# Dynamic Multipoint VPNs

bintec Dm768-I

## Legal Notice

Warranty

This publication is subject to change.

bintec offers no warranty whatsoever for information contained in this manual.

bintec is not liable for any direct, indirect, collateral, consequential or any other damage connected to the delivery, supply or use of this manual.

# Table of Contents

# I Related Documents

bintec Dm710-I PPP Interface

bintec Dm715-I BRS

bintec Dm719-I IP Tunnel Interface (TNIP)

bintec Dm722-I Common Configuration Interfaces

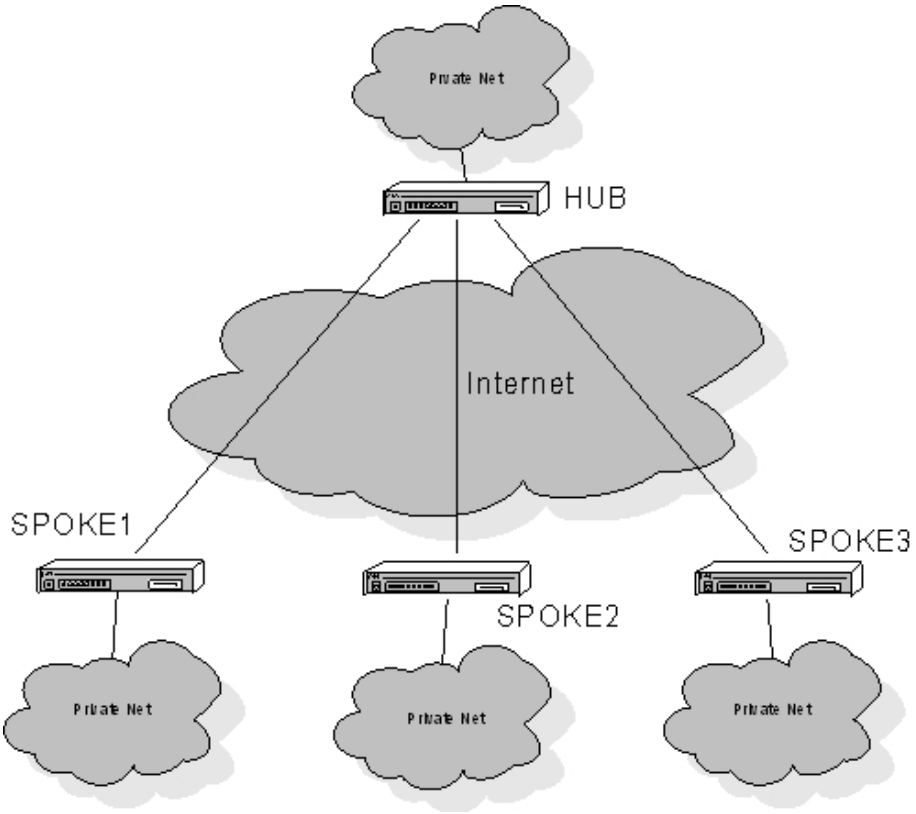
# Chapter 1 Introduction

## 1.1 Introduction to Dynamic Multipoint Networks

This manual describes how IPSec-based dynamic multipoint virtual private networks (DMVPN) work. The goal is to show their usefulness and serve as reference when it comes to creating DMVPNs in our routers.

The aim of DMVPNs is to connect multiple branch offices to a main office and to each other, without traffic between remote sites having to go through the main office. The main advantage of DMVPNs is that they achieve this using encrypted tunnels across public networks (such as the Internet) without the need to lease point-to-point landlines, which are far less economical. Internet access is relatively cheap and most companies already have it in their offices.

Thanks to the IPSec protocol, encrypted point-to-point connections can be set up in public networks without having to sacrifice the integrity and privacy of data. The best way to organize a large IPSec-based Internet network is to use a full or partial mesh network or *hub* and *spoke* scheme. *Hub-and-spoke* is generally chosen because most traffic is between the *spokes* and the *hub*, with the rest tending to be sporadic traffic between *spokes*. The problem with full mesh networks is that they demand more power from the *spokes* (so they can connect to the rest of the *spokes*), which can be high in number in large networks. In a DMVPN topology, the Internet is used to connect the *hub* to the *spokes* and, because the Internet is a mesh network, the *spokes* are able to connect to each other.




Since a connection between the *spokes* is possible, inter-spoke traffic can be transferred directly between the *spokes* without having to go through the *hub*. This saves *hub* resources and the outward and return paths along which information flows are more optimal than those passing through the *hub*. Moreover, since the *hub*'*s* geographical location is unknown, it may be further away than the *spoke* we wish to communicate with.

### 1.1.1 What problems occur when creating a DMVPN?

Deploying this type of network gives rise to certain problems that need to be addressed:

#### 1.1.1.1 IPSec does not support dynamic routing protocols

In order to send the *spoke* routing table to the network, dynamic routing protocols are used. Dynamic routing protocols use *broadcast* and *multicast* packets. The IPSec protocol is point-to-point and only accepts *unicast* packets. To use dynamic routing protocols with IPSec, we turn to the GRE encapsulation protocol (*Generic Routing Encapsulation*). GRE packets only need to know their source and destination and, since they are *unicast*, the IPSec protocol can encrypt them. Given that the GRE protocol encapsulates the packets, there is no need for IPSec to re-encapsulate them and add another IP header (allowing you to use **IPSec in transport mode** and save 20 bytes per packet).

#### 1.1.1.2 Tunnel Source and Destination

Since the GRE protocol needs to know the source and destination for each tunnel, configuring *hubs* in a large network can be very time-consuming. This is because the *hub* must know the addresses for all the *spokes* it services. In the same way, if you want a *spoke* to be able to connect to any other *spoke* in the network, this means you need to create a completely meshed network (which will cause the *spoke* configuration too become overly complex).

#### 1.1.1.3 Dynamic IP Addresses

Internet connections usually use dynamic IP addresses obtained from a server through DHCP or PPP. This complicates the scheme even further and makes it impossible to know the tunnel destination address in advance. As already mentioned, the GRE protocol needs to know these addresses *a priori* to create the tunnel between the source and destination endpoints.

To solve these problems, DMVPNs use two protocols: Multipoint GRE (mGre) and NHRP (Next Hop Resolution Protocol). The former allows you to create multipoint dynamic tunnels, while the latter helps find the tunnel destination address (next hop address).

### 1.1.2 NHRP Protocol

The NHRP our devices implement complies with the *RFC2332 NBMA Next Hop Resolution Protocol (NHRP)* standard.

NHRP simplifies device configuration for spokes and hubs. Devices that behave as hubs no longer need the addresses of any of the network spokes to be configured. As a result, spoke addresses can be dynamically assigned. Spokes only need one or several hub addresses to be configured.

On start-up, each spoke registers with its configured hub, thereby establishing a permanent tunnel between each spoke and the hub. In turn, each spoke is configured with a *multicast* address to which it sends packets from the dynamic routing protocol it has configured. The *multicast* address is usually the address of the hub configured on the spoke. This way, the spoke informs the hub of its network routes using a dynamic routing protocol. For its part, the hub creates a *multicast* entry for each registered spoke and shares the routes with all the registered network spokes. This means all *spokes* can access each other's networks, as in a fully meshed network, without complicating the *spoke* configurations.

Tunnels between spokes in a DMVPN are dynamically established. The hub acts as an NHRP server for preregistered spokes. To establish a tunnel between two spokes, spoke1 sends a request to the hub to find out the public address for spoke2, which it wants to connect to (tunnel destination address). If spoke2 is registered in the hub, the latter will send the requested information. When spoke2 receives a request form spoke1 to establish a tunnel, it also requests information from the hub for the next hop to spoke1. Subsequently, both spokes negotiate the creation of an IPSec tunnel over an mGre interface (multipoint GRE) and establish communications. When the tunnel is no longer needed by the spokes, it is torn down after a user-configurable inactivity period. The tunnel that has been created directly connects both spokes, but the hub only intervenes to facilitate destination addresses so the spokes can create a direct tunnel between them. This frees the hub from having to perform any additional routing and encryption tasks.

NHRP on each device maintains the next-hop information (public address of the interface for a known private address) for each tunnel activated by the device in a cache. A hub contains information on all the spokes that have registered with it. A spoke has information on each of the hubs it has registered with and information requested from the hubs to establish tunnels with other spokes.

### 1.1.3 NHRP Protocol Feature

DMVPNs are configured over a TNIP (IP tunnel) virtual tunnel interface. The TNIP interface has its own IP address, over which mGre is configured. The operating parameters for NHRP, which provides service for mGre, are configured in the TNIP interface. The TNIP interface is described in *bintec* manual *Dm719-IP Tunnel Interface (TNIP)* .

The TNIP interface is configured with a private IP address and, in turn, has a public IP address (which is the physical interface address over which the tunnel is established) associated with it. NHRP is used to discover the TNIP interface public address of the remote device that the protocol wishes to establish communications with through a tunnel. The private address is already known thanks to the dynamic routing protocol running on the network:

- Each spoke periodically registers with the configured hub(s) in order to keep any established tunnels active.
- Multicast packets from the dynamic routing protocol travel through these tunnels.
- The hub sends the routes from the rest of the DMVPN devices to the spokes through the tunnels, i.e., it transmits the next hop (which usually matches the private address).

To discover a tunnel destination IP address, mGre sends a request to the NHRP with the next hop private address. NHRP checks its cache for an available entry with this private IP address. If an entry is found, it returns the public IP

address it has stored and mGre creates the tunnel. If an entry is not found, NHRP generates a resolution request packet to the active primary *hub* (NHS *Next Hop Server*) that it has configured. While it waits for the response, the NHS returns its own public address to the mGre protocol. This prevents the packet from getting lost while waiting for the hub to respond to the resolution request. While waiting for the address that would allow an optimum *spoke-to-spoke* path, packets are transported from the spoke or the hub and, finally, to another spoke ( *spoke-hub-spoke* path). As soon as the response is received from the hub, a tunnel is created between the *spokes* trying to send a packet.

As soon as a dynamically-established tunnel between two spokes ceases to be useful and reaches the pre-configured inactivity threshold, it expires and the entries that were created with the tunnel are deleted from the spokes' cache.

### 1.1.4 Types of NHRP Packets

Seven types of packets are possible in NHRP and travel between the NHCs (*Next Hop clients*) and the NHSs (*Next Hop Servers*):

(1) **Registration Request**: NHC registration request to the NHS.

(2) **Registration Reply**: NHS response to the NHC registration request.

(3) **Resolution Request**: resolution request for the next hop address that the NHC sends to the NHS.

(4) **Resolution Reply**: NHS response to the NHC with the requested next hop address.

(5) **Purge Request**: purge request for a cache entry which the NHS sends to the NHC when the information on said entry is no longer valid.

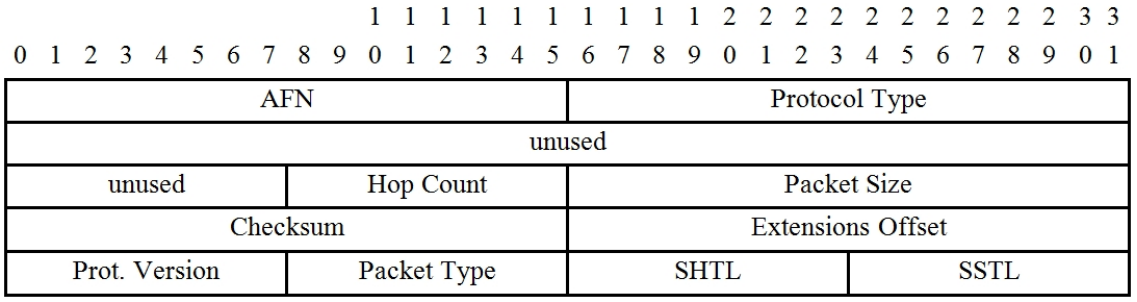
(6) **Purge Reply**: NHS response to the NHC regarding a purge request for a cache entry.

(7) **Error Indication**: error packet indicating some sort of problem in one of the packets received in the device generating the error packet.

NHRP packets are made up of a fixed header, a mandatory part and a part with optional extensions.

