information? [No]:(Yes/No)?
Assign circuit name[]?
Nuplus3 FR config>ADD PROTOCOL-ADDRESS
IP Address [0.0.0.0]? 5.5.5.1
Circuit number[16]?
Nuplus3 FR config>EXIT
```

Next, configure the tunnel

```
Nuplus3 Config>NETWORK 2

-- IP Tunnel Net Config --
Nuplus3 TNIP config>SET SOURCE

Tunnel Source address [0.0.0.0]? 5.5.5.2
Nuplus3 TNIP config>SET DESTINATION

Tunnel Destination address [0.0.0.0]? 194.6.1.1
Nuplus3 TNIP config>ENABLE TUNNEL
Nuplus3 TNIP config>SET PROTOCOL

-- GRE Config --
Nuplus3 GRE config>ENABLE CHECKSUM
Nuplus3 GRE config>ENABLE KEY

Tunnel key: [0]? 1234
Nuplus3 GRE config>ENABLE SEQUENCE
Nuplus3 GRE config>EXIT
Nuplus3 TNIP config>EXIT
```

Aggregate the necessary routes

```
Nuplus3 Config>PROTOCOL IP
Internet protocol user configuration
Nuplus3 Conf IP>LIST ROUTE

Nuplus3 Conf IP>ADD ROUTE
IP destination [0.0.0.0]? 194.6.1.1
Address mask [0.0.0.0]? 255.255.255.255
Via gateway at [0.0.0.0]? 5.5.5.1
Cost[1]?
Nuplus3 Conf IP>ADD ROUTE
IP destination [0.0.0.0]? 193.6.1.0
Address mask [0.0.0.0]? 255.255.255.0
Via gateway at [0.0.0.0]? 0.0.0.2
Cost[1]?
Nuplus3 Conf IP>EXIT
Nuplus3 Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Nuplus3 Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

```
Nuplus3 *P 3
Console Operator
Nuplus3 +PROTOCOL IP
Nuplus3 IP>INTERFACE
Interface  IP Address(es)        Mask(s)
   Eth/0     195.6.1.1        255.255.255.0
    FR/0     5.5.5.2          255.255.255.0
  TNIP/0     0.0.0.0          0.0.0.0
Nuplus3 IP>DUMP
Type   Dest net           Mask  Cost  Age  Next hop(s)

 Dir*  5.5.5.0          FFFFFF00  1    0    FR/0
 Est*  193.6.1.0        FFFFFFFF  1    0    TNIP/0
 Est*  194.6.1.1        FFFFFFFF  1    0    5.5.5.1
 Dir*  195.6.1.0        FFFFFF00  1    0    Eth/0

Routing table size: 768 nets (49152 bytes), 4 nets known
Nuplus3 IP>
```

2. Steps to follow. IP tunnel over SRT

2.1. Steps to follow at each end of the tunnel

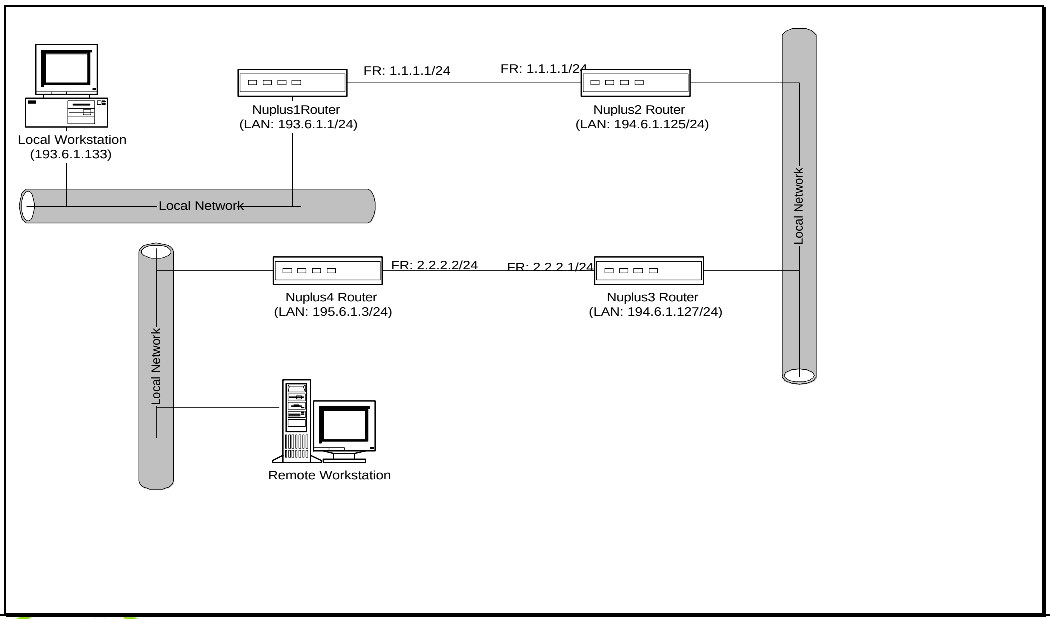
- Create the IP tunnel interface, save and restart.
- Enable the bridge.
- Aggregate a port in the bridge for the tunnel interface.
- Configure the tunnel source which should be the IP address of an existing interface but not that of the tunnel itself.
- Configure tunnel destination. Aggregate the necessary IP route in order to reach the destination.
- Configure the encapsulating protocol which will travel in the tunnel (or type of tunnel).
- Enable required options (checksum, sequence number and/or identifier).
- Enable tunnel, save and restart.

2.2. Steps to follow for those devices which use the tunnel

- Aggregate the necessary routes so the tunnel source and destination are accessible.

2.3. Example: IP over SRT with GRE

Configure a tunnel with Nuplus1 source and Nuplus4 destination so that the networks 193.6.1.0/24 and 195.6.1.0/24 can communicate through NetBEUI traffic. To achieve this, you need to establish an IP tunnel over SRT between them both.



a) Nuplus1 Configuration

Similarly to the previous example, the FR interface and the IP tunnel interface (TNIP) must be added

```
*P 4
Config>SET HOSTNAME
What is the new router name?[]?Nuplus1
Config>SET DATA FR
which port will be changed[1]?
Config>ADD DEVICE TNIP
Added TNIP interface with num: 2

TNIP tunnel encapsulated type:[GRE]?
Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

Configure the interface addresses

```
Nuplus1 *P 4
Nuplus1 Config>PROTOCOL IP
Internet protocol user configuration
Which net is this address for[0]?
New address [0.0.0.0]? 193.6.1.133
Address mask [255.255.255.0]? 255.255.255.0
Nuplus1 Conf IP>ADD ADDRESS
Which net is this address for[0]? 1
New address[0.0.0.0]? 1.1.1.1
Address mask [255.255.255.0]? 255.255.255.0
Nuplus1 Conf IP>ADD ADDRESS
Which net is this address for[0]? 2
New address [0.0.0.0]? 0.0.0.2
Nuplus1 Conf IP>LIST ADDRESS
IP addresses for each interface:
   intf  0   193.6.1.133      255.255.255.0    NETWORK broadcast,    fill 0
   intf  1   1.1.1.1          255.255.255.0    NETWORK broadcast,    fill 0
   intf  2   0.0.0.2          0.0.0.0          NETWORK broadcast,    fill 0
   intf  3                                     IP disabled on this interface
Nuplus1 Conf IP>ex
```

Following that, configure the tunnel

```
Nuplus1 Config>NET 2

-- IP Tunnel Net Config --
Nuplus1 TNIP config>SET SOURCE

Tunnel Source address [0.0.0.0]? 1.1.1.1
Nuplus1 TNIP config>SET DESTINATION

Tunnel Destination address [0.0.0.0]? 2.2.2.2
Nuplus1 TNIP config>ENABLE TUNNEL
Nuplus1 TNIP config>SET PROTOCOL

-- GRE Config --
Nuplus1 GRE config>ENABLE CHECKSUM
Nuplus1 GRE config>ENABLE KEY

Tunnel key: [0]? 1234
Nuplus1 GRE config>ENABLE SEQUENCE
Nuplus1 GRE config>ENABLE PROP_SEQ
Nuplus1 GRE config>EXIT
Nuplus1 TNIP config>EXIT
```

Next, configure the Frame Relay interface

```
Nuplus1 Config>NETWORK 1

-- Frame Relay user configuration --
Nuplus1 FR config>DISABLE LMI
Nuplus1 FR config>SET FRAME-SIZE
Interface MTU in bytes [2048]? 1500
Nuplus1 FR config>ADD PVC
Circuit number[16]?
Committed Information Rate (CIR) in bps[16000]? 64000
Committed Burst Size (Bc) in bits[16000]?
Excess Burst Size (Be) in bits[0]?
Encrypt information? [No]:(Yes/No)?
Assign circuit name[]?
Nuplus1 FR config>ADD PROTOCOL-ADDRESS
IP Address [0.0.0.0]? 1.1.1.2
Circuit number[16]?
Nuplus1 FR config>EXIT
```

Then aggregate the necessary routes