The packet's structure is as follows:

#### 1.1.4.1 Fixed part




**AFN**

Indicates the type of link layer address. In this case, the value is 0x1 (i.e., addresses with IPv4 format)

**Protocol Type**

0x0800, which stands for Ethertype.

**Hop Count**

This is the maximum number of NHSs a packet can go through before being dropped.

**Packet Size**

This is the size of the NHRP packet in octets.

**Checksum**

This is the IP checksum over the entire NHRP packet.

**Extensions Offset**

If this is 0, it indicates that the packet does not have extensions. If this is nonzero, this is the distance in octets from the NHRP packet header to the beginning of the extensions.

**Protocol Version**

Version of NHRP. In our case this is 1, indicating this is the version defined in RFC 2332.

**Packet Type**

NHRP Resolution Request (1), NHRP Resolution Reply (2), NHRP Registration Request (3), NHRP Registration Reply (4), NHRP Purge Request (5), NHRP Purge Reply (6), or NHRP Error Indication (7).

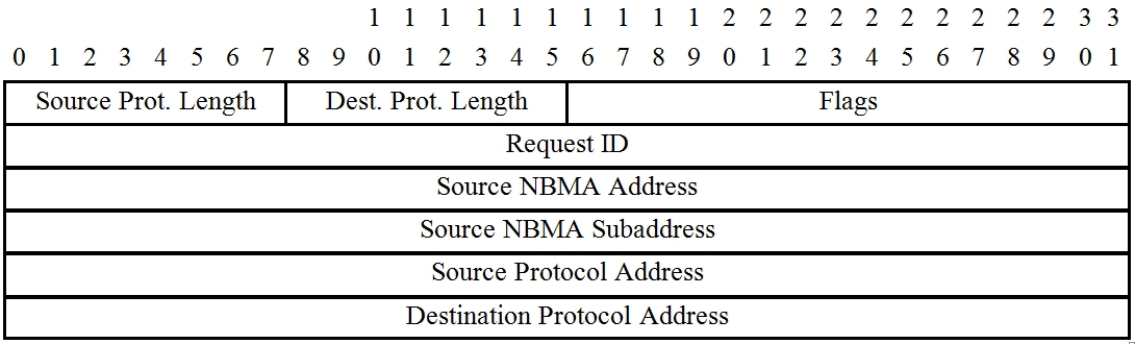
**SHTL**

Type and length of the source NBMA (*Non Broadcast Multiaccess Network*) address. For IPv4 addresses, this value is 4. If this is 0, the packet does not have a *Source NBMA Address* field in the mandatory part.

**SSTL**

For IPv4 addresses, this is always 0. This implies that the *Source NBMA subaddress* field in the mandatory part does not exist in any packet.

### 1.1.4.2 Mandatory Part




**Source Protocol Length**

Length in octets of the source private address. For IPv4 addresses, this value is always 4. If it is 0, the *Source Protocol Length* does not exist in the packet.

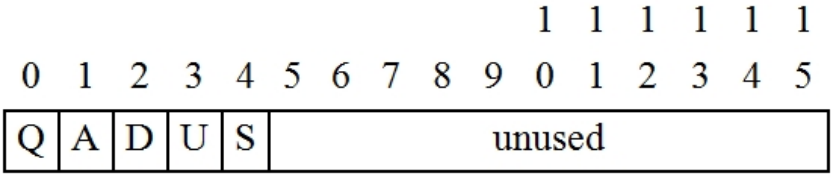
**Destination Protocol Length**

Length in octets of the destination private address. For IPv4 addresses, this value is always 4. If it is 0, the *Destination Protocol Length* does not exist in the packet.

**Flags**

These flags depend on the type of packet:

- Resolution Request:




**Q**: indicates that the device generating the *Resolution Request* is a *router*.

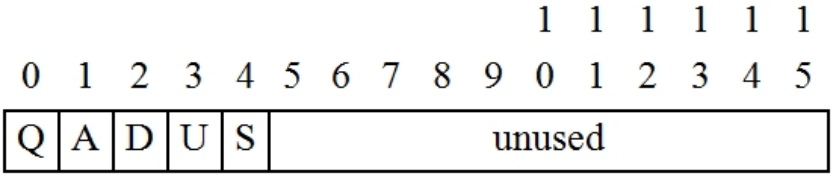
**A**: the requested information must be authorized, obtained from a registration packet in the NHS.

**D**: not used, this should be set to 0.

**U**: indicates that the required information must be unique. A unique CIE (client information entry) must be obtained.

**S**: the device information on the packet is stable and accurate.

- Resolution Reply:




**Q**: value copied from the *Resolution Request*.

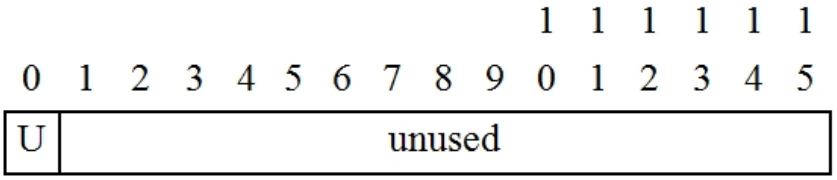
**A**: the information included in the packet CIE is authorized, obtained from a registration packet in the NHS.

**D**: the information the CIE contains is stable and its value is guaranteed during the time specified in the CI Holding Time field.

**U**: value copied from the Resolution Request.

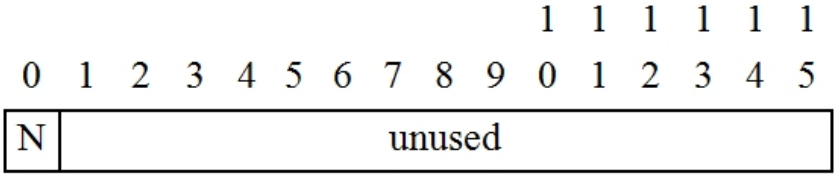
**S**: value copied from the Resolution Request.

- Registration Request:




**U**: indicates that the information we are registering is unique.

- Purge Request:




**N**: if this is 1, the device generating the packet is not expecting a *Purge Reply* packet as a response.

**Request ID**

Together with the source address value, this helps identify the NHRP packet.

**Source NBMA Address**

This is the source public address (i.e., the address of the device that has been sent in the *request* packet). If the *SHTL* field in the fixed part of the NHRP packet is zero, then the field for this address will not exist in the packet.

**Source NBMA Subaddress**

For IPv4 addresses, this field does not exist in the packet. The *SSTL* field in the fixed part of the NHRP packet is 0.

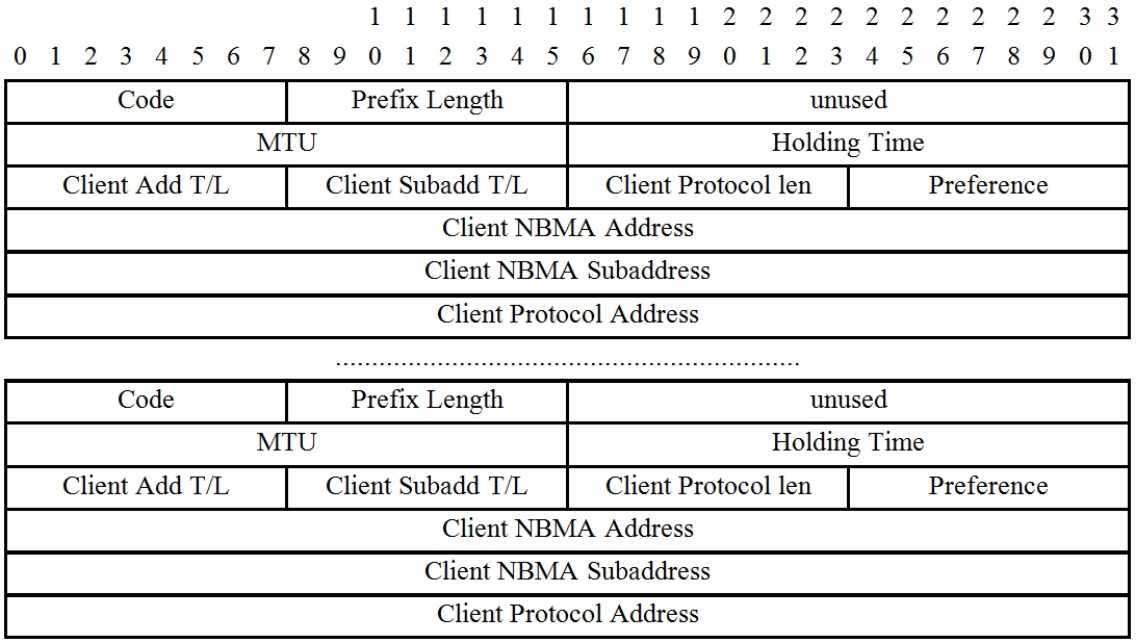
Source Protocol Address

This is the source private address of the *request* packet and the destination private address of a *reply* packet.

**Destination Protocol Address**

This is the destination private address for a *request* packet.

### 1.1.4.3 CIE's (Client Information Entries)




**Code**

In *request* packets this is 0. In *reply*, a positive acknowledgement (ACK) takes the value 0 and a negative acknowledgement (NAK) any other value. For instance:

0. Successful Registration.

1. Unrecognized Extension.

3. NHRP Loop Detected.

4. Administratively Prohibited.

5. Insufficient Resources.

6. Protocol Address Unreachable.

7. Protocol Error.

8. NHRP SDU Size Exceeded.

9. Invalid Extension.

10. Invalid NHRP Resolution Reply Received.

11. Authentication Failure.

12. No Internetworking Layer Address to NBMA Address Binding Exists.

13. Binding Exists But Is Not Unique.

14. Unique Internetworking Layer Address Already Registered.

15. Hop Count Exceeded.

**Prefix Length**

If this is not 0 and the 0xFF is different, the information the CIE contains corresponds to a network and is the number of one of the network masks. If this is 0, this address corresponds to a single device.

**MTU**

This value shows the maximum transmission unit for the client device. In this case, the default value is 1514 octets.

**Holding Time**

Specifies the number of seconds during which the information for the next hop contained in the CIE is considered valid. When this times out, this information must be deleted from the cache. This must be 0 for a NAK.

**Client Address Type & Length**

This is the *Client NBMA Address* field length in octets. For IPv4 addresses, this is 4 if the *Client NBMA Address* field appears in the CIE and 0 if it doesn't.

**Client Subaddress Type & Length**

In our example, this is always 0 and there is no *Client NBMA Subaddress* field in the CIE.

**Client Protocol Length**

This is the *Client Protocol Address* field length in octets. For IPv4addresses, this is 4 if the *Client Protocol Address* field appears in the CIE and 0 if it doesn't.

**Preference**

When the NHRP packet contains several CIEs, this indicates the preference for one with respect to the others. A higher value indicates greater preference.

**Client NBMA Address**

Public address for the next hop.

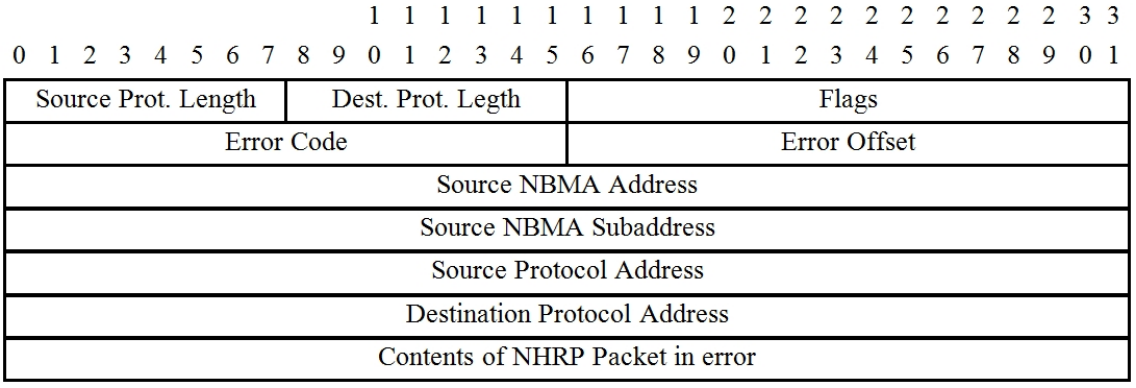
**Client NBMA Subaddress**

This field does not exist for IPv4 addresses (our example).

**Client Protocol Address**

Private address for the next hop.

### 1.1.4.4 Error Packet Mandatory Part




**Error Code**

Contains the code for the error caused. The possible values are: 1, 3, 6, 7, 8, 9, 10, 11, 15. To view the error messages that correspond to each code, check the CIE *Code* field found in the mandatory part of the NHRP packet.

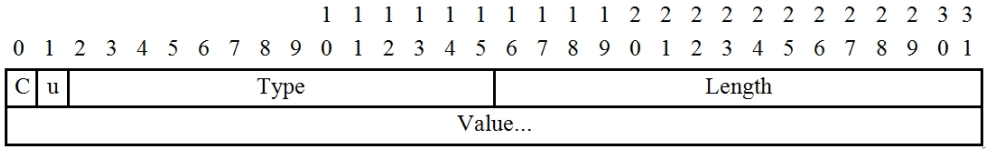
**Error Offset**

This is the *offset* in octets from the beginning of the fixed part of the original NHRP packet, where the error has been observed, up to the erroneous field in the original packet.

**Contents of NHRP Packet in error**

The full contents of the packet where an error has been observed, from the fixed part up to the extensions.

#### 1.1.4.5 Extensions




**Compulsory bit**

Indicates that the extension is mandatory and cannot be ignored.

**Type**

Type of extension:

0. The End of Extensions.

3. Responder Address Extension.

4. NHRP Forward Transit NHS Record Extension.

5. NHRP Reverse Transit NHS Record Extension.

7. NHRP Authentication Extension.

8. NHRP Vendor-Private Extension.

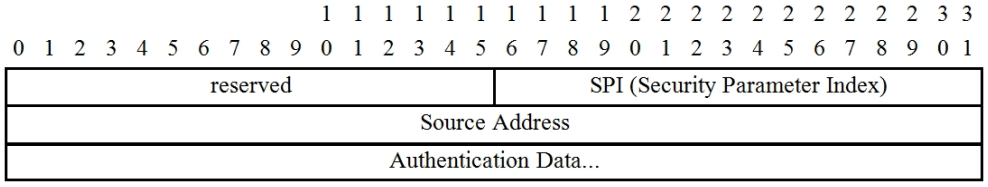
**Length**

Length in octets of the extension *Value* field (excluding the *Type* and *Length* fields).

**Value**

Depends on the type of extension. The *End of Extensions* does not have a *Value* field. *The Responder Address Extension* includes a CIE with the addresses of the device generating the *reply* packet. The *Forward Transit NHS Record Extension* has a CIE for each NHS the packet has passed through on the outward route (*request* packet). The *Reverse Transit NHS Record Extension* includes a CIE for each NHS the packet has passed through on the return route (*reply* packet). In our example, the *Vendor-Private Extension* does not exist. The content of the *Authentication Extension Value* field is detailed in the following point.

#### 1.1.4.6 Authentication Extension Value




**SPI**

This is the *Security Parameter Index*. In our case this is 1, indicating that the authentication is carried out with an 8-character authentication password.

**Source Address**

In our example, this field does not exist.

**Authentication Data**

8-character password functioning as authentication password.

### 1.1.5 Per-Tunnel QoS for DMVPN feature

Thanks to the Per-Tunnel QoS designed for the DMVPN feature, a quality of service (QoS) policy can be applied on a DMVPN *hub* (on a per-tunnel instance in the egress direction for DMVPN *hub-to-spoke* tunnels). This way, tunnel traffic can be shaped to accommodate individual *spokes* (a parent policy) and identify individual data flows going through the tunnel for policing (a child policy).

The QoS policy that the *hub* uses for a particular *spoke* is selected by the *NHRP group* in which the *spoke* is con-

figured.

The *NHRP group* is communicated to the *hub* in each of the periodic NHRP *registration requests* sent from the *spoke* to the *hub*.

NHRP group-to-QoS policy mappings are configured on the hub. The NHRP group string coming from a *spoke* is mapped to a QoS policy, which is applied to that *hub-to-spoke* tunnel in the egress direction.

If a NHRP group is not received from the *spoke*, then a QoS policy is not applied. Moreover, any existing QoS policy previously applied to that *spoke* is removed.

Note that, for the moment, this feature is only implemented on the *spoke* side.

## 1.2 References

RFC-2332: NBMA Next Hop Resolution Protocol (NHRP), April 1998.

*bintec Dm719-I IP Tunnel Interface (TNIP)* .

# Chapter 2 Configuration

## 2.1 Configuring a DMVPN

This section describes how to configure a DMVPN on our devices. To access the configuration menu, execute the **Config** or **PROCESS 4** command over the device's console root menu.

```
*PROCESS 4
Config>
```

To create a DMVPN, create a TNIP or IP tunnel virtual interface on each device. To create an IP Tunnel interface, enter **ADD DEVICE tnip <tunnel identifier>** in the global configuration menu.

```
Config>ADD DEVICE tnip 1
Added TNIP interface tnip1
Config>
```

Once this has been set up, simply enter **NETWORK tnipX** to access the configuration. **X** represents the tunnel identifier:

```
Config>NETWORK tnip1
-- IP Tunnel Net Configuration --
tnip1 config>
```

The protocol supported over the TNIP interface is IP. To activate the IP over the TNIP interface, assign an IP address to the selected interface or configure it as an unnumbered interface.

Example with a known IP address:

```
tnip1 config>ip address 5.5.5.1 255.255.0.0
tnip1 config>
```

Example with an unnumbered interface:

```
tnip1 config>ip address unnumbered
tnip1 config>
```

## 2.2 Configuring the IP Tunnel interface (TNIP)

This section describes the tnip interface configuration commands. To access the TNIP configuration environment, enter **NETWORK <tnip interface>.**

```
*P 4
Config>NETWORK tnip1
-- IP Tunnel Net Configuration --
tnip1 config>?
  description           Enter interface description
  destination           Destination address
  disable               Disable the tunnel interface
  enable                Enable the tunnel interface
  encapsulation         Encapsulation configuration
  ip                    Interface Internet Protocol config commands
  keepalive             Enable keepalive
  list                  Show tunnel interface configuration
  mode                  Encapsulation mode for the tunnel interface
  nhrp                  NHRP protocol configuration
  nhrp-label            Mark NHRP packets with an internal label
  nhrp-tos              Mark NHRP packets with a TOS value
  no
  path-mtu-discovery    Enable Path MTU Discovery on tunnel
  qos-pre-classify      QoS pre-classify
  shutdown              Change state to administratively down
  source                Source address or source interface
  update                Update a level indicator
  vrf-encap             Specify parameters for a VPN Routing/Forwarding
                        instance
  exit
```

```
tnip1 config>
```

All device interfaces have certain commands that are the same. These commands are described in *bintec* manual *Dm722-I Common Configuration Interfaces*.

The available commands are as follows:

| Command | Function |
|---|---|
| *? (HELP)* | Lists the available commands or their options. |
| *DESTINATION* | Configures the tunnel destination IP address. |
| *DISABLE* | Disables the tunnel interface. |
| *ENABLE* | Enables the tunnel interface. |
| *ENCAPSULATION* | Accesses the encapsulation protocol configuration menu. |
| *KEEPALIVE* | Enables *keepalive* maintenance. |
| *LIST* | Displays the configured parameters. |
| *MODE* | Selects the encapsulation mode in the tunnel interface (encapsulation protocol). |
| *NHRP* | NHRP configuration commands. |
| *NHRP-LABEL* | Marks NHRP packets with an internal label. |
| *NHRP-TOS* | Marks NHRP packets with a TOS value. |
| *NO* | Disables or eliminates functionalities. |
| *PATH-MTU-DISCOVERY* | Enables *Path MTU Discovery* in the tunnel. |
| *QOS-PRE-CLASSIFY* | Enables preclassification of BRS packets. |
| *SOURCE* | Configures tunnel source IP address. |
| *VRF-ENCAP* | Specifies parameters for a *VPN Routing/Forwarding* instance. |
| *EXIT* | Exits the TNIP configuration menu. |

### 2.2.1 DESTINATION

This command should only be activated in scenarios where point-to-point IP tunnels are used. It must not be used in DMVPN configurations with NHRP.

### 2.2.2 DISABLE

Disables the tunnel interface. The tunnel interface is disabled by default.

*Syntax:*

```
tnip1 config>DISABLE ?
  <cr>
tnip1 config>
```

*Example:*

```
tnip1 config>DISABLE
tnip1 config>
```

### 2.2.3 ENABLE

Enables the tunnel interface. The tunnel interface is disabled by default.

*Syntax:*

```
tnip1 config>ENABLE ?
  <cr>
tnip1 config>
```

*Example:*

```
tnip1 config>ENABLE
tnip1 config>
```

### 2.2.4 ENCAPSULATION

Accesses the encapsulation protocol's configuration. The only encapsulation protocol currently supported is GRE (*Generic Routing Encapsulation*). This submenu is described in section 3.

*Syntax:*

```
tnip1 config>ENCAPSULATION ?
  <cr>
tnip1 GRE config>
```

*Example:*

```
tnip1 config>ENCAPSULATION
-- GRE Configuration --
tnip1 GRE config>
```

### 2.2.5 KEEPALIVE

This command should only be activated in scenarios where point-to-point IP tunnels are used. It must not be used in DMVPN configurations with NHRP.

NHRP manages the state of the links through the information stored in its cache. As a result, whether the tnip interface is UP or DOWN (after executing the **device tnipx** monitoring command where x is the tnip interface number) is irrelevant. A link appears if information in the cache can resolve the public address from the private address of the packet destination device (next hop), or if this information can be obtained from the next-hop server (NHS).

### 2.2.6 LIST

Displays the IP tunnel configuration.

*Syntax:*

```
tnip1 config>LIST ?
  <cr>
tnip1 config>
```

*Example:*

```
tnip1 config>LIST
Tunnel mode: GRE (enabled)
Tunnel source 212.95.195.132, destination 66.187.232.56
QoS preclassify: disabled
Keepalive enabled with period 10, 3 retries
NHRP type of service: 25
tnip1 config>
```

| | |
|---|---|
| *Tunnel mode:* | Indicates the type of encapsulation and the status (enabled/disabled). |
| *Tunnel source / destination:* | Tunnel source / destination IP addresses. |
| *QoS preclassify:* | Indicates if the BRS pre-classification is enabled. |
| *Keepalive:* | Displays the keepalive maintenance configuration. |
| *NHRP type of service:* | Displays the type of service selected for the NHRP packets. |

### 2.2.7 MODE

Selects the encapsulation mode. GRE (*Generic Routing Encapsulation*) and mGre (*Multipoint GRE*) are supported.

GRE mode is only logical in point-to-point IP tunnels. Configure **mGRE** mode in configurations with DMVPN and NHRP.

*Syntax:*

```
tnip1 config>MODE ?
  gre    Generic Routing Encapsulation Protocol
tnip1 config>MODE GRE ?
  ip           Over IP
  multipoint   Over IP (multipoint)
```

```
  <cr>
tnip1 config>
```

*Example 1:*

```
tnip1 config>MODE GRE IP
tnip1 config>
```

*Example 2:*

```
tnip1 config>MODE GRE MULTIPOINT
tnip1 config>
```

### 2.2.8 NHRP

Configures NHRP. For further information, please see *Configuring the NHRP protocol* on page 19.

*Syntax:*

```
tnip1 config>NHRP ?
  authentication    Authentication string
  enable            Enable NHRP protocol
  group             Group name string
  holdtime          Advertised holdtime
  ignore-purge-req  Do not process purge request packets
  list              List NHRP protocol's configuration
  map               Map dest IP addresses to NBMA addresses
  nhs               Specify a next hop server
  rate-limit        Rate limit NHRP traffic
  record            Enable NHRP transit record extensions
  registration      Change registration mode
  responder         Responder interface
  server-only       Disable NHRP requests
  track-status      The state of IP static routes depends on NHRP register status
  use               Minimum use for sending requests
tnip1 config>
```

### 2.2.9 NHRP-LABEL

Marks NHRP packets with an internal label. This way, NHRP packets can be filtered or classified in a manner that, for example, be allows them to be prioritized by QoS policies.