```
Nuplus1 Config>PROTOCOL IP
Internet protocol user configuration
Nuplus1 Conf IP>LIST ROUTE

Nuplus1 Conf IP>ADD ROUTE
IP destination [0.0.0.0]? 2.2.2.0
Address mask [0.0.0.0]? 255.255.255.0
Via gateway at [0.0.0.0]? 1.1.1.2
Cost[1]?
Nuplus1 Conf IP>EXIT
Nuplus1 Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Nuplus1 Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

Lastly, configure the bridge.

```
*P 4

Nuplus1 Config>PROTOCOL ASRT

-- ASRT Bridge user configuration --
Nuplus1 ASRT config>ENABLE BRIDGE
Nuplus1 ASRT config>ADD PORT
Interface Number[0]? 2
Port Number[2]?
Nuplus1 ASRT config>DISABLE STP
Nuplus1 ASRT config>LIST BRIDGE
Source Routing Transparent Bridge Configuration
              ====================================================

Bridge:    Enabled                              Bridge behavior: STB
                   +-----------------------------------------+
-------------------|       SOURCE ROUTING INFORMATION        |-----------------
                   +-----------------------------------------+
Bridge Number:          00                      Segments:          0
Max ARE Hop Cnt:        00                      Max STE Hop cnt:  00
1:N SRB:                Not Active              Internal Segment:  0x000
LF-bit interpret:       Extended
                   +-----------------------------------------+
-------------------|             SR-TB INFORMATION           |-----------------
                   +-----------------------------------------+
SR-TB Conversion:        Disabled
TB-Virtual Segment:      0x000                  MTU of TB-Domain:  0
                   +-----------------------------------------+
-------------------|    SPANNING TREE PROTOCOL INFORMATION   |------------------

                   +-----------------------------------------+
Bridge Address:          Default                Bridge Priority:   32768/0x8000

STP Participation:       Disabled
                   +-----------------------------------------+
-------------------|        TRANSLATION INFORMATION          |------------------

                   +-----------------------------------------+
FA<=>GA Conversion:      Enabled                UB-Encapsulation:  Disabled
DLS for the bridge:      Disabled
                   +----------------------------------------+
-------------------|            PORT INFORMATION            |-------------------

                   +----------------------------------------+
Number of ports added: 2
Port:   1      Interface:       0      Behavior:    STB Only   STP: Enabled

Port:   2      Interface:       2      Behavior:    STB Only   STP: Enabled


Nuplus1 ASRT config>EXIT
Nuplus1 Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Nuplus1 Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

b) Nuplus2 Configuration

You need to aggregate the Frame Relay interface in a similar way to the previous examples.

Configure the interface addresses.

Aggregate the necessary route so the tunnel source and destination are accessible.

```
Nuplus2 Config>PROTOCOL IP
Internet protocol user configuration
Nuplus2 Conf IP>ADD ROUTE
IP destination [0.0.0.0]? 2.2.2.0
Address mask [0.0.0.0]? 255.255.255.0
Via gateway at [0.0.0.0]? 194.6.1.127
Cost[1]?
Nuplus2 Conf IP>
```

c) Nuplus3 Configuration

You need to aggregate the Frame Relay interface in a similar way to the previous examples.

Configure the interface addresses.

Aggregate the necessary route so the tunnel source and destination are accessible.

```
Nuplus3 Config>PROTOCOL IP
Internet protocol user configuration
Nuplus3 Conf IP>ADD ROUTE
IP destination [0.0.0.0]? 1.1.1.0
Address mask [0.0.0.0]? 255.255.255.0
Via gateway at [0.0.0.0]? 194.6.1.125
Cost[1]?
Nuplus3 Conf IP>
```

d) Nuplus4 Configuration

The Frame Relay interface and the IP tunnel interface (TNIP) are added in a similar way to the Nuplus1 example above.

```
*P 4
Config>SET HOSTNAME
What is the new router name?[]?Nuplus4
Config>SET DATA FR
which port will be changed[1]?
Config>ADD DEVICE TNIP
Added TNIP interface with num: 2

TNIP tunnel encapsulated type:[GRE]?
Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

Next, configure the interface addresses

```
Nuplus4 *p 4
Nuplus4 Config>PROTOCOL IP
Internet protocol user configuration
Which net is this address for[0]?
New address [0.0.0.0]? 195.6.1.3
Address mask [255.255.255.0]? 255.255.255.0
Nuplus4 Conf IP>ADD ADDRESS
Which net is this address for[0]? 1
New address[0.0.0.0]? 2.2.2.2
Address mask [255.255.255.0]? 255.255.255.0
Nuplus4 Conf IP>ADD ADDRESS
Which net is this address for[0]? 2
New address [0.0.0.0]? 0.0.0.2
Nuplus4 Conf IP>LIST ADDRESS
IP addresses for each interface:
   intf  0   195.6.1. 3       255.255.255.0    NETWORK broadcast,    fill 0
   intf  1   2.2.2.2          255.255.255.0    NETWORK broadcast,    fill 0
   intf  2   0.0.0.2          0.0.0.0          NETWORK broadcast,    fill 0
   intf  3                                     IP disabled on this interface
Nuplus4 Conf IP>EXIT
```

Following that, configure the tunnel

```
Nuplus4 Config>NETWORK 2

-- IP Tunnel Net Config --
Nuplus4 TNIP config>SET SOURCE

Tunnel Source address [0.0.0.0]? 2.2.2.2
Nuplus4 TNIP config>SET DESTINATION

Tunnel Destination address [0.0.0.0]? 1.1.1.1
Nuplus4 TNIP config>ENABLE TUNNEL
Nuplus4 TNIP config>SET PROTOCOL

-- GRE Config --
Nuplus4 GRE config>ENABLE CHECKSUM
Nuplus4 GRE config>ENABLE KEY

Tunnel key: [0]? 1234
Nuplus4 GRE config>ENABLE SEQUENCE
Nuplus4 GRE config>ENABLE PROP_SEQ
Nuplus4 GRE config>EXIT
Nuplus4 TNIP config>EXIT
```

Then configure the Frame Relay interface

```
Nuplus4 Config>NETWORK 1

-- Frame Relay user configuration --
Nuplus4 FR config>DISABLE LMI
Nuplus4 FR config>SET FRAME-SIZE
Interface MTU in bytes [2048]? 1500
Nuplus4 FR config>ADD PVC
Circuit number[16]?
Committed Information Rate (CIR) in bps[16000]? 64000
Committed Burst Size (Bc) in bits[16000]?
Excess Burst Size (Be) in bits[0]?
Encrypt information? [No]:(Yes/No)?
Assign circuit name[]?
Nuplus4 FR config>ADD PROTOCOL-ADDRESS
IP Address [0.0.0.0]? 2.2.2.1
Circuit number[16]?
Nuplus4 FR config>EXIT
```

Add the necessary routes

```
Nuplus4 Config>PROTOCOL IP
Internet protocol user configuration
Nuplus4 Conf IP>LIST ROUTE

Nuplus4 Conf IP>ADD ROUTE
IP destination [0.0.0.0]? 1.1.1.0
Address mask [0.0.0.0]? 255.255.255.0
Via gateway at [0.0.0.0]? 2.2.2.1
Cost[1]?
Nuplus4 Conf IP>EXIT
Nuplus4 Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Nuplus4 Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

Lastly configure the bridge.

```
*P 4

Nuplus4 Config>PROTOCOL ASRT

-- ASRT Bridge user configuration --
Nuplus4 ASRT config>ENABLE BRIDGE
Nuplus4 ASRT config>ADD PORT
Interface Number[0]? 2
Port Number[2]?
Nuplus4 ASRT config>DISABLE STP
Nuplus4 ASRT config>EXIT
Nuplus4 Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Nuplus4 Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

Chapter 5
IP Tunnel interface Events (TNIP)

1. IP Tunnel event monitoring (TNIP)

This permits real time monitoring of the events which occur over one or various TNIP interfaces when the event logging system is enabled for this protocol.