*Syntax:*

```
tnip1 config>NHRP-LABEL ?
  <0..65535>    Internal label value
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP-LABEL 99
tnip1 config>
```

**Command history:**

| Release | Modification |
|---|---|
| 11.01.07 | The "*NHRP-LABEL*" command was introduced as of version 11.01.07. |

### 2.2.10 NHRP-TOS

Marks NHRP packets with a TOS value (*type of service*). Thus, NHRP packets can be filtered in order to, for example, prevent NHRP registration request packets from launching calls in the UMTS link.

*Syntax:*

```
tnip1 config>NHRP-TOS ?
  <0..255>    IPv4 type of service value
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP-TOS 25
tnip1 config>
```

### 2.2.11 PATH-MTU-DISCOVERY

Activates the feature that finds the most appropriate MTU value so that no packet fragmentation occurs on the packets sent from one end of the tunnel to the other. This feature identifies the minimum MTU on the path between the two tunnel ends and prevents fragmentation. On activating said feature, the DF (*don't fragment*) bit in the sent TCP/IP packets is enabled.

*Syntax:*

```
tnip1 config>PATH-MTU-DISCOVERY ?
  <cr>
tnip1 config>
```

*Example:*

```
tnip1 config>PATH-MTU-DISCOVERY
tnip1 config>
```

### 2.2.12 QOS-PRE-CLASSIFY

Allows the BRS to pre-classify packets. When enabling this option, the BRS (please see *bintec* manual *Dm715-I BRS*) classifies packets that reach the tunnel before they are encapsulated. This means different types of IP traffic sent through the tunnel can be differentiated. If this option is disabled, packets are classified after being encapsulated. If this occurs, all traffic processed by the tunnel have the same IP header (added by the tunnel) and are classified under the same BRS class.

To disable this parameter, enter **no qos-pre-classify.**

*Syntax:*

```
tnip1 config>QOS-PRE-CLASSIFY ?
  <cr>
tnip1 config>
```

*Example:*

```
tnip1 config>QOS-PRE-CLASSIFY
tnip1 config>
```

### 2.2.13 SOURCE

Configures the interface or the source IP address for the IP tunnel. For IP addresses, these must match an IP address of one of the interfaces configured on the router (Ethernet, PPP, Loopback, etc.), **except** for the tunnel's IP address. Where an interface is specified as the tunnel source, this should be one of the interfaces configured on the router (except that of the tunnel itself).

If the tunnel source IP address does not match any of the router's interfaces, the router considers that packets arriving destined for this IP address are not meant for it (the router) and tries to route them to another device.

If the tunnel source is a PPP interface that receives dynamically assigned IP addresses (please see *bintec* manual *Dm710-I PPP Interface*), then you need to specify said PPP interface as the tunnel source.

*Syntax:*

```
tnip1 config>SOURCE ?
  <a.b.c.d>      Tunnel source address
  <interface>    Tunnel source interface
tnip1 config>
```

*Example 1:*

```
tnip1 config>SOURCE 212.95.195.132
tnip1 config>
```

*Example 2:*

```
tnip1 config>SOURCE ppp1
```

```
tnip1 config>
```

### 2.2.14 VRF-ENCAP

Associates the IP tunnel destination address with a VRF instance. The route for said destination is checked in the associated VRF routing tables. Both the tunnel source and destination must be in the same VRF.

*Syntax:*

```
tnip1 config>VRF-ENCAP ?
  <1..32 chars>    VPN Routing/Forwarding instance name
tnip1 config>
```

*Example:*

```
tnip1 config>VRF-ENCAP thisIsAnExample
tnip1 config>
```

## 2.3 Configuring the GRE protocol

This section describes the configuration commands for the Generic Routing Encapsulation protocol (GRE). To access the GRE configuration environment, enter **encapsulation** in the tunnel interface configuration menu (with the interface configured in GRE mode).

```
*P 4
Config>NETWORK tnip1
-- IP Tunnel Net Configuration --
TNIP config>ENCAPSULATION
-- GRE Configuration --
GRE config>?
  checksum              End-to-end checksum
  cipher                RC4 Ciphering
  cipher-key            Cipher key
  key                   ID key for the tunnel interface
  list                  Show GRE configuration
  no
  sequence-datagrams    Drop out-of-order datagrams
  exit
GRE config>
```

The available commands are as follows:

| Command | Function |
|---|---|
| *CHECKSUM* | Enables end-to-end checksum (GRE). |
| *CIPHER* | Enables RC4 cipher in the GRE tunnel. |
| *CIPHER-KEY* | Configures RC4 cipher key. |
| *KEY* | Configures tunnel identifier. |
| *LIST* | Displays the configured parameters. |
| *SEQUENCE-DATAGRAMS* | Drops datagrams received out of order. |

### 2.3.1 CHECKSUM

Enables the option to send a checksum in a GRE packet. By default, the tunnel does not guarantee packet integrity. With this option enabled, the router sends GRE packets with a checksum field. If a packet is received containing a checksum field, the latter is always verified. Packets with an invalid checksum are dropped (regardless of whether or not this option is enabled on the router).

To disable this option, enter **no checksum**.

*Syntax:*

```
tnip1 GRE config>CHECKSUM ?
  <cr>
tnip1 GRE config>
```

*Example:*

```
tnip1 GRE config>CHECKSUM
tnip1 GRE config>
```

### 2.3.2 CIPHER

Activates the RC4 cipher for packets encapsulated in the GRE tunnel. The cipher is disabled by default.

To disable the RC4 cipher, enter **no cipher**.

*Syntax:*

```
tnip1 GRE config>CIPHER ?
  <cr>
tnip1 GRE config>
```

*Example:*

```
tnip1 GRE config>CIPHER
tnip1 GRE config>
```

### 2.3.3 CIPHER-KEY

Configures the tunnel interface cipher key. Said key admits a maximum of 32 alphanumeric characters. When displaying the configuration, the key appears encrypted.

To reestablish the default cipher in GRE tunnels, enter **no cipher-key**.

*Syntax:*

```
tnip1 GRE config>CIPHER-KEY ?
  ciphered         Enter a ciphered key
  <1..32 chars>    Text
tnip1 GRE config>
```

*Example:*

```
tnip1 GRE config>CIPHER-KEY example
tnip1 GRE config>show config
          cipher-key ciphered 0x92B067B7AEBE3742
```

#### 2.3.3.1 CIPHERED

Enables the setting of a ciphered key. When displaying the configuration, the key appears encrypted (i.e., as entered).

*Example:*

```
tnip1 GRE config>cipher-key ciphered 0x4D0279AF2D5AB11D
tnip1 GRE config>show config
          cipher-key ciphered 0x4D0279AF2D5AB11D
```

**Command history:**

| Release | Modification |
|---|---|
| 11.01.04 | Ciphered option was introduced as of version 11.01.04 |
| 11.00.05.10.05 | Ciphered option was introduced as of version 11.00.05.10.05 |
| 10.08.34.05.15 | Ciphered option was introduced as of version 10.08.34.05.15 |

### 2.3.4 KEY

Enables 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. The identifier is a whole number between 0 and 4294967295 (32 bits). This option is disabled on the tunnel by default.

When the tunnel identifier is enabled, the router drops any packets arriving with a different identifier to the one configured.

Each tunnel must have a unique identifier. If you configure two or more tunnels, this identifier must be configured on each tunnel (using the **key** command) and must be different for each one.

*Syntax:*

```
tnip1 GRE config>KEY ?
  plain              ID plain key for the tunnel interface
  ciphered           ID ciphered key for the tunnel interface
  <0..4294967295>    Value in the specified range
tnip1 GRE config>
```

*Example:*

```
tnip1 GRE config>KEY 5
tnip1 GRE config>
```

#### 2.3.4.1 PLAIN

Enables setting a plain key. When displaying the configuration, the key appears encrypted. However, note the difference when using the **key <0..4294967295>** command, where the entered key is not shown encrypted.

*Example:*

```
tnip1 GRE config>key plain 5
tnip1 GRE config>show config

         key ciphered 0x0E4DCE3F8B680B09
```

**Command history:**

| Release | Modification |
|---|---|
| 11.01.04 | Plain option was introduced as of version 11.01.04 |
| 11.00.05.10.05 | Plain option was introduced as of version 11.00.05.10.05 |
| 10.08.34.05.15 | Plain option was introduced as of version 10.08.34.05.15 |

#### 2.3.4.2 CIPHERED

Enables setting a ciphered key. When displaying the configuration, the key appears encrypted (i.e., as entered).

*Example:*

```
tnip1 GRE config>key ciphered  0x0E4DCE3F8B680B09
tnip1 GRE config>show config

         key ciphered 0x0E4DCE3F8B680B09
```

**Command history:**

| Release | Modification |
|---|---|
| 11.01.04 | Ciphered option was introduced as of version 11.01.04 |
| 11.00.05.10.05 | Ciphered option was introduced as of version 11.00.05.10.05 |
| 10.08.34.05.15 | Ciphered option was introduced as of version 10.08.34.05.15 |

### 2.3.5 LIST

Displays the GRE protocol configuration.

*Syntax:*

```
tnip1 GRE config>LIST ?
  <cr>
tnip1 GRE config>
```

*Example:*

```
tnip1 GRE config>LIST
RC4 Cipher.................: enabled
End-to-End Checksumming....: enabled
```

```
Tunnel identification key..: enabled [5]
Drop Out-of-Order Datagrams: disabled
tnip1 GRE config>
```

*RC4 Cipher:* indicates if RC4 cipher is enabled.

*End-to-End Checksumming:* indicates if end-to-end checksum is enabled.

*Tunnel identification key:* tunnel identifier (if this is enabled).

*Drop Out-of-Order Datagrams:* drops datagrams received out of order.

### 2.3.6 SEQUENCE-DATAGRAMS

Enables the option to guarantee data packets arrive in order. On enabling this option, the router checks the sequence number included in the GRE header and drops any packets that arrive out of order. Default is disabled.

To disable the sequence number, enter **no sequence-datagrams.**

*Syntax:*

```
tnip1 GRE config>SEQUENCE-DATAGRAMS ?
  <cr>
tnip1 GRE config>
```

*Example:*

```
tnip1 GRE config>SEQUENCE-DATAGRAMS
tnip1 GRE config>
```

## 2.4 Configuring the NHRP protocol

This section describes the configuration commands for NHRP. To access the NHRP configuration environment, enter **network <tnip interface>.**

```
*P 4
Config>NETWORK tnip1
-- IP Tunnel Net Configuration --
tnip1 config>nhrp ?
  authentication    Authentication string
  enable            Enable NHRP protocol
  group             Group name string
  holdtime          Advertised holdtime
  ignore-purge-req  Do not process purge request packets
  list              List NHRP protocol's configuration
  map               Map dest IP addresses to NBMA addresses
  nhs               Specify a next hop server
  rate-limit        Rate limit NHRP traffic
  record            Enable NHRP transit record extensions
  registration      Change registration mode
  responder         Responder interface
  server-only       Disable NHRP requests
  track-status      The state of IP static routes depends on NHRP register status
  use               Minimum use for sending requests
tnip1 config>
```

To enter an NHRP command, the command should be preceded by the word **nhrp** as follows: **nhrp <command>**. Once in the IP tunnel interface configuration menu, the available NHRP commands are as follows:

| Command | Function |
|---|---|
| *? (HELP)* | Lists the available commands or their options. |
| *AUTHENTICATION* | Specifies the authentication password. |
| *ENABLE* | Enables the NHRP protocol. |
| *GROUP* | Specifies an NHRP group on the spoke. |
| *HOLDTIME* | Information validity time for the next hop sent by the NHRP. |
| *IGNORE-PURGE-REQ* | Makes the router ignore NHRP *Purge request* packets and only respond to them, |

| | |
|---|---|
| | without searching the cache for the entry to delete. |
| *LIST* | Displays the configured parameters. |
| *MAP* | Creates a static entry in the NHRP cache. |
| *NHS* | Specifies the private address for a *Next Hop Server* (NHS) |
| *RATE-LIMIT* | Specifies a maximum transfer rate for NHRP packets. |
| *RECORD* | Enables *NHRP Transit Record Extensions* |
| *REGISTRATION* | Changes the registration mode in an NHS permitting multiple entries. |
| *RESPONDER* | Specifies the interface that answers the NHRP requests. |
| *SERVER-ONLY* | Disables the generation of Resolution Request packets and the cache. |
| *TRACK-STATUS* | Specifies if the state of the static routes depends on the state of the NHRP register. |
| *USE* | Minimum number of packets needed to set up a tunnel. |

### 2.4.1 AUTHENTICATION

Lets you enter a configuration word of up to 8 characters. The aim of the configuration word is to stop other devices, which aren't configured with the same configuration word, from communicating through NHRP. Therefore, any devices that will be forming part of the same network and exchanging information via NHRP, must be configured with the same authentication word. By default no word is configured and so authentication is disabled.