This is enabled from the configuration menu as shown below:

```
*P 4

-- ELS Config --
ELS Config>
ELS Config>ENABLE TRACE SUBSYSTEM TNIP ALL
ELS Config>EXIT
Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

In the same way, these can be enabled from the monitoring menu at any time without needing to store this in the configuration as shown below:

```
*P 3

+EVENT

-- ELS Monitor --
ELS>ENABLE TRACE SUBSYSTEM TNIP ALL
ELS>EXIT
```

The available events list for the TNIP protocol is the following (This does not include the dynamic tunnel specific events defined in chapter 6):

TNIP.001

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.001 Pack rec *str_enc_protocol*, ext prt 0x*num_external_protocol*, (*source_ip_address->destination_ip_address*), int *interface ID*

Long Syntax:

TNIP.001 Packet received with encapsulation *str_enc_protocol*, delivery protocol 0x*num_external_protocol*, (source *source_ip_address* destination *destination_ip_address*), on interface *interface ID*

Description:

A packet with encapsulated protocol given has been received, transported in a delivery protocol with a given number and the source and destination is the given. The interface controlling the desencapsulation is the given.

TNIP.002

Level: Abnormal external error, ERROR-AE/UE-ERROR

Short Syntax:

TNIP.002 Pack rec *str_enc_protocol*, ext prt 0x*num_external_protocol*, (*source_ip_address->destination_ip_address*), no tunnel

Long Syntax:

TNIP.002 Received packet with encapsulation *str_enc_protocol*, delivery protocol 0x*num_external_protocol*, (source *source_ip_address* destination *destination_ip_address*) There is no tunnel for it.

Description:

A packet with encapsulation protocol given has been received. The delivery protocol is the given. With the given source and destination, the tunnel controlling the desencapsulation has not been found.

Cause:

An external device is sending packets to the router, but they are discarded because the source and destination are not configured in the tunnel interface.

Action:

Change the tunnel configuration in order to accept the packets if you require these. Identify the external device and configure it to prevent further packets being sent.

TNIP.003

Level: Abnormal internal error, ERROR-AI/UI-ERROR

Short Syntax:

TNIP.003 Incom pack disc no act int, int *interface ID*

Long Syntax:

TNIP.003 Incoming packet discarded. The interface is down. Interface *interface ID*

Description:

The tunnel interface incoming function has discarded the packet as the interface is down.

TNIP.004

Level: Abnormal internal error, ERROR-AI/UI-ERROR

Short Syntax:

TNIP.004 Incom pack disc no conf int, int *interface ID*

Long Syntax:

TNIP.004 Incoming packet discarded. The interface has no configuration. Interface *interface ID*

Description:

The tunnel interface incoming function has discarded the packet as the interface is not configured.

TNIP.005

Level: Abnormal internal error, ERROR-AI/UI-ERROR

Short Syntax:

TNIP.005 Incom pack disc encapsul prot reject,int *interface ID*

Long Syntax:

TNIP.005 Incoming packet has been discarded because the interface configuration has a different encapsulation protocol, interface *interface ID*

Description:

The tunnel interface incoming function has discarded the packet as the interface is configured in a distinct mode (expected encapsulation protocol) from the packet being received.

Cause:

The interface has been incorrectly configured in one of the tunnel ends.

Action:

Configure the same operating mode at both ends of the tunnel.

TNIP.006

Level: Abnormal internal error, ERROR-AI/UI-ERROR

Short Syntax:

TNIP.006 Err GRE header flag routing act, int *interface ID*

Long Syntax:

TNIP.006 Error in GRE header. Flags have the routing option, interface *interface ID*

Description:

The GRE desencapsulation function has discarded the packet as the routing option in the packet is active.

Cause:

The other end of the tunnel is sending GRE packets with a routing field. These types of packets are currently not accepted.

Action:

Configure the end without GRE routing.

TNIP.007

Level: Abnormal external error, ERROR-AE/UE-ERROR

Short Syntax:

TNIP.007 Err GRE header chksm 0x*checksum* (exp 0x*expected_checksum*), int *int ID*

Long Syntax:

TNIP.007 Error in GRE header, checksum 0x*checksum* (expected 0x*expected_checksum*), interface *interface ID*

Description:

This message is generated when a packet contains an invalid checksum. The received checksum is displayed alongside the correct one.

Cause:

The most likely cause is due to a damaged packet. This could be due to a node constructing an erroneous header.

Action:

If the problem persists, check the trace to determine where the packet is damaged.

TNIP.008

Level: Abnormal external error, ERROR-AE/UE-ERROR

Short Syntax:

TNIP. Dsc GRE pack key *key* (exp *expected_key*), int *interface ID*

Long Syntax:

TNIP.008 Packet GRE discarded, key *key* (expected *expected_key*), interface *interface ID*

Description:

This message is generated when a packet contains an invalid key. The received key is displayed alongside the correct one.

Cause:

This could be due to a node configured with an incorrect key.

Action:

Configure the key correctly.

TNIP.009

Level: Abnormal external error, ERROR-AE/UE-ERROR

Short Syntax:

TNIP.009 Dsc GRE pack with no key(exp *expected_key*), int *interface ID*

Long Syntax:

TNIP.009 Packet GRE with no key discarded (expected *expected_key*), interface *interface ID*

Description:

This message is generated when a packet does not contain the key and the tunnel is configured to check the key identification.

Cause:

The node may be incorrectly configured.

Action:

Configure the key correctly.

TNIP.010

Level: Abnormal external error, ERROR-AE/UE-ERROR

Short Syntax:

TNIP.010 Dsc GRE pack with key (exp sin key), int *interface ID*

Long Syntax:

TNIP.010 Packet GRE with key discarded, key *key*(not expected key), interface *interface ID*

Description:

This message is generated when the packet contains a key and the tunnel is not configured to check the key identification.

Cause:

The node may be incorrectly configured.

Action:

Configure the key correctly.

TNIP.011

Level: Abnormal external error, ERROR-AE/UE-ERROR

Short Syntax:

TNIP.011 Dsc GRE pack seq *seq_num* less exp *expected_seq_num*, int *interface ID*

Long Syntax:

TNIP.011 Packet GRE discarded, sequence number *seq_num* less than expected *expected_seq_num*, interface *interface ID*

Description:

This message is generated when a packet arrives with an invalid sequence number. The received sequence number is displayed alongside the correct one.

Cause:

The ends are not synchronized.

Action:

Wait for them to synchronize. If the sequence numbers are very different and the forecast synchronization time is excessive, you can opt to restart the devices.

TNIP.012

Level: Abnormal internal error, ERROR-AI/UI-ERROR

Short Syntax:

TNIP.012 Err GRE header flag recurs act, int *interface ID*

Long Syntax:

TNIP.012 Error in GRE header. Recursion option active, interface *interface ID*

Description:

The GRE desencapsulation function has discarded the packet because it contains the recursion option active. Multiple encapsulation is not accepted.

Cause:

The other end of the tunnel is sending GRE packets with multiple encapsulation.

Action:

Change the configuration to prevent GRE packets entering the tunnel.

TNIP.013

Level: Abnormal internal error, ERROR-AI/UI-ERROR

Short Syntax:

TNIP.013 Err GRE header wrong vrsn *version*, int *interface ID*

Long Syntax:

TNIP.013 Error in GRE header, invalid encapsulation version *version*, interface *interface ID*

Description:

The GRE desencapsulation function has discarded the packet as it contains an unknown version.

Cause:

The other end of the tunnel is sending GRE packets with a prior version. Or the header may be incorrectly constructed.

Action:

Change the GRE version.

TNIP.014

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.014 Desencap pack GRE , inter prt 0x*protocol_num*, (*source_ip_address->destination_ip_address*), int *interface ID*, seq *seq_num*

Long Syntax:

TNIP.014 Desencapsulated GRE packet with success, payload protocol 0x*protocol_num* (source *source_ip_address* and destination *destination_ip_address*), interface *interface ID*, sequence number *seq_num*

Description:

The desencapsulation procedure of a GRE packet has been successfully completed. The protocol traveling inside the GRE packet (source and destination) and the interface controlling the desencapsulation are specified.

TNIP.015

Level: Abnormal external error, ERROR-AE/UE-ERROR

Short Syntax:

TNIP.015 Desencap pack GRE, inter prt 0x*protocol_num*, unknown, int *interface ID*

Long Syntax:

TNIP.015 Error in desencapsulated GRE packet, unknown payload protocol 0x*protocol_num*, , interface *interface ID*

Description:

The desencapsulation of a GRE packet has been processed, the protocol traveling inside the GRE packet is unknown.

TNIP.016

Level: Abnormal internal error, ERROR-AI/UI-ERROR

Short Syntax:

TNIP.016 Dsc out pack int no act, int *interface ID*

Long Syntax:

TNIP.016 The interface is down so the outgoing packet has been discarded, interface *interface ID*

Description:

The tunnel interface outgoing function has discarded a packet as the interface is not active.

TNIP.017

Level: Abnormal internal error, ERROR-AI/UI-ERROR

Short Syntax:

TNIP.017 Dsc out pack int no conf, int *interface ID*

Long Syntax:

TNIP.017 The interface is not configured so the outgoing packet has been discarded, interface *interface ID*

Description:

The tunnel interface outgoing function has discarded a packet as the interface is not configured.

TNIP.018

Level: Abnormal internal error, ERROR-AI/UI-ERROR

Short Syntax:

TNIP.018 Dsc out pack invalid inter prt 0x*protocol_num*, int *interface ID*

Long Syntax:

TNIP.018 The payload protocol 0x*protocol_num* has been rejected so the outgoing packet has been discarded, interface *interface ID*

Description:

The tunnel interface outgoing function has discarded a packet. It does not accept protocol payload as it is encapsulated.

TNIP.019

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.019 Encap GRE pack, inter prt 0x*protocol_num*, (*source_ip_address->destination_ip_address*), int *interface ID*

Long Syntax:

TNIP.019 Successful encapsulated GRE packet, payload protocol 0x*protocol_num*, (source *source_ip_address* and destination *destination_ip_address*), interface *interface ID*

Description:

The tunnel interface outgoing function has successfully encapsulated a GRE packet. The packet protocol that has been encapsulated is specified as well as the source and the destination.

TNIP.020

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.020 Snd pack *str_enc_protocol*, ext prt 0x*num_external_protocol*, (*source_ip_address->destination_ip_address*), int *interface ID*, seq *seq_num*

Long Syntax:

TNIP.020 Sending packet *str_enc_protocol*, ext prt 0x*num_external_protocol*, (source *source_ip_address* and destination *destination_ip_address*), interface *interface ID*, sequence number *seq_num*

Description:

A packet has been sent with a given protocol, the delivery protocol is the given, the source and destination address is also specified as well as the interface controlling the encapsulation.