*Syntax:*

```
tnip1 config>NHRP AUTHENTICATION ?
  <1..8 chars>    Authentication word
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP AUTHENTICATION bintec
tnip1 config>
```

### 2.4.2 ENABLE

Enables NHRP in the corresponding TNIP interface. This example enables NHRP in the **tnip1** interface.

*Syntax:*

```
tnip1 config>NHRP ENABLE ?
  <cr>
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP ENABLE
tnip1 config>
```

**Note**

After running **no nhrp enable** in a dynamic configuration, run the **shutdown** and **no shutdown** commands on the TNIP interface. This brings both the TNIP interface and NHRP back up again.

### 2.4.3 GROUP

Enters the NHRP Group Name sent by the spoke to the hub on periodic registration request packets through the new Per-Tunnel QoS for DMVPN feature described in *Per-Tunnel QoS for DMVPN feature* on page 9.

*Syntax:*

```
tnip1 config>NHRP GROUP ?
  <1..40 chars>    Group name
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP GROUP spoke_group1
tnip1 config>
```

**Command history:**

| Release | Modification |
|---|---|
| 10.09.24.20.07 | Command was introduced to perform *Per-Tunnel QoS for DMVPN* feature (on Spoke side). |
| 11.00.04 | Command was introduced to perform *Per-Tunnel QoS for DMVPN* feature (on Spoke side). |
| 11.01.00 | Command was introduced to perform *Per-Tunnel QoS for DMVPN* feature (on Spoke side). |

### 2.4.4 HOLDTIME

Specifies how long the addresses given out by the router (in positive responses to NHRP requests) are valid. Default is 60 minutes. Said time is specified in seconds and must be between 1 and 65535 seconds.

This value also establishes the time period used to send registration requests to an NHS. If said NHS is active, it sends registration requests at a rate of 1/3 of the configured holdtime.

*Syntax:*

```
tnip1 config>NHRP HOLDTIME ?
  <1..65535>    seconds
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP HOLDTIME 300
tnip1 config>
```

### 2.4.5 IGNORE-PURGE-REQ

Configures the router so that NHRP ignores all NHRP Purge request packets received on a specific TNIP interface. They are ignored by responding to them but without taking any further action. The router does not search the NHRP cache for the entry indicated in the received packet. This avoids overloading an NHS when lots of NHRP Purge request packets are sent.

*Syntax:*

```
tnip1 config>NHRP IGNORE-PURGE-REQ
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP IGNORE-PURGE-REQ
tnip1 config>
```

### 2.4.6 LIST

Displays the NHRP configuration.

*Syntax:*

```
tnip1 config>NHRP LIST ?
  <cr>
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP LIST
NHRP protocol: enabled
Multicast NBMA: 10.1.1.2
Destination ip: 1.1.1.1 255.255.255.255
Destination NBMA: 10.1.1.2
NHS: 1.1.1.1
Authentication word: bintec
Group name: spoke_group1
Holding time: 300 seconds
Max packet count: 100 packets
Rate interval: 1 seconds
```

```
Responder interface: internal
Minimum packets for request: 1 packets
Record extensions: ON
Registration no unique: OFF
Server only mode: OFF
Server no caching: OFF
tnip1 config>
```

| | |
|---|---|
| *NHRP protocol:* | Indicates whether NHRP is enabled. |
| *Multicast NBMA:* | Public IP address that the *multicast* packets are sent to. |
| *Destination ip:* | Private IP address for a static tunnel destination. |
| *Destination NBMA:* | Public IP address for a static tunnel destination. |
| *NHS:* | *Next Hop Server* private IP address. |
| *Authentication word:* | Protocol authentication word. |
| *Group name:* | NHRP group name. |
| *Holding time:* | Validity period for the next hop information sent by NHRP. |
| *Max packet count:* | Maximum number of NHRP packets that can be transmitted in a rate interval. |
| *Rate interval:* | Time interval for sending the *max packet count* number of packets. |
| *Responder interface:* | Responder interface for NHRP requests. |
| *Minimum packets for request:* | Minimum number of queries to an NHRP before sending a Resolution Request packet. |
| *Record extensions:* | Indicates if Transit Record Extensions are activated. |
| *Registration no unique:* | Indicates if no-unique register mode is activated. |
| *Server only mode:* | Indicates if server-only mode is activated. |
| *Server no caching:* | Indicates if no caching mode is activated (cache disabled). |

**Command history:**

| Release | Modification |
|---|---|
| 10.09.24.20.07 | As of version 10.09.24.20.07, this command output has changed to show the NHRP Group Name configured on the Spoke. |
| 11.00.04 | As of version 11.00.04, this command output has changed to show the NHRP Group Name configured on the Spoke. |
| 11.01.00 | As of version 11.01.00, this command output has changed to show the NHRP Group Name configured on the Spoke. |

### 2.4.7 MAP

The **MAP** command has two functionalities:

- Creates static entries in the NHRP cache. The correspondence between the public and private addresses of the devices in the network are saved in said entries.
- Configures the addresses that multicast packets use. For example, packets generated by routing protocols such as RIP.

To select either feature, enter **multicast** after a **map** command or, leave it out.

To configure a static entry in a cache: enter a private IP address, a subnet mask and the public IP address (corresponding to said private IP address) after a **map** command. A static entry must be configured on each device with the addresses of the NHS providing the service. This ensures a permanently established tunnel between the spoke and the hub (NHS).

*Syntax:*

```
tnip1 config>NHRP MAP ?
  multicast    Multicast mode
  <a.b.c.d>    Point to point mode, destination ip address
tnip1 config>NHRP MAP MULTICAST ?
  dynamic      Dynamic multicast mode
  <a.b.c.d>    Destination NBMA ip address
tnip1 config>
```

*Example 1:*

```
tnip1 config>NHRP MAP 1.1.1.1 255.255.255.255 10.1.1.2
tnip1 config>
```

There are two instances where *multicast* packets can be configured as destinations. First, a Next Hop Server (NHS) is ready to receive registration requests from other devices. The entries are dynamic and, initially, it's unknown where the multicast packets must be sent. The **dynamic** option is used to ensure the device creates a dynamic entry for each device that registers. Consequently, the multicast packets are sent to all devices registered in the NHS.

*Example 2:*

```
tnip1 config>NHRP MAP MULTICAST DYNAMIC
tnip1 config>
```

The second option is for an NHRP client. Configured in this mode, multicast packets can be sent to the client or to the NHSs (preconfigured) so they can distribute said packets over the rest of the network. The **multicast** option is preceded by the NHS address where the multicast packets are sent.

*Example 3:*

```
tnip1 config>NHRP MAP MULTICAST 10.1.1.2
tnip1 config>
```

You can create multiple entries by using the **map** command and various different addresses.

### 2.4.8 NHS

Configures a private address for the Next Hop that services the device. You can create multiple addresses for different NHSs by executing the command using different IP addresses. The principal NHS is the first one configured and where the NHRP requests are sent (provided this is active). The device registers with all NHSs it has configured.

The router considers an NHS inactive if if doesn't respond to any registration request packets sent or, if the NHS rejects said registration requests for some reason. Maximum time needed to detect an inactive NHS is one third of the holdtime (preconfigured through the **holdtime** command) plus 2 seconds (time needed for the router to realize it hasn't had a response). This means if you configure a holdtime of 300 seconds, the router takes up to 102 seconds to realize the NHS is inactive.

If there is no response to a registration request, registration retries are executed every 2, 4, 8, 16, 32 and 64 seconds. Once the last retry has been executed, the router waits for one third of the holdtime (minus the 64 seconds that have lapsed since the last attempt) to execute another series of retries.

*Syntax:*

```
tnip1 config>NHRP NHS ?
  <a.b.c.d>    Next hop server ip address
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP NHS 1.1.1.2
tnip1 config>
```

### 2.4.9 RATE-LIMIT

Changes the maximum NHRP packet transfer rate by configuring the maximum number of packets and the intervals they are sent in. Maximum transfer **default** rate is 100 packets per second.

*Syntax:*

```
tnip1 config>NHRP RATE-LIMIT ?
  <1..65535>    Max packet count
tnip1 config> NHRP RATE-LIMIT 20 ?
  <1..65535>    Time interval in seconds
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP RATE-LIMIT 20 10
tnip1 config>
```

In the example, the device has been configured at a maximum transfer rate of 20 packets per 10 seconds. The val-

ues, both for the maximum number of packets and the interval, must be between 1 and 65535.

### 2.4.10 RECORD

Enables the sending of Transit Record Extension in NHRP packets for Forward and Reverse. Said extensions provide information on all the NHSs the packet has traveled through (useful for debugging). When enabled, the protocol can detect loops in the network. Default is disabled.

*Syntax:*

```
tnip1 config>NHRP RECORD ?
  <cr>
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP RECORD
tnip1 config>
```

### 2.4.11 REGISTRATION

Disables the Unique bit in the register packets. By default, the devices only register with NHSs, and include a unique entry containing information on the next hop (CIE) in the register packet (unique bit enabled).

*Syntax:*

```
tnip1 config>NHRP REGISTRATION ?
  no-unique    Turns off nhrp unique flag
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP REGISTRATION NO-UNIQUE
tnip1 config>
```

### 2.4.12 RESPONDER

When a device executes an NHRP request and wants to know which device generates the response, a *Responder Extension* must be included in the packet. The responding device fills out the extension with data on the first IP address for the interface. By default, this is the TNIP interface the packet arrives through. To select the interface (for the responder extension) enter **respond.**

*Syntax*

```
tnip1 config>NHRP RESPONDER ?
  <interface>    Interface name
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP RESPONDER serial0/2
tnip1 config>
```

### 2.4.13 SERVER-ONLY

Use the following command to ensure a device only behaves as a server (i.e., does not generate NHRP requests and is limited to responding to requests).

*Syntax:*

```
tnip1 config>NHRP SERVER-ONLY ?
  <cr>
  non-caching    Disable NHRP cache memory
tnip1 config>
```

*Example 1:*

```
tnip1 config>NHRP SERVER-ONLY
tnip1 config>
```

You can also deactivate the cache so it does not retain information by using the **non-caching** option. This option is usually used in a router located between two other routers.

*Example 2:*

```
tnip1 config>NHRP SERVER-ONLY NON-CACHING
tnip1 config>
```

### 2.4.14 TRACK-STATUS

Specifies if the IP static routes activate/deactivate, depending on the state of the NHRP register (not simply on the TNIP interface state). When tracking is active, it checks that the gateway for each static route is registered in the NHRP cache.

- If it's not, the route deactivates.
- If:

  the entry in the NHRP cache is dynamic, then the route activates.

  the entry in the NHRP cache is static:

  - and it matches an NHS, the route activates whenever registration successfully terminates.

  - and it doesn't correspond to an NHS, the route activates.

*Syntax:*

```
tnip1 config>NHRP TRACK-STATUS
tnip1 config>
```

### 2.4.15 USE

This command tells the device to wait for a certain number of packets before executing a request through NHRP to find out and establish the best route for said packets. This avoids creating tunnels in paths containing little traffic where they aren't needed. Said command is followed by the required number of requests to NHRP before the protocol sends a resolution request to its NHS.

*Syntax:*

```
tnip1 config>NHRP USE ?
  <1..65535>    Minimum number of packets to send a NHRP request
tnip1 config>
```

*Example:*

```
tnip1 config>NHRP USE 5
tnip1 config>
```

In this example, no NHRP resolution request is generated until 5 requests are sent to NHRP. The five packets are not lost, but sent over a less efficient path (created thanks to knowing the information on the next hop). The number of the packets must be between 1 and 65535. Default is 1.