TNIP.025

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.025 Pack GRE ciph, inter prt 0x*internal_protocol_num*, (*source_ip_address->destination_ip_address*), int *interface ID*

Long Syntax:

TNIP.025 Successful ciphered GRE packet, payload protocol 0x*internal_protocol_num*, (source *source_ip_address->* destination *destination_ip_address*), interface *interface ID*

Description:

A GRE packet has been ciphered with a encapsulation protocol given among the specific addresses.

TNIP.026

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.026 Pack GRE deciph, inter prt 0x*internal_protocol_num*, (*source_ip_address->destination_ip_address*), int *interface ID*

Long Syntax:

TNIP.026 Successful deciphered GRE packet, payload protocol 0x*internal_protocol_num*, (source *source_ip_address->* destination *destination_ip_address*), interface *interface ID*

Description:

A GRE packet payload has been successfully deciphered. The protocol traveling inside the GRE packet (source and destination) and the interface controlling the desencapsulation are specified.

TNIP.027

Level: Abnormal internal error, ERROR-AI/UI-ERROR

Short Syntax:

TNIP.027 Err deciph GRE, int *interface ID*

Long Syntax:

Error deciphering GRE, interface *interface ID*

args %3"interface ID"

Description:

This message is generated when a packet is deciphered and gives an invalid cipher checksum or the protocol is not ciphered. The most likely possibility is the cipher key is incorrect or that the packet is not ciphered.

TNIP.028

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.028 Desencap pack GRE , inter prt 0x*protocol_num*, int *interface ID*, sec *seq_num*

Long Syntax:

TNIP.028 Desencapsulated GRE packet with success, payload protocol 0x*protocol_num* interface *interface ID*, sequence number *seq_num*

Description:

A GRE packet has been successfully desencapsulated. The protocol traveling inside the GRE packet and the interface controlling the desencapsulation are specified.

TNIP.029

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.029 Pack GRE ciph, inter prt 0x*internal_protocol_num*, int *interface ID*

Long Syntax:

TNIP.029 Successful ciphered GRE packet, payload protocol 0x*internal_protocol_num* interface *interface ID*

Description:

A GRE packet has been ciphered with a given encapsulation protocol.

TNIP.030

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.030 Encap GRE pack, inter prt 0x*protocol_num*, int *interface ID*

Long Syntax:

TNIP.030 Successful encapsulated GRE packet, payload protocol 0x*protocol_num* interface *interface ID*

Description:

A GRE packet has been successfully encapsulated. The encapsulated packet protocol is specified.

TNIP.031

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.031 Desencap pack GRE , inter prt 0x*protocol_num*, (*source_mac_address->destination_mac_address*), int *interface ID*, sec *seq_num*

Long Syntax:

TNIP.031 Desencapsulated GRE packet with success, payload protocol 0x*protocol_num* (source *source_mac_address* and destination *destination_mac_address*), interface *interface ID*, sequence number *seq_num*

Description:

A GRE packet has been successfully desencapsulated. The protocol traveling inside the GRE packet (source and destination mac addresses) and the interface controlling the desencapsulation are specified.

TNIP.032

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.032 Pack GRE ciph, inter prt 0x*internal_protocol_num*, (*source_mac_address->destination_mac_address*), int *interface ID*

Long Syntax:

TNIP.032 Successful ciphered GRE packet, payload protocol 0x*internal_protocol_num*, (source *source_mac_address->* destination *destination_mac_address*), interface *interface ID*

Description:

A GRE packet has been ciphered with a given encapsulation protocol among the specified mac addresses.

TNIP.033

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

TNIP.033 Encap GRE pack, inter prt 0x*protocol_num*, (*source_mac_address->destination_mac_address*), int *interface ID*

Long Syntax:

TNIP.033 Successful encapsulated GRE packet, payload protocol 0x*protocol_num*, (source *source_mac_address* and destination *destination_mac_address*), interface *interface ID*

Description:

The tunnel interface outgoing function has successfully encapsulated a GRE packet. The encapsulated packet protocol as well as the source and destination mac addresses are specified.

Chapter 6
Dynamic Tunnels (Internet)

1. Description

1.1. Introduction

If we apply tunnel technology to the public networks and Internet, interconnection between the diverse local networks becomes possible in both an efficient and cost effective way. From tunnel technology seen in prior chapters this can be carried out, but we do come up against some problems which are dealt with in this chapter among which are the following:

- In order to establish a tunnel between two points, is imperative than both know the end IP address. This means that access is only possible with authenticated accesses and fixed IPs (this is a limited resource within Internet and expensive).
- Connecting *n* local networks require you to configure and establish *n-1* tunnels for each one.

The solution to these problems can be found in a central device (with a fixed known IP address) which supports *n* tunnels and manages the inter-tunnel traffic thus automatically resolving the second problem. The solution to the first problem consists in giving this device the capability to adapt the configuration of its tunnels so it can support connections to devices whose IP address is different each time they connect. **This dynamic tunnel reconfiguration for each new connection is the reason why these tunnels are known as 'dynamic' tunnels**.

From this point onwards, the central device will be referred to as the 'ISP' router *(CENTRIX-ISP)*, as its normal location is to be found in an Internet Server Provider, while the routers connecting to it will be referred to as 'client' routers.

1.2. Scenarios/Presented problems

- **Scenario 4:** Access of local network devices to Internet for network services (html, ftp etc.) through a router (normally with E-NAT).
- **Scenario 5**: Remote local network interconnection with ISP local network through routers with tunnels.
- **Scenario 6:** Interconnection between local networks and with the ISP local network through routers with tunnels.

If your objective is scenario 5/6, the configuration for the 'client' routers is simple as you can predict that any IP address which is not local is accessible through the tunnel. However if the intention is to simultaneously permit scenario 4 as well, the 'client' routers must be able to distinguish if a specific destination address is accessible through the tunnel or out of it (Internet address). This means that the 'clients' must know which networks can be reached through the tunnel. The solution rests in configuring all the possible routes in all the clients or configure them in the ISP central router and that the ISP router informs the clients through RIP.

Furthermore, the ISP device needs to know which networks are accessible through the 'client' routers together with a 'wake-up' mechanism should this be required. This mechanism consists of a table which should associate the ‘clients’ local network addresses with its ISDN number. When you wish to establish a communication with an unconnected ‘client’, the ISP device sends a call indicating to the ‘client’ it needs to establish a connection. The client rejects the call and then carries out a normal connection to Internet.

NOTE 1: As this first call does not connect, it is not charged.

NOTE 2: RIP protocol over the tunnel does not provoke ISDN calls, nor does it release ISDN calls due to absence of data.

NOTE 3: If you do not need scenario 4 but you want the tunnel route by default, you need to disable the *Change route by default* option in the PPP link.

1.3. Types of tunnels

We are going to carry out the following tunnel classification with the idea of analyzing the distinct behavior in each case:

- **Static**: This refers to tunnels where the source and destination addresses are fixed (as seen in chapter 4). These tunnels are not dealt with in this chapter.
- **Dynamic**: When one of the tunnel’s addresses (source or destination) is unknown before making the connection and the device being connected to is unknown.
- **Semi-dynamic**: This deals with a special dynamic tunnel case where despite not knowing the address of the device being connected to, we do know the address of this one as the tunnel identifier (GRE key field) is unique.

(Dynamic and semi-dynamic tunnels are normal in Internet when the address acquired by the remote device is not pre-assigned).

a) ‘Dynamic’ Tunnels

These are the easiest tunnels to configure and at the same time the most flexible. These are more recommendable than the other types.

The ‘ISP’ device offers *n* tunnels to the ‘clients’ which these use as they connect: when a client stops sending information, the tunnel providing the service, becomes free for another connection.

The strength of this configuration is based in the use of RIP where the ‘client’ provides the ‘ISP’ with all the information on the networks that can be accessed through it and vice-versa.

Usually you configure a set of tunnels to provide service to a determined entity with the same ‘key’. This means that the IP groups configuration assigns the same routing table to all the interfaces.

b) Semi-dynamic Tunnels

These are an extension of the dynamic tunnels where we select the remote devices permitted to connect in each ISP tunnel. In order to do this you configure a unique ‘identifier’ in the tunnel which must coincide with that configured in the remote device (GRE Key field). In reality this is like configuring a dynamic tunnel for each remote device.

These tunnels add two functions:

- Remote device identification.
- As we already know the ‘client’ connects to each tunnel, we can aggregate routes and therefore disable RIP.

This means that these tunnels are only recommended when you wish to dispense with RIP or the security and access control are essential, as the configuration is more complex as identifiers must be associated with the remote devices

NOTE 1: As we don’t know which ‘client’ connects with a dynamic tunnel, the use of RIP is essential.

NOTE 2: Wherever you need to ‘wake-up’ a client router, you need to maintain RIP from the client to the ISP.

1.4. The importance of RIP

One of the most delicate aspects of these type of tunnels is the status control; from an ‘idle state’ the status changes to ‘connected’ with the client remote device requesting this. While the tunnel is being used, the status remains ‘connected’ and once terminated, it should return to the initial idle state for future use.

One of the most critical aspects when establishing dynamic tunnels is the if the tunnel remains in use or not as the remote device may have disconnected or been switched off without prior warning. Dialogue is therefore necessary between the routers maintaining the tunnels and standard RIP protocol is used for this.

If you wish to avoid tunnel ‘reuse’ and exclusively reserve it to give service to a ‘client’ device, you can identify it with a unique ‘key’.