# Chapter 3 Monitoring

## 3.1 Monitoring the NHRP protocol

Access the NHRP monitoring menu by entering **protocol NHRP** in the monitoring menu (+).

```
*P 3
+PROTOCOL NHRP
-- NHRP protocol monitor --
nhrp+
```

You can also access from the TNIP interface monitoring menu by entering **NHRP <option>**.

```
*P 3
+NETWORK TNIP1
-- TNIP protocol monitor --
tnip1+NHRP <option>
```

Once you have accessed the NHRP monitoring environment, you can enter the following commands:

| Command | Function |
|---|---|
| *? (HELP)* | Lists the available commands or their options. |
| *CLEAR* | Deletes cache entries or the statistics. |
| *LIST* | Displays monitoring information on the NHRP protocol. |

### 3.1.1 ? (HELP)

Lists the valid commands found at the level where the router is programmed. Said command can also be used after a specific command to list the available options.

*Syntax:*

```
tnip1+NHRP ?
  clear    Permit you to delete the cache entries or the statistics
  list     Display monitoring information on the NHRP protocol
tnip1+
```

### 3.1.2 CLEAR

Enter **clear** to set all statistic counters back to zero and delete cache entries.

*Syntax:*

```
tnip1+NHRP CLEAR ?
  cache         Delete the NHRP protocol cache entries
  statistics    Zeroize the NHRP protocol statistic counters
tnip1+
```

#### 3.1.2.1 CLEAR CACHE

Deletes NHRP cache entries. Statistic entries cannot be deleted as they are entered through the configuration ( **map** command).

There are two ways to delete cache entries: you can delete a specific cached entry or you can clear the entire cache. To delete a specific entry, enter the private IP address identifying the entry and the corresponding mask.

*Example 1:*

```
tnip1+NHRP CLEAR CACHE ALL
tnip1+
```

*Example 2:*

```
tnip1+NHRP CLEAR CACHE ADDRESS 1.1.1.3 255.255.255.255
tnip1+
```

#### 3.1.2.2 CLEAR STATISTICS

Sets NHRP statistic counters back to zero.

*Example:*

```
tnip1+NHRP CLEAR STATISTICS
tnip1+
```

### 3.1.3 LIST

Displays information about the current state of the NHRP protocol, including any relevant statistics.

*Syntax:*

```
tnip1+NHRP LIST ?
  cache          Display the NHRP protocol cache content
  nhs            Display the state of the NHSs configured
  state          Display information on the state of the NHRP protocol
  statistics     Display the NHRP protocol statistics
tnip1+
```

#### 3.1.3.1 LIST CACHE

Displays NHRP cache content. You can display all the cache or, through filters, only relevant cache entries. You can also specify *high detail* to view devices that provided information on said entry through NHRP.

*Syntax:*

```
nhrp+LIST CACHE [<filter>] [<filter>] … [high-detail]
nhrp+LIST CACHE ?
  target          Target IP address resolved by NHRP
      <ip-address>    IP address
          mask           Prefix mask of the target address resolved by NHRP
              <ip-mask>      IP prefix mask
  interface         Tunnel IP interface
      <name>          Interface name
  type            Type of entry in the NHRP cache
      static          NHRP cache entries of type static
      dynamic         NHRP cache entries of type dynamic
      incomplete      NHRP cache entries of type incomplete
  high-detail     Detail level of shown information
  <cr>
```

*Example 1*: Case for an NHS

```
nhrp+ LIST CACHE HIGH-DETAIL
10.1.1.2 255.255.255.255, tnip1 04/08/14, 02:13:37 expire 0h 1m 22s
  Type: dynamic Flags: authoritative unique registered
  NBMA address: 172.10.0.2
  Requester:  10.1.1.3 Request ID: 263
10.1.1.3 255.255.255.255, tnip1 04/08/04, 02:13:37 expire 0h 1m 19s
  Type: dynamic Flags: authoritative unique registered
  NBMA address: 172.10.0.3
  Requester:  10.1.1.2 Request ID: 1775
```

The **first line** displays the entry private address, the mask, the interface to which the cache entry belongs, the date and time the entry was created and the time remaining until the entry *expires* and is considered invalid. If it is not refreshed before this event occurs, the entry is deleted from the cache.

The **second line** shows the type of entry, which can be *dynamic, static* or *incomplete*. The NHRP *Flags* indicate what type of entry information this is and its state.

The **third line** displays the public address or the next hop address.

The **following lines** show the private IP address for the devices that have been given information on the entry and the *request* ID from the NHRP resolution request packet.

*Example 2*: NHC Case

```
nhrp+ list cache target 10.1.1.3
```

```
10.1.1.3 255.255.255.255, tnip1 04/08/14, 02:13:37 expire 0h 1m 2s
  Type: dynamic Flags: authoritative unique registered
  NBMA address: 172.10.0.3
```

*Example 3*: NHC Case

```
nhrp+ LIST CACHE INTERFACE TNIP1 TYPE STATIC
10.1.1.0 255.255.255.0 via  10.1.1.1, tnip1 04/08/14, 14:50:26 never expire
  Type: static Flags: authoritative used
  NBMA address: 172.10.0.1
```

#### 3.1.3.2 LIST NHS

Displays the state of the NHSs configured for the device and the interface. This also displays the time remaining before a new registration request NHRP packet is sent to the NHS. Each NHS is known by its private address and its status can be *alive, not alive, alive and expecting reg reply, not alive and expecting reg replay.* An NHS is considered *not alive* if it does not respond to a registration request. When a positive response is received to a request, the NHS is considered *alive.*

*Example 1*:

```
tnip1+NHRP LIST NHS
 Interface tnip1:
  1.1.1.1 alive (next reg req in 0h 1m 27s)
tnip1+
```

*Example 2*:

```
tnip1+NHRP LIST NHS
 Interface tnip1:
  1.1.1.1 not alive and expecting reg reply (next reg req in 0h 0m 6s)
tnip1+
```

#### 3.1.3.3 LIST STATE

Displays information on the state of NHRP for the interface. This indicates if it is enabled and if the protocol is up.

*Example:*

```
tnip1+NHRP LIST STATE
    Interface tnip1 is UP and ENABLED
tnip1+
```

#### 3.1.3.4 LIST STATISTICS

Displays NHRP statistics for the current interface.Indicates the number of packets received and sent for each type of NHRP packet and the total number of packets sent and received.

*Example:*

```
tnip1+NHRP LIST STATISTICS
Interface tnip1:
  Sent:
    Resolution Request: 21
    Resolution Reply: 0
    Register Request: 1745
    Register Reply: 0
    Purge Request: 0
    Purge Reply: 0
    Error: 0
      Total sent: 1766
  Received:
    Resolution Request: 0
    Resolution Reply: 21
    Register Request: 0
    Register Reply: 1736
    Purge Request: 0
    Purge Reply: 0
    Error: 0
      Total received: 1757
```

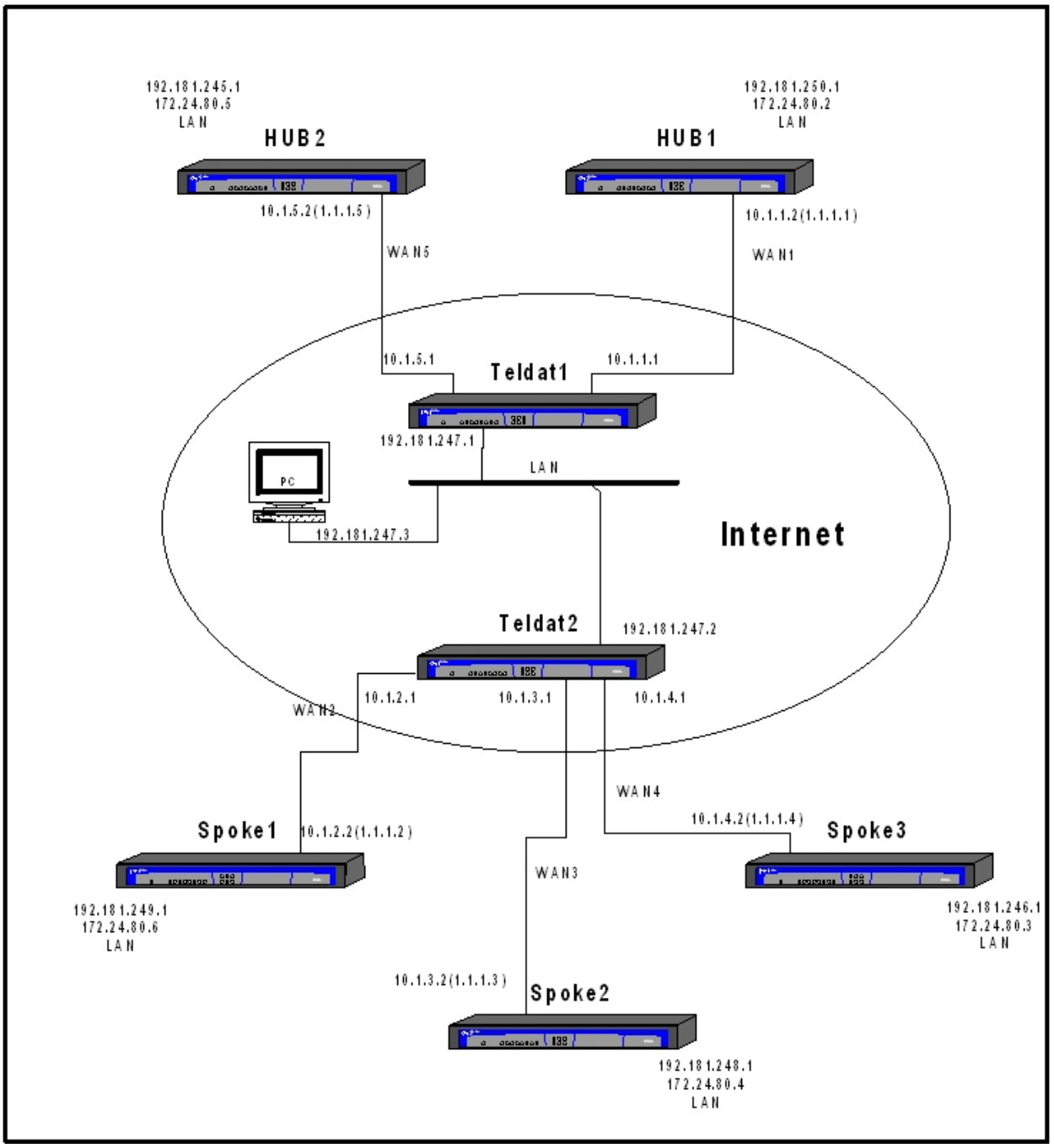
```
tnip1+
```

# Chapter 4 Examples

## 4.1 DMVPN Configuration Example

DMVPN Configuration example using NHRP:

### 4.1.1 Scenario




In the above scenario, you can easily control the path the packets are routed over. Moreover, you are able to monitor said packets without carrying out changes in the model structure.

We have created a simulated public network connecting two devices through a LAN, where WANs are the end connections to which the *Spokes* (or remote devices) and *Hubs* (or servers) are connected. This way, if we disconnect cables, switch off devices and disable interfaces, we can simulate incidences that can occur in a real network.

In the LAN connecting these two devices, we have connected a laptop with software that is capable of analyzing the network traffic. Thus, we can register the information exchanged in all the communications.

Using DMVPN, we can securely interconnect the remote devices in a public network without having to add an entry each time a new spoke is introduced (thanks to IPSec-protected mGre tunnels, **where the remoteend is not con-**

**figured** and allows a simple, dynamic configuration).

To establish the mGre tunnel + IPSec (allowing connections between spokes), you need the remote spoke public address. The tool used to learn said address is NHRP (Next Hop Resolution Protocol).

As said, the spokes configuration is very simple. Almost all the information is obtained dynamically through NHRP and routing protocols. We basically configure:

- A multigre tunnel (mGre) without destination address.
- The address assigned to the interface in all the spokes must belong to the same network.
- The server addresses (NHS) and the static mapping in NHRP.
- An IPSec tunnel without destination address.
- RIP in the interface.

This makes cloning spokes extremely simple. As you will see further on, HUB configuration is also simple, the difference lies in that these devices need to support a large quantity of open tunnels.

### 4.1.2 Functionality

When Spoke2 needs to establish a link with Spoke1, with its loopback address (192.181.248.1) as source and Spoke1's loopback address (192.181.249.1) as destination, the following procedure is executed (always assuming that HUB1 and HUB2 are operating correctly):

- On startup, the Spokes send Registration Requests to the HUBs.

  Information on the 10.1.X.1 1.1.1.X relationship is sent in said requests.

  They are sent periodically.
- The HUBs respond with Registration Replies.

  The spokes can identify the live HUBs and select which one will receive their Resolution Requests.
- At this point, the HUBs know the 10.1.X.1 1.1.1.X relationship for all spokes. In turn, the Spokes are aware of the same relationship for the HUBs 10.1.5.2 1.1.1.5 and 10.1.1.2 1.1.1.1. This means there is a multigre tunnel between each HUB and every Spoke.
- The information registered in the HUB 1 NHRP cache is similar to that shown below:

```
1.1.1.2 255.255.255.255, tnip1 06/28/05, 10:47:49 expire 0h 4m 55s
  Type: dynamic Flags: authoritative unique registered used
  NBMA address: 10.1.2.2
1.1.1.3 255.255.255.255, tnip1 06/28/05, 10:47:24 expire 0h 4m 30s
  Type: dynamic Flags: authoritative unique registered
  NBMA address: 10.1.3.2
1.1.1.4 255.255.255.255, tnip1 06/28/05, 10:47:51 expire 0h 4m 57s
  Type: dynamic Flags: authoritative unique registered used
  NBMA address: 10.1.4.2
1.1.1.5 255.255.255.255, tnip1 06/28/05, 10:46:15 expire 0h 8m 22s
  Type: dynamic Flags: authoritative unique registered
  NBMA address: 10.1.5.2
```

And for Spoke2:

```
1.1.1.1 255.255.255.255, tnip1 06/28/05, 10:02:43 never expire
  Type: static Flags: authoritative used
  NBMA address: 10.1.1.2
1.1.1.5 255.255.255.255, tnip1 06/28/05, 10:02:43 never expire
  Type: static Flags: authoritative used
  NBMA address: 10.1.5.2
```

- The Spokes send their routing traffic through this tunnel.
- The HUBs send the entire learned routing table to all their Spokes, **without changing the next-hop**. That is, they *reflect* the routes back out the same interface they were learned from. Consequently, these features must be enabled in the protocols where they are deactivated by default. For example,

  RIP:

  - Default is next-hop is not changed when reflecting the route.

  - *Split-horizon* must be deactivated.

EIGRP:

- Default is next-hop is changed when reflecting the route.

- *Split-horizon* must be deactivated.

OSPF:

- These aspects are irrelevant for OSPF; however, make sure *broadcast* is active. Also, HUB priority must be greater than 0 and Spoke priority must be 0.

- At this point, Spoke2 knows the next-hop in network 192.181.249.0 is 1.1.1.2, having learned this from the routing (dynamic or static).

HUB1 routing information is as follows:

```
·      Type                 Dest net/Mask   Cost Age  Next hop(s)
·
·      Stat(1)[0]            0.0.0.0/0    [ 60/1 ] 0   serial0/1
·      Sbnt(0)[0]            1.0.0.0/8    [240/1 ] 0   None
·       Dir(0)[1]            1.1.1.0/24   [  0/1 ] 0   tnip1
·      Sbnt(0)[0]           10.0.0.0/8    [240/1 ] 0   None
·       Dir(0)[1]           10.1.1.0/30   [  0/1 ] 0   serial0/1
·       Dir(0)[1]         172.24.0.0/16   [  0/1 ] 0   ethernet0/0
·      Sbnt(0)[0]      192.181.245.0/24   [240/1 ] 0   None
·       RIP(0)[0]      192.181.245.1/32   [100/2 ] 20  1.1.1.5 (tnip1)
·      Sbnt(0)[0]      192.181.246.0/24   [240/1 ] 0   None
·       RIP(0)[0]      192.181.246.1/32   [100/2 ] 0   1.1.1.4 (tnip1)
·      Sbnt(0)[0]      192.181.248.0/24   [240/1 ] 0   None
·       RIP(0)[0]      192.181.248.1/32   [100/2 ] 10  1.1.1.3 (tnip1)
·      Sbnt(0)[0]      192.181.249.0/24   [240/1 ] 0   None
·       RIP(0)[0]      192.181.249.1/32   [100/2 ] 20  1.1.1.2 (tnip1)
Sbnt(0)[0]      192.181.250.0/24   [240/1 ] 0   None
        Dir(0)[1]      192.181.250.1/32   [  0/1 ] 0   loopback1
```

Spoke2 routing information is as follows:

```
Stat(1)[0]             0.0.0.0/0    [ 60/1 ] 0   serial0/0
Sbnt(0)[0]             1.0.0.0/8    [240/1 ] 0   None
 Dir(0)[1]             1.1.1.0/24   [  0/1 ] 0   tnip1
Sbnt(0)[0]            10.0.0.0/8    [240/1 ] 0   None
 Dir(0)[1]            10.1.3.0/30   [  0/1 ] 0   serial0/0
 Dir(0)[1]          172.24.0.0/16   [  0/1 ] 0   ethernet0/0
Sbnt(0)[0]       192.181.245.0/24   [240/1 ] 0   None
 RIP(0)[0]       192.181.245.1/32   [100/3 ] 10  1.1.1.5 (tnip1)
Sbnt(0)[0]       192.181.246.0/24   [240/1 ] 0   None
 RIP(0)[0]       192.181.246.1/32   [100/3 ] 10  1.1.1.4 (tnip1)
Sbnt(0)[0]       192.181.249.0/24   [240/1 ] 0   None
 RIP(0)[0]       192.181.249.1/32   [100/3 ] 10  1.1.1.2 (tnip1)
Sbnt(0)[0]       192.181.250.0/24   [240/1 ] 0   None
 RIP(0)[0]       192.181.250.1/32   [100/2 ] 10  1.1.1.1 (tnip1)
Sbnt(0)[0]       192.181.248.0/24   [240/1 ] 0   None
 Dir(0)[1]       192.181.248.1/32   [  0/1 ] 0   loopback1
```

- Now we need to know the IP address associated with 1.1.1.2. Thus, a Resolution Request is sent to HUB1.
- HUB1 searches its NHRP tables and responds (*Resolution Reply*) with address 10.1.2.2.
- The information registered in Spoke2's NHRP cache is:

```
·      1.1.1.2 255.255.255.255, tnip1 06/28/05, 10:59:31 expire 0h 3m 40s
·        Type: dynamic Flags: router used
·        NBMA address: 10.1.2.2
·
·      1.1.1.1 255.255.255.255, tnip1 06/28/05, 10:02:43 never expire
·        Type: static Flags: authoritative used
·        NBMA address: 10.1.1.2
·
·      1.1.1.5 255.255.255.255, tnip1 06/28/05, 10:02:43 never expire
·        Type: static Flags: authoritative used
·        NBMA address: 10.1.5.2
```

- Spoke1 opens a GRE tunnel with Spoke2 to route all its traffic intended for network 192.181.249.0.

We can add encryption protection to this entire scenario through IPSec in transport mode, configured so all information in GRE encapsulation is protected.

### 4.1.3 HUB1 Configuration

```
; Showing System Configuration ...
; XXX Router 2 8 Version 10.6.0-Alfa
log-command-errors
no configuration
set hostname hub1
set inactivity-timer disabled
add device tnip 1
add device loopback 1
set data-link frame-relay serial0/0
set data-link frame-relay serial0/1
set data-link frame-relay serial0/2
feature access-lists
                ; -- Access Lists user configuration -
                ; This is the access control list to be associated to IPSec.
                ; All the HUB's outgoing traffic with source 10.1.1.2 encapsulated in GRE is
                ; encrypted.
                ; The NHRP negotiation packets also go over GRE, consequently this negotiation
                ; is also encrypted.
                ; No unknown destination is configured as we are dealing with an mGre tunnel.
                ; Later on advanced pkt-dest-isakmp-dest is configured in IPSec that implies
                ; that the access control list destination coincides with the packet
                ; destination, which in turn is the remote end for the IPSec negotiation
                ; and SAs. The packet destination is the GRE tunnel's remote end.
   access-list 100
;
      entry 1 default
      entry 1 permit
      entry 1 source address 10.1.1.2 255.255.255.255
      entry 1 protocol gre
;
   exit
;
   access-list 1
;
      entry 1 default
      entry 1 deny
      entry 1 source address 10.1.1.0 255.255.255.252
;
      entry 2 default
      entry 2 permit
;
   exit
;
exit
;
network ethernet0/0
; -- Ethernet Interface User Configuration --
   ip address 172.24.80.2 255.255.0.0
;
exit
;
;
network serial0/1
; -- Frame Relay user configuration --
   pvc 16 default
;
   point-to-point-line 16
   set line-speed 2048000
   no lmi
;
   ip address 10.1.1.2 255.255.255.252
;
exit
;
```

```
network loopback1
; -- Loopback interface configuration --
   ip address 192.181.250.1 255.255.255.255
;
exit
;
;
network tnip1
; -- IP Tunnel Net Configuration --
   enable
   ip address 1.1.1.1 255.255.255.0
                ; mGRE is enabled
   mode gre multipoint
                ; Because this is an mGRE, the destination address is not configured.
   source 10.1.1.2
   nhrp enable
   nhrp holdtime 600
   nhrp map multicast dynamic
                ; Record is configured for debugging purposes, however this is not essential for
                ; the proper operation of NHRP.

   nhrp record
   nhrp responder tnip1
   encapsulation
; -- GRE Configuration --
      key 123456
   exit
;
exit
;
event
; -- ELS Config --
   enable trace subsystem NHRP ALL
   enable filter
   filter 1 default
   filter 1 text "Datagram Rcvd"
   filter 1 action red
   filter 2 default
   filter 2 text "Datagram Sent"
   filter 2 action blue
exit
;
;
protocol ip
; -- Internet protocol user configuration --
;
   route 0.0.0.0 0.0.0.0 serial0/1
;
   classless
;
;
   ipsec
; -- IPSec user configuration -
                ; Note that pkt-dest-isakmp-dest is configured as previously commented.
                ; Due to this, destination addresses are not configured in the templates.
      assign-access-list 100
;
      template 1 isakmp des sha1
      template 1 life duration seconds 1d
      template 1 ike natt-version rfc
      template 1 keepalive dpd
;
      template 2 dynamic esp des md5
      template 2 source-address serial0/1
      template 2 encap transport
;
      map-template 100 2
```

```
      key preshared ip 0.0.0.0 ciphered 0x0B5F13472B61C799
      advanced pkt-dest-isakmp-dest
      advanced dpd no always-send
   exit
;
exit
;
protocol rip
; -- RIP protocol user configuration --
   enable
   compatibility 172.24.80.2 send none
   compatibility 172.24.80.2 receive none
;
   compatibility 10.1.1.2 send none
   compatibility 10.1.1.2 receive none
;
                ; The HUB can send and receive RIP over the GRE interface
   compatibility 1.1.1.1 send rip2-multicast
   compatibility 1.1.1.1 receive rip2
;
   compatibility 192.181.250.1 send none
   compatibility 192.181.250.1 receive none
;
   distribute-list out 1
                ; The HUB must reflect the GRE interface routes.

   sending 1.1.1.1 no split-horizon
;
exit
;
dump-command-errors
end
; --- end ---
```