Another important point is access to clients who do not have a connection to Internet established. The solution to this is to configure a table containing the accessible address(es), network(s) or subnet(s) through each client in the *ISP* device together with the *ISP* ISDN access number, the function of each being: When the *ISP* device receives information for a **destination existing in the table but is not connected through any tunnel,** the devices makes an ISDN call to the *client* device. The *client* device rejects the call (therefore it is not charged) and carries out a normal connection to Internet.

The client when carrying out the connection must inform the ISP device of the network(s) or subnet(s)accessible through it so the ISP device does not make any further calls and instead delivers any traffic directed to those destinations to the client router. RIP is also used for this.

Summary of the RIP features:

a) Controls the ‘disconnection’ in order to reuse the tunnels.
b) Controls the ‘connection’ when the *clients* devices must be ‘woken-up’.
c) Gives information on the accessible network(s).

Problems that the RIP can present:

When the IP addresses at each end of the tunnel are distinct networks, this can give rise to the router receiving access information on the tunnel destination network through the tunnel interface itself therefore losing access to the remote end. This does not occur in Internet where the address acquired by the client belongs to the ISP device network however it does in all other cases.

The solution for this is to simply add a static route to access the destination provided this is previously known.

In any case, this situation is detected by the routers who in turn report the events and statistics.

2. User scenarios

It is essential to define the use of the router before configuring the tunnels in it in order to take maximum advantage of it and the communications line for this so the configuration corresponds to the needs.

The most important decision is based on configuration of the static route or whether to leave this to a routing protocol. This would make for easier configuration but would diminish performance as some part of the bandwidth would have to be used to exchange messages between routers.

Although a general norm does not exist, we can base this on some guidelines that help us to take the most adequate decision. It is therefore essential to define the scenario where the router is going to operate. For this reason you must have in the access device the following:

- **Scenario 5/6 (tunnel):** All the non local addresses are accessible through the tunnel i.e. there are sufficient to configure this as a default route (with the exception of the tunnel remote end which must be configured as static).
- **Tunnel through Internet + Surfing through Internet:** If the remote local networks are not 10.X.X.X you do not need to configure the routes.

You also need to bear in mind some aspects seen in previous chapters:

- When you use *dynamic* tunnels, you need to configure RIP in order to reuse them.

Similarly when you wish to *wake* a client, you must configure RIP from this towards to ISP.

2.1. Tunnel function without surfing through Internet (Scenario 5/6)

The 'client' configuration is based on defining the tunnel as the default route towards any destination (except local destinations or ISP). Here you may disable the RIP arriving from the ISP therefore improving the line performance due to the fact that RIP traffic flow is high if the ISP supports a large number of tunnels.

a) *Minimum configuration through RIP*

When employing dynamic tunnels (i.e. reusable) it is essential to use RIP in the "client"→ "ISP" direction in order to know the client's accessible networks.

Surfing Permitted:	No
Type of tunnel:	Dynamic
Default route:	Tunnel
RIP:	“Client” $\Rightarrow$ “ISP”
Security:	Low
Difficulty level in configuration	Very low
Efficiency	High

b) More complex configuration reducing RIP traffic

Through dedicated tunnels for each remote device requiring access, the client is identified and RIP is not essential in the “client”$\rightarrow$ “ISP” direction. On the other hand, you must maintain the identifiers when configuring the tunnels in ‘ISP’ and ‘clients’.

Surfing Permitted:	No
Type of tunnel:	Semi-dynamic
Default route:	Tunnel
RIP:	No (“Client” $\Rightarrow$ “ISP”) If you need to ‘wake’ the client
Security:	High
Difficulty level in configuration	Medium
Efficiency	Very high

2.2. Tunnels and Simultaneous Surfing (Scenario 4 + 5/6)

In the ‘clients’, the default route accesses any Internet destination. Therefore any destinations in networks accessible through the tunnel must be known. This is achieved by static configuration in each of them or configured in the ‘ISP’ which in turn informs the ‘clients’ through RIP.

a) Maximum load in the network/Minimum configuration

In cases where there are not very many clients, the routing mechanism can be given to the RIP protocol thus avoiding the need to configure static routes in the ‘clients’.

Surfing Permitted:	No
Type of tunnel:	Dynamic
Default route:	Internet
RIP:	”Clients” ⇔“ISP”
Security:	Low
Difficulty level of configuration	Low
Efficiency	Low if there are many tunnels

This does not exclude specific cases where it is not convenient to use RIP and where a unique key can be dedicated.

This scenario is very useful when you wish to interconnect just a few local networks.

b) Minimum load in the network / More complex configuration

When the number of routes known to the ISP device are high (either due to a high number of tunnels or the remote local networks are complex), the RIP traffic can seriously affect the tunnel performance. In this case you need to evaluate the possibility to dispense with the routing protocol in the ISP→Clients direction. This means you must statically configure the networks needing to be accessed through the tunnel in the clients.

Surfing Permitted:	Yes
Type of tunnel:	Dynamic
Default route:	Internet
RIP:	”Clients” ⇒ “ISP”
Security:	Medium
Difficulty level of configuration	Medium
Efficiency	High

As in the above cases, you can dispense with RIP in the Clients→ISP direction by dedicating a unique key.

This scenario is particularly useful when you wish to access remote local networks through the ISP and not the interconnection of remote local networks. E.g. when an entity has an ISP and wishes to give the branches access.

c) Void overload in the network/More complex configuration/Client control

This scenario is totally based on the use of distinct ‘keys’ so that each tunnel interface is perfectly defined. I.e. you know who the connected client is and the networks that can be accessed through this.

Surfing Permitted:	Yes
Type of tunnel:	Semi-Dynamic
Default route:	Internet
RIP:	No (“Client” ⇒ “ISP”) If you need to ‘wake’ client
Security:	High
Difficulty level for configuration	High
Efficiency	Maxim

This reduces overload to zero as the RIP traffic is totally eliminated by specifically configuring all the accessible routes (unless you need to ‘wake-up’ the client).

Given the greater client control in this scenario, this is useful when the ISP offers network interconnection services to third parties.

Note1: It is possible to use the ‘identifier’ field to distinguish between clients when this service is offered to third parties be it through an identifier dedicated to each remote end or an identifier shared among a maximum number of tunnels simultaneously contracted by a given client.

Note2: When services are offered to a third party, you must avoid duplicating the addresses between client installations. It is an idea to insist that clients use specific networks or subnets to ensure this as well as making the configuration easier. **Or separate the distinct clients through the IP group option.**

3. Security

When you are trying to connect local networks through a public network, the relative security aspects become very important. You must have authentication mechanisms for the connections and mechanisms to avoid undesired inter-tunnel traffic.

The first authentication mechanism is the use of fixed addresses although this is expensive in Internet. If this option cannot be used you need to resort to another mechanism such as GRE protocol identifier (key) which must be known to both ends (as seen in chapter 1, these fields consist of 32 bits providing more than 400 million possible combinations).

AS the inter-tunnel traffic is carried out through the 'ISP' device, you have complete control over it and through the union of the GRE identifiers and the IP groups you can totally isolate the traffic between distinct client tunnels as well as permitting and managing the duplication of addresses between distinct clients without any problem.

4. Configuration

The configuration possibilities are numerous so we are going to concentrate on the most common scenarios 5 and 6 with dynamic tunnels (reusable) and RIP.

4.1. Keys to configure dynamic tunnels

Due to the particulars of these connections (as seen in chapter 1) the following aspects should be borne in mind:

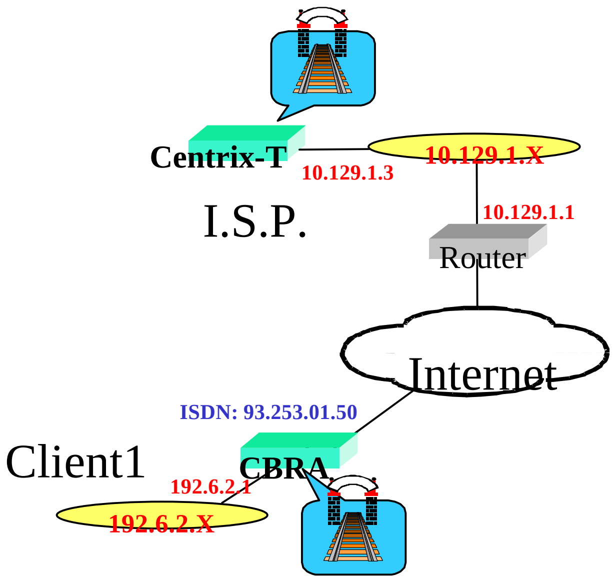
- **On the ISP side** the dynamic tunnels are configured with the following characteristics :
  a) The interfaces of the same client are grouped in the same IP Group
  b) The tunnel source IP address is the address of the valid device in the public network and can be accessed by the clients.
  c) The tunnel destination IP address should be left as 0.0.0.0, as this refers to the client's access device address. This is not previously known and is distinct each time the client initiates a tunnel
  d) The key assigned to the client is configured.
  e) Enable RIP over the tunnel if necessary.

  Also the remote networks ⇔ Nº ISDN table is configured .

- **On the client side** a single tunnel is configured with the following particulars:
  a) The destination address is the address configured as source in the ISP device.
  b) The source address is not previously known, so it must be configured as 0.0.0.0.
  c) The key assigned to the client is configured.
  d) Enable RIP over the tunnel if necessary.

- **Default Route decision,** in cases where the default route is the tunnel instead of the WAN link preconfigured in the router, the *'Change default route'* option must be disabled in the corresponding PPP encapsulation.

4.2. Example 1: Network interconnection and simultaneous access via Internet (Scenario 4 + 5)



From an existing ISP it is possible to add a device to terminate the tunnel at the server end without modifying the existing structure; the only condition being that the device (the "Centrix-T" in the above example) has a public network address assigned that can be accessed by the clients. (In the above example the ISP has 256 Internet addresses assigned, type 10.129.1.X and 10.129.1.3 address assigned to the "Centrix-T").

The CBRA remote office ISDN # has also been configured in order to '*wake*' the device.

If you intend to create an ISP instead of incorporating tunnels in an existing ISP, you can use the "Centrix-T" simultaneously as a device to terminate tunnels and an access Router by simply correctly configuring a Frame Relay line towards the public network.

a) Centrix-T Configuration (ISP Device)

Configuration of the device name and the creation of interfaces:

```
*P 4

Config>SET HOSTNAME CENTRIX
Config>ADD DEVICE TNIP
Added TNIP interface with num: 1

TNIP tunnel encapsulated type:[GRE]?
Config>ADD DEVICE PPP
Type basic access ISDN [2]?1
If you are going to config more than two DIAL interfaces, you must config what t
hey have CSR:F011640 and CSR:F011660 over the ISDN 2 connector
Ifc number to delete: [0]?8
Added PPP-DIAL interface with num: 3
Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

Tunnel interface configuration:

```
Centrix *P 4
Centrix Config>NETWORK 1
-- IP Tunnel Net Config --
Centrix TNIP config>ENABLE TUNNEL
Centrix TNIP config>SET SOURCE 10.129.1.3
```

IP protocol configuration:

```
Centrix TNIP config>EXIT
Centrix Config>PROTOCOL IP
Internet protocol user configuration
Centrix Conf IP>ADD ADDRESS 0 10.129.1.3 255.255.255.0
Centrix Conf IP>ADD ADDRESS 1 0.0.0.1
Centrix Conf IP>ADD ISDN 192.6.2.0 255.255.255.0 *
ISDN Number[]? 932530150
Cost[1]? 5
Centrix Conf IP>ADD ROUTE 192.6.1.0 255.255.255.0 * 10.219.1.2 2
Centrix Conf IP>SET DEFAULT NETWORK-GATEWAY * 10.129.1.1 1
```

RIP protocol configuration:

```
Centrix Conf IP>EXIT
Centrix Config>PROTOCOL RIP
RIP protocol user configuration
Centrix Conf RIP>ENABLE RIP
Centrix Conf RIP>SET COMPATIBILITY 10.129.1.3 * 1 4
```

List of interfaces:

```
Centrix Config>LIST DEVICES
Con    Ifc Type of interface                  CSR     CSR2  int
---      1 IP Tunnel                            0             0
---      4 Router->Node                         0             0
---      5 Node->Router                         0             0
LAN      0 Ethernet                       9000000            1C
WAN1     6 X25                            F001600  F000C00   9E
WAN2     7 X25                            F001620  F000D00   9D
ISDN 1   2 ISDN                           F001640  F000E00   9C
ISDN 1   3 channel B: PPP                       0             0
ISDN 1   8 channel D: X.25                A000000            1B
ISDN 2   9 channel D: X.25                A200000            1B
ISDN 2  10 channel B: X.25                F001660  F000F00   9B
```

List of tunnel interfaces:

```
Centrix Config>NETWORK 1

-- IP Tunnel Net Config --
Centrix TNIP config>LIST ALL
Tunnel Mode: GRE
Tunnel Addresses
Source:      10.129.1.3
Destination: 0.0.0.0
Tunneling IP:   enable
Centrix TNIP config>SET PROTOCOL

-- GRE Config --
Centrix GRE config> LIST ALL
Cipher:         disabled
GRE Options GRE
End-to-End Checksumming:      disabled
Tunnel identification key:    disabled
Drop Out-of-Order Datagrams:  disabled
Proprietary sequence number:  disabled
Centrix GRE config>
```

IP protocol list:

```
Centrix Config>PROTOCOL IP
Internet protocol user configuration
Centrix Conf IP>LIST ALL
Interface addresses
IP addresses for each interface:
   intf  0  (       *) 10.129.1.3    255.255.255.0    NETWORK broadcast,   fill 0
   intf  1  (       *) 0.0.0.1       0.0.0.0          NETWORK broadcast,   fill 0
   intf  2                                            IP disabled on this interface
   intf  3                                            IP disabled on this interface
   intf  4                                            IP disabled on this interface
Routing
route to 192.6.1.0,255.255.255.0 via 10.219.1.2, cost 2 group *
route to 0.0.0.0,0.0.0.0 via 10.129.1.1, cost 1 group *
route to 192.6.2.0,255.255.255.0 via ISDN 932530150 cost 5 group *
Protocols
Directed broadcasts: enabled
RIP: enabled
OSPF: disabled
Per-packet-multipath: disabled
Ip classless: disabled
```

RIP protocol list:

```
Centrix Config>PROTOCOL RIP
RIP protocol user configuration
Centrix Conf RIP>LIST ALL
RIP: enable
RIP default origination: disabled
Options per interface address:
Interface:  0
    Address: 10.129.1.3
        RIP sending disabled on this interface.
        RIP receiving disabled on this interface.
        Authentication:..................No.
        Aggregation type:................Do not aggregate.
        Allow disconnected subnetted networks:..Yes
        Per interface additional cost: 0
Interface:  1
    Address: 0.0.0.1
        Send network routes:.............Yes
        Send subnetwork routes:..........Yes
        Send static routes:..............No
        Send direct routes:..............Yes
        Send default routes:.............No
        Poison reverse enabled:..........Yes
        Autonomous system label:.........0
        Sending compatibility:...........RIP2 Broadcast.
        Receive network routes:..........Yes
        Receive subnetwork routes:.......Yes
        Overwrite default routes:........No
        Overwrite static routes:.........No
        Receiving compatibility:.........RIP1 or RIP2.
        Authentication:..................No.
        Aggregation type:................Do not aggregate.
        Allow disconnected subnetted networks:..Yes
        Per interface additional cost: 0
Accept RIP updates always for:
[NONE]

RIP timers:
Periodic sending timer: 30
Route expire timer: 180
Route garbage timer: 120
Limit RIP: disabled.
```

b) CBRA Configuration (Client device)

You need to correctly configure the CBRA device for Internet connection which can be easily carried out with the Windows environment configuration program and from there carry out the rest of the configuration.

Creation of tunnel interface:

```
*P 4
Config>ADD DEVICE TNIP
Added TNIP interface with num: 7

TNIP tunnel encapsulated type:[GRE]?
Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Config>
*RESTART
Are you sure to restart the system?(Yes/No)? y
```

Tunnel interface configuration:

```
*P 4

Config>NETWORK 7
-- IP Tunnel Net Config --
TNIP config>ENABLE TUNNEL
TNIP config>SET DESTINATION 10.129.1.3
```

IP protocol configuration:

```
TNIP config>EXIT
Config>PROTOCOL IP
Internet protocol user configuration
Conf IP>ADD ADDRESS 7 0.0.0.7
Conf IP>ADD ROUTE 192.6.4.0 255.255.255.0 192.6.2.2 0
Conf IP>SET DEFAULT NETWORK-ROUTER 0.0.0.7 0
Conf IP>DELETE ACCESS-CONTROL 1
                                             Beg End  Beg   End   Beg  End
Type        Source            Destination   Pro Pro  SPrt  SPrt  DPrt DPrt
1 E         0.0.0.0/0           0.0.0.0/0   17  17      0 65535  520   520
Are you sure this is the record you want to delete(Yes/No)? y
Deleted
C_TUNNEL IP config>
```

RIP protocol configuration:

```
Conf IP>EXIT
Config>PROTOCOL RIP
RIP protocol user configuration
Conf RIP>ENABLE RIP
Conf RIP>SET COMPATIBILITY 192.6.2.1 1 4
Conf RIP>SET COMPATIBILITY 192.168.253.1 1 4
Conf RIP>SET COMPATIBILITY 192.168.254.1 1 4
```

Callback configuration via Internet (Exclusively for callback via Internet):

```
Config>NETWORK 5
Circuit Config
Circuit Config>ENCAPSULATOR

-- Interface PPP. Configuration --
PPP Config>ENABLE CALLBACK
PPP Config>EXIT
Circuit Config>EXIT
```

Callback configuration via Internet (Exclusively for callback via Internet):

```
Config>NETWORK 3
ISDN Config
Config ISDN>SET LOCAL-ADDRESS
Local destination[]?Ninguna
Config ISDN>EXIT
Config>net 6
Circuit Config
Circuit Config>ENABLE INCOMING
Circuit Config>ENCAPSULATOR

-- Interface PPP. Configuration --
PPP Config>ENABLE CALLBACK
PPP Config>EXIT
Circuit Config>EXIT
```

List of interfaces:

```
Config>LIST DEVICES
Con    Ifc Type of interface                    CSR     CSR2  int
---      7 IP Tunnel                              0             0
LAN      0 Quicc Ethernet                   F001600  F000C00   9E
WAN1     1 AT COM                           F001620  F000D00   9D
WAN1     2 PPP AT COM                             0             0
ISDN 1   3 ISDN                             F001640  F000E00   9C
ISDN 1   5 Channel B: PPP                         0             0
ISDN 2   4 ISDN                             F001660  F000F00   9B
ISDN 2   6 Channel B: PPP                         0             0
```

Tunnel interface list:

```
TNIP config>LIST ALL
-- IP Tunnel Net Config --

Tunnel Mode: GRE
Tunnel Addresses
Source:       0.0.0.0
Destination: 10.129.1.3
Tunneling IP:   enable
TNIP config>SET PROTOCOL

-- GRE Config --
GRE config>LIST ALL
Cipher:         disabled
GRE Options GRE
End-to-End Checksumming:      disabled
Tunnel identification key:    disabled
Drop Out-of-Order Datagrams:  disabled
Proprietary sequence number:  disabled
GRE config>
```

IP protocol list:

```
Conf IP>LIST ALL
Interface addresses
IP addresses for each interface:
   intf  0   192.6.2.1        255.255.255.0    NETWORK broadcast,    fill 0
   intf  1                                     IP disabled on this interface
   intf  2                                     IP disabled on this interface
   intf  3                                     IP disabled on this interface
   intf  4                                     IP disabled on this interface
   intf  5   192.168.253.1    255.255.255.0    NETWORK broadcast,    fill 0
   intf  6                                     IP disabled on this interface
   intf  7   0.0.0.7          0.0.0.0          NETWORK broadcast,    fill 0
Routing
route to 10.0.0.0,255.0.0.0 via 192.168.253.2, cost 0
route to 192.6.4.0,255.255.255.0 via 192.6.2.2, cost 2
route to 0.0.0.0,0.0.0.0 via 0.0.0.7, cost 0
Protocols
Directed broadcasts: disabled
RIP: enabled
OSPF: disabled
Per-packet-multipath: disabled
Ip classless: disabled
```

Access control list:

```
Conf IP>LIST ACCESS-CONTROL
Access Control is: enabled
List of access control records:
                                            Beg End  Beg   End   Beg  End
Type        Source            Destination   Pro Pro  SPrt  SPrt  DPrt DPrt
1 I         0.0.0.0/0           0.0.0.0/0    0 255     0 65535     0 65535
```

RIP protocol list:

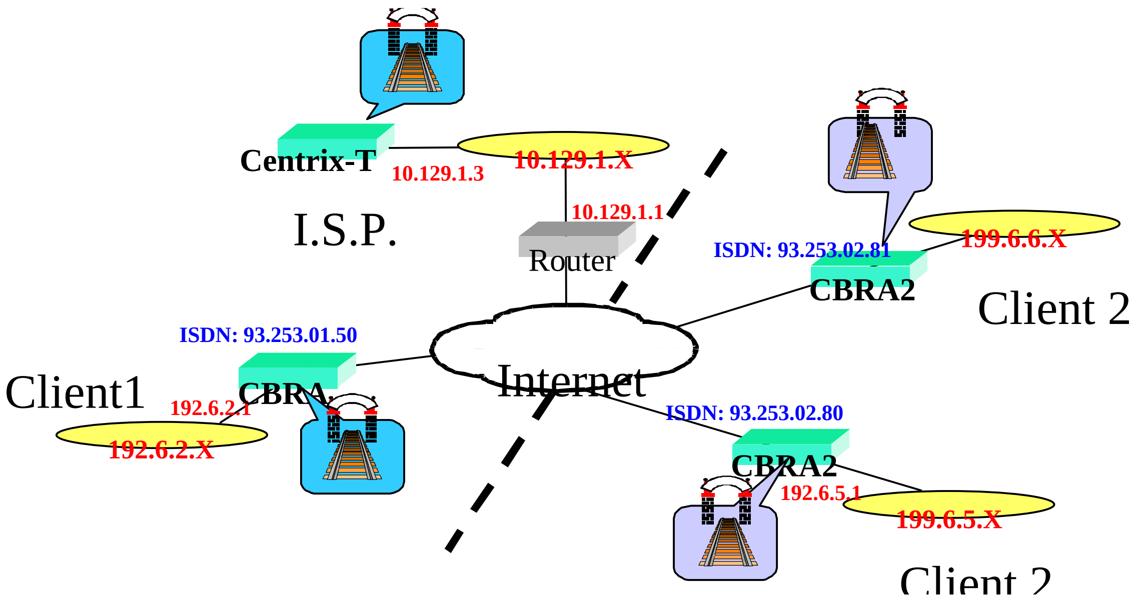
```
Config>PROTOCOL RIP
RIP protocol user configuration
RIP config>LIST ALL
RIP: enabled
RIP default origination: disabled
Options per interface address:
Interface:  0
    Address: 192.6.2.1
        RIP sending disabled on this interface.
        RIP receiving disabled on this interface.
        Authentication:.................No.
        Aggregation type:...............Do not aggregate.
        Allow disconnected subnetted networks:..Yes
        Per interface additional cost: 0
Interface:  5
    Address: 192.168.253.1
        RIP sending disabled on this interface.
        RIP receiving disabled on this interface.
        Authentication:.................No.
        Aggregation type:...............Do not aggregate.
        Allow disconnected subnetted networks:..Yes
        Per interface additional cost: 0
```

```
Interface:  7
    Address: 0.0.0.7
        Send network routes:.............Yes
        Send subnetwork routes:..........Yes
        Send static routes:..............No
        Send direct routes:..............Yes
        Send default routes:.............No
        Poison reverse enabled:..........Yes
        Autonomous system label:.........0
        Sending compatibility:...........RIP2 Broadcast.
        Receive network routes:..........Yes
        Receive subnetwork routes:.......Yes
        Overwrite default routes:........No
        Overwrite static routes:.........No
        Receiving compatibility:.........RIP1 or RIP2.
        Authentication:..................No.
        Aggregation type:................Do not aggregate.
        Allow disconnected subnetted networks:..Yes
        Per interface additional cost: 0
Accept RIP updates always for:
[NONE]

RIP timers:
Periodic sending timer: 30
Route expire timer: 180
Route garbage timer: 120
Limit RIP: disabled.
```

4.3. Example 2: Scenario 6 (Via Internet)

From the previously example, we will connect a third party entity with two local networks through tunnels, completely isolating the new traffic from the existing traffic. I.e. the new tunnels are seen but not together with the old tunnels nor the ISP network.



a) Necessary Modifications in the CENTRIX-T (ISP device)

Creating the new tunnel interfaces:

```
Centrix *P 4

Centrix Config>ADD DEVICE TNIP
Added TNIP interface with num: 2
Centrix Config>ADD DEVICE TNIP
Added TNIP interface with num: 3

TNIP tunnel encapsulated type:[GRE]?
Centrix Config>SAVE
Save configuration [n]? y
Saving configuration...OK
Centrix Config>^p
Centrix *RESTART
Are you sure to restart the system?(Yes/No)? y
```

Configuring the tunnel interfaces:

```
Centrix *P 4

Centrix Config>NETWORK 2
-- IP Tunnel Net Config --
Centrix TNIP config>SET SOURCE 10.129.1.3
Centrix TNIP config>ENABLE TUNNEL
Centrix TNIP config>SET PROTOCOL

-- GRE Config --
Centrix GRE config>ENABLE KEY

Tunnel key: [0]? 333
Centrix GRE config>EXIT
Centrix TNIP config>EXIT
Centrix Config>NETWORK 3
-- IP Tunnel Net Config --
Centrix TNIP config>SET SOURCE 10.129.1.3
Centrix TNIP config>ENABLE TUNNEL
Centrix TNIP config>SET PROTOCOL

-- GRE Config --
Centrix GRE config>ENABLE KEY

Tunnel key: [0]? 333
Centrix GRE config>EXIT
Centrix TNIP config>EXIT
```

Configuring IP protocol:

```
Centrix TNIP config>EXIT
Centrix Config>PROTOCOL IP
Internet protocol user configuration
Centrix Conf IP>ADD ADDRESS 2 0.0.0.2
Centrix Conf IP>ADD ADDRESS 3 0.0.0.3

Centrix Conf IP>GROUP
Centrix GIP>ADD Client1 2
Centrix GIP>ADD Client1 3
Centrix GIP>EXIT

Centrix Conf IP>ADD ISDN 199.6.5.0 255.255.255.0 Client1
ISDN Number[]? 932530280
Cost[1]? 2
Centrix Conf IP>ADD ISDN 199.6.6.0 255.255.255.0 Client1
ISDN Number[]?932530281
Cost[1]? 2
```

List of interfaces:

```
Centrix Config>LIST DEVICE
Con    Ifc Type of interface                  CSR      CSR2  int
---      1 Tunel IP                             0              0
---      2 Tunel IP                             0              0
---      3 Tunel IP                             0              0
---      4 Router->Node                         0              0
---      5 Node->Router                         0              0
LAN      0 Ethernet                       9000000             1C
WAN1     6 X25                            F001600  F000C00    9E
WAN2     7 X25                            F001620  F000D00    9D
ISDN 1   8 channel D: X.25                A000000             1B
ISDN 1  10 channel B: X.25                F001640  F000E00    9C
ISDN 2   9 channel D: X.25                A200000             1B
ISDN 2  11 channel B: X.25                F001660  F000F00    9B
```

List of tunnel interfaces:

```
Centrix Config>NETWORK 2

-- IP Tunnel Net Config --
Centrix TNIP config>LIST ALL
Tunnel Mode: GRE
Tunnel Addresses
Source: 10.129.1.3
Destination: 0.0.0.0
Tunneling IP:   enabled
Centrix TNIP config>SET PROTOCOL

-- GRE Config --
Centrix GRE config>LIST ALL
Cipher:         disabled
GRE Options GRE
End-to-End Checksumming:      disabled
Tunnel identification key:    enabled
--------------------> key:    333
Drop Out-of-Order Datagrams:  disabled
Proprietary sequence number:  disabled
Centrix GRE config>EXIT
```

```
Centrix TNIP config>EXIT
Centrix Config>net 3
-- IP Tunnel Net Config --
Centrix TNIP config>LIST ALL
Tunnel Mode: GRE
Tunnel Addresses
Source: 10.129.1.3
Destination: 0.0.0.0
Tunneling IP:   enabled
Centrix TNIP config>SET PROTOCOL
-- GRE Config --
Centrix GRE config>LIST ALL
Cipher:         disabled
GRE Options GRE
End-to-End Checksumming:      disabled
Tunnel identification key:    enabled
--------------------> key:    333
Drop Out-of-Order Datagrams:  disabled
Proprietary sequence number:  disabled
```

IP protocol list:

```
Centrix Conf IP>LIST ALL
Interface addresses
IP addresses for each interface:
   intf  0 (       *)  10.129.1.3     255.255.255.0   NETWORK broadcast,    fill 0
   intf  1 (       *)     0.0.0.1           0.0.0.0   NETWORK broadcast,    fill 0
   intf  2 (Cliente1)     0.0.0.2           0.0.0.0   NETWORK broadcast,    fill 0
   intf  3 (Cliente1)     0.0.0.3           0.0.0.0   NETWORK broadcast,    fill 0
   intf  4                                            IP disabled on this interface
Routing
route to 192.6.1.0,255.255.255.0 via 10.219.1.2, cost 1 *
route to 0.0.0.0,0.0.0.0 via 10.129.1.1, cost 0 *
route to 199.6.5.0,255.255.255.0 via ISDN 932530150 cost 2 Client1
route to 199.6.6.0,255.255.255.0 via ISDN 932530280 cost 2 Client1
Protocols
Directed broadcasts: enabled
RIP: enabled
OSPF: disabled
Per-packet-multipath: disabled
Ip classless: disabled
```

b) Configuration needed in CBRAs (Client 1 device)

Client device configuration does not need important changes in order to work in scenarios 5 or 6 and given that the default route is the tunnel, no further configuration is necessary except for the Key. The configuration of the devices for this new client is explained in section 4.2.b plus the GRE key:

```
*P 4

Config>NETWORK 2
-- IP Tunnel Net Config --
TNIP config>SET PROTOCOL

-- GRE Config --
GRE config>ENABLE KEY 333
```

5. Events

Due to the reuse of the dynamic tunnels and the ‘wake’ client option, the following events have been added:

The list of added events for the TNIP protocol is the following::

TNIP.021

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

> TNIP.021 Set up tn int *interface_num* type *interface_id* (*source_ip_address->source_ip_address*)

Long Syntax:

> TNIP.021 Interface *interface_num* sets up a tunnel type *interface_id* source address *source_ip_address* and destination *source_ip_address*

Description:

> One of the free dynamic tunnels has been used to create a new tunnel.

TNIP.022

Level: Abnormal external error, ERROR-AE/UE-ERROR

Short Syntax:

> TNIP.022 Err tn with no routes rel, int *interface_num* type *interface_id*

Long Syntax:

> TNIP.022 Error tunnel with no routes release, interface *interface_num* type *interface_id*

Description:

> No routes via RIP have been received in an established dynamic tunnel nor does it have a static route therefore it has been released to be reused.

TNIP.023

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

> TNIP.023 Rele tn int *interface_num* typ *interface_id* Rx idem IP *ip_address*

Long Syntax:

> TNIP.023 Release tunnel interface *interface_num* type *interface_id* set up another tunnel with the same address IP *ip_address*

Description:

> TNIP.023 A new tunnel has been created with the IP address of an existing tunnel and has not been given the release conditions. It needs to be released now.

TNIP.024

Level: Abnormal external error, ERROR-AE/UE-ERROR

Short Syntax:

TNIP.024 Err loop in tunnel, int *interface_num* type *interface_id*

Long Syntax:

TNIP.024 Error loop in tunnel, interface *interface_num* type *interface_id*

Description:

TNIP.024 A packet which has been encapsulated within a tunnel has provoked an endless loop.

The list of events added for the RIP protocol is as follows:

RIP.029

Level: Trace per packet, TRAZA-P/P-TRACE

Short Syntax:

RIP.029 Tn nt int *interface_num* no routes.

Long Syntax:

RIP.029 Tunnel interface *interface_num* released, no routes.

Description:

RIP.029. Deleted routes learned by RIP are checked to see if they are from a tunnel. If there are no routes for the tunnel, the tunnel is released.

6. Monitoring

Further to the non-dynamic tunnel statistics seen in chapter 3, there are other specific statistics for dynamic tunnels:

6.1. Viewing of the monitoring prompt

The TNP monitoring commands are described in this section. In order to access the TNP monitoring environment, you need to follow the subsequent steps:

1. At the GESTCON (*) prompt, enter **PROCESS 3** or (**P 3**).

2. At the MONITOR (+) prompt, enter **NETWORK #,** where # is the interface number corresponding to an IP tunnel.

3. At the IP Tunnel monitoring prompt (TNIP>) enter the required control commands.

Example:

```
*P 3
+NETWORK 2
TNIP protocol monitor
TNIP>
```

6.2. Monitoring Commands

In order to view the interface statistics enter the following from the console menu:

Command	Function
? (HELP)	Lists the commands or their available options
LIST	Displays the IP tunnel interface statistics
EXIT	Exits the TNIP monitoring process

The letters written in **bold** are the minimum number of characters required in order to make the command effective.

a) ?(HELP)

You use the **?(HELP)** command in order to list the commands available at the prompt where you are working. You can also subsequently use this command after a specific command in order to list the available options.

Syntax:

```
TNIP>?
```

Example:

```
TNIP >?
LIST
EXIT
TNIP config>
```

b) LIST

This command is used to list the IP tunnel interface statistics.

Syntax:

```
TNIP>LIST ?
STATE
```

STATE

This displays the status of the tunnel statistics. These statistics are for dynamic tunnels.

Example:

```
TNIP>LIST STATE

Int Source IP      Dest. IP        Start Time  Connections Descon.err  Loop
--- ---------      ----------      ----------  ----------  ----------  ----
 1 195.80.0.2      195.78.0.3         09:10        113              1     0
 2 195.80.0.2      192.0.56.1         13:15         28              0     0
 3 10.125.0.2      10.4.123.3         12:10         36              0     0
```

Start Time: the last connection's initiation time.

Connections: number of connections since the last router restart.

Discon.err: number of disconnections due to not receiving routes through this tunnel.

Loop: informs you if there has been a loop.