### 4.1.4 SPOKE2 Configuration

```
; Showing System Configuration ...
; XXX Router 2 8 Version 10.6.0-Alfa

log-command-errors
no configuration
set hostname spoke2
set inactivity-timer disabled
add device tnip 1
add device loopback 1
set data-link frame-relay serial0/0
set data-link frame-relay serial0/1
set data-link frame-relay serial0/2
feature access-lists
; -- Access Lists user configuration --
   access-list 100
                ; This is the access control list to be associated to IPSec.
                ; All the HUB's outgoing traffic with source 10.1.3.2 from Spoke2 encapsulated in
                ; GRE is encrypted.
                ; The NHRP negotiation packets also go over GRE, consequently this negotiation
                ; is also encrypted.
                ; No unknown destination is configured as we are dealing with an mGre tunnel.
                ; Later on advanced pkt-dest-isakmp-dest is configured in IPSec that implies
                ; that the access control list destination coincides with the packet
                ; destination, which in turn is the remote end for the IPSec negotiation
                ; and SAs. The packet destination is the GRE tunnel's remote end.
;
      entry 1 default
      entry 1 permit
      entry 1 source address 10.1.3.2 255.255.255.255
      entry 1 protocol gre
;
   exit
;
   access-list 1
;
      entry 1 default
      entry 1 deny
      entry 1 source address 10.1.3.0 255.255.255.252
;
      entry 2 default
      entry 2 permit
;
   exit
;
exit
;
network ethernet0/0
; -- Ethernet Interface User Configuration --
   ip address 172.24.80.4 255.255.0.0
;
exit
;
;
network serial0/0
; -- Frame Relay user configuration --
   pvc 16 default
;
   point-to-point-line 16
   set line-speed 2048000
   no lmi
;
   ip address 10.1.3.2 255.255.255.252
;
exit
```

```
;
network loopback1
; -- Loopback interface configuration --
   ip address 192.181.248.1 255.255.255.255
;
exit
;
;
network tnip1
; -- IP Tunnel Net Configuration --
   enable
   ip address 1.1.1.3 255.255.255.0
                ; mGRE is enabled
   mode gre multipoint
                ; Because this is an mGRE, the destination address is not configured.
   source 10.1.3.2
   nhrp enable
   nhrp holdtime 300
   nhrp map multicast 10.1.1.2
   nhrp map multicast 10.1.5.2
   nhrp map 1.1.1.1 255.255.255.255 10.1.1.2
   nhrp map 1.1.1.5 255.255.255.255 10.1.5.2
   nhrp nhs 1.1.1.1
   nhrp nhs 1.1.1.5
   nhrp record
   encapsulation
; -- GRE Configuration --
      key 123456
   exit
;
exit
;
event
; -- ELS Config --
   enable trace subsystem NHRP ALL
   enable filter
   filter 1 default
   filter 1 text "Datagram Rcvd"
   filter 1 action red
   filter 2 default
   filter 2 text "Datagram Sent"
   filter 2 action blue
exit
;
;
protocol ip
; -- Internet protocol user configuration --
;
   route 0.0.0.0 0.0.0.0 serial0/0
;
   classless
;
;
   ipsec
; -- IPSec user configuration --
                ; Note that pkt-dest-isakmp-dest is configured as previously commented.
                ; Due to this, destination addresses are not configured in the templates.
      assign-access-list 100
;
      template 1 isakmp des sha1
      template 1 life duration seconds 1d
      template 1 ike natt-version rfc
      template 1 keepalive dpd
;
      template 2 dynamic esp des md5
      template 2 source-address serial0/0
      template 2 encap transport
```

```
;
      map-template 100 2
      key preshared ip 0.0.0.0 ciphered 0x0B5F13472B61C799
      advanced pkt-dest-isakmp-dest
      advanced dpd no always-send
   exit
;
exit
;
protocol rip
; -- RIP protocol user configuration --
   enable
   compatibility 172.24.80.4 send none
   compatibility 172.24.80.4 receive none
;
   compatibility 10.1.3.2 send none
   compatibility 10.1.3.2 receive none
;
; El Spoke puede enviar y recibir RIP por el interfaz GRE
   compatibility 1.1.1.3 send rip2-multicast
   compatibility 1.1.1.3 receive rip2
;
   compatibility 192.181.248.1 send none
   compatibility 192.181.248.1 receive none
;
   distribute-list out 1
exit
;
dump-command-errors
end
; --- end ---
```

### 4.1.5 Device1 Configuration

```
; Showing System Configuration ...
; XXX Router 2 8 Version 10.6.0-Alfa
log-command-errors
no configuration
set data-link frame-relay serial0/0
set data-link frame-relay serial0/1
set data-link x25 serial0/2
set hostname device1
feature access-lists
; -- Access Lists user configuration --
   access-list 101
;
      entry 1 default
      entry 1 deny
      entry 1 source address 192.181.250.0 255.255.255.0
;
      entry 2 default
      entry 2 permit
;
   exit
;
   access-list 102
;
      entry 1 default
      entry 1 deny
      entry 1 source address 192.181.245.0 255.255.255.0
;
      entry 2 default
      entry 2 permit
;
   exit
;
exit
;
network ethernet0/0
; -- Ethernet Interface User Configuration --
   ip address 192.181.247.1 255.255.255.0
;
exit
;
;
network serial0/0
; -- Frame Relay user configuration --
   pvc 16 default
;
   point-to-point-line 16
   set line-speed 2048000
   no lmi
;
   ip address 10.1.1.1 255.255.255.252
;
   ip access-group 101 in
;
exit
;
network serial0/1
; -- Frame Relay user configuration --
   pvc 16 default
;
   point-to-point-line 16
   set line-speed 2048000
   no lmi
;
   ip address 10.1.5.1 255.255.255.252
```

```
;
   ip access-group 102 in
;
exit
;
;
protocol ip
; -- Internet protocol user configuration --
;
;
   route 10.1.2.0 255.255.255.252 192.181.247.2
   route 10.1.3.0 255.255.255.252 192.181.247.2
   route 10.1.4.0 255.255.255.252 192.181.247.2
   route 172.24.0.0 255.255.0.0 192.181.247.2
;
;
   classless
;
;
exit
;
dump-command-errors
end
; --- end ---
```

### 4.1.6 Device2 Configuration

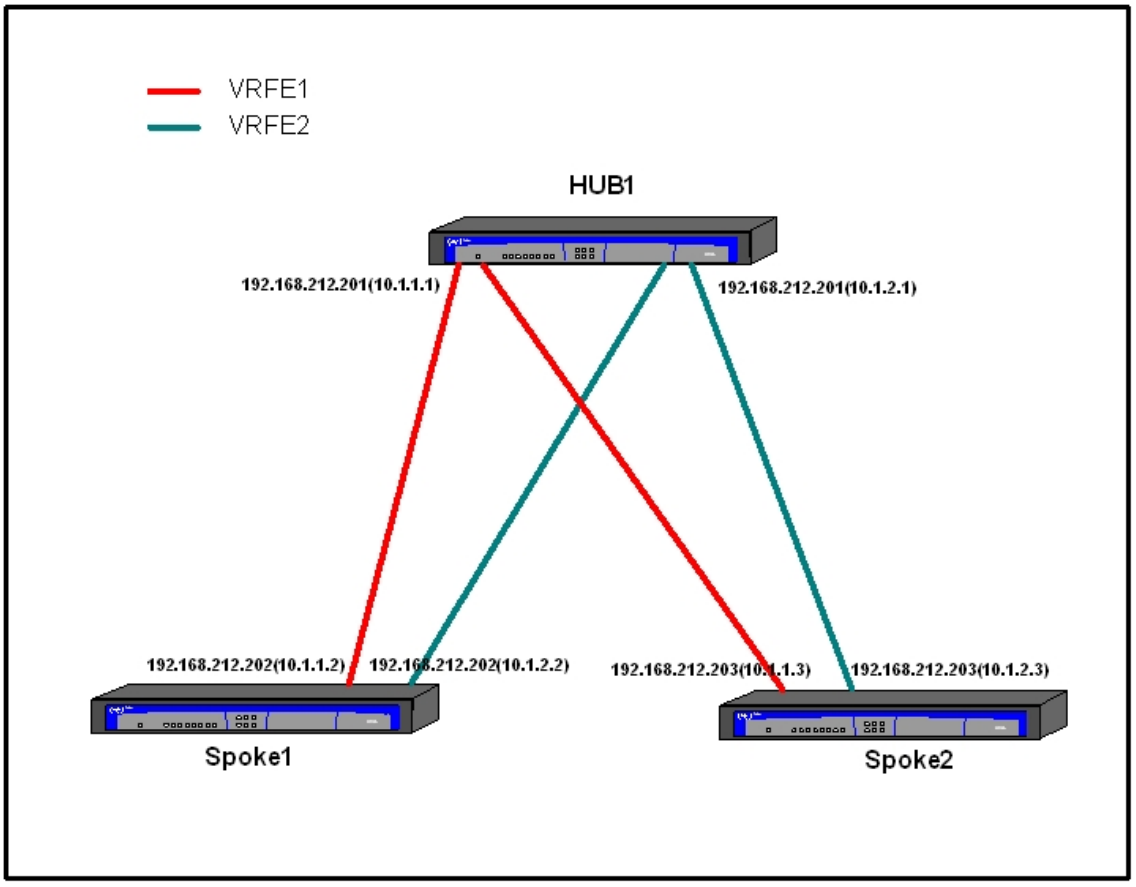
```
; Showing System Configuration ...
; CENTRIX SEC (a) Router 2 11 Version 10.5.8-Alfa
log-command-errors
no configuration
set data-link frame-relay serial0/0
set data-link frame-relay serial0/1
set data-link frame-relay serial0/2
set hostname device2
feature access-lists
; -- Access Lists user configuration --
   access-list 102
;
      entry 1 default
      entry 1 deny
      entry 1 source address 192.181.249.0 255.255.255.0
;
      entry 2 default
      entry 2 permit
;
   exit
;
   access-list 103
;
      entry 1 default
      entry 1 deny
      entry 1 source address 192.181.248.0 255.255.255.0
;
      entry 2 default
      entry 2 permit
;
   exit
;
   access-list 101
;
      entry 1 default
      entry 1 deny
      entry 1 source address 192.181.246.0 255.255.255.0
;
      entry 2 default
      entry 2 permit
;
   exit
;
exit
;
network ethernet0/0
; -- Ethernet Interface User Configuration --
   ip address 192.181.247.2 255.255.255.0
;
exit
;
;
network serial0/0
; -- Frame Relay user configuration --
   pvc 16 default
;
   point-to-point-line 16
   set line-speed 2048000
   no lmi
;
   ip address 10.1.4.1 255.255.255.252
;
   ip access-group 101 in
;
```

```
exit
;
network serial0/1
; -- Frame Relay user configuration --
   pvc 16 default
;
   point-to-point-line 16
   set line-speed 2048000
   no lmi
;
   ip address 10.1.2.1 255.255.255.252
;
   ip access-group 102 in
;
exit
;
network serial0/2
; -- Frame Relay user configuration --
   pvc 16 default
;
   point-to-point-line 16
   set line-speed 2048000
   no lmi
;
   ip address 10.1.3.1 255.255.255.252
;
   ip access-group 103 in
;
exit
;
;
protocol ip
; -- Internet protocol user configuration --
;
;
   route 10.1.1.0 255.255.255.252 192.181.247.1
   route 172.24.0.0 255.255.0.0 10.1.4.2
   route 10.1.5.0 255.255.255.252 192.181.247.1
;
;
   classless
;
;
exit
;
dump-command-errors
end
; --- end ---
```

## 4.2 DMVPN with multi VRF: Example

This scenario is made up of three routers where different VRFs have been defined. One router acts as the HUB and the other two as spokes. Each spoke opens two separate GRE tunnels to the HUB, each one in a different VRF. There are two external and two internal VRFs that allow the HUB device and the remote spoke devices to use the same addressing in the local networks (simulated by loopback interfaces) and the same IP address in the external subinterfaces of all the devices. This way, addressing in networks with multiple spoke devices can be simplified.

In the example, traffic routed through a VRF (VRFE1) is coded and the traffic routed through the other VRF (VRFE2) goes in clear.


VRFE1
VRFE2
HUB1
192.168.212.201(10.1.1.1)
192.168.212.201(10.1.2.1)
192.168.212.202(10.1.1.2)
192.168.212.202(10.1.2.2)
192.168.212.203(10.1.1.3)
192.168.212.203(10.1.2.3)
Spoke1
Spoke2

### 4.2.1 Hub Configuration

```
NHRP-VRF-HUB Config$sho con
; Showing Menu and Submenus Configuration for access-level 15 ...
; XXX Router 2 8 Version 10.6.0-Alfa
   log-command-errors
   no configuration
   set hostname NHRP-VRF-HUB
   set inactivity-timer disabled
   feature vrf
; -- VRF user configuration --
      vrf VRFI1
      vrf VRFI2
      vrf VRFE1
      vrf VRFE2
   exit
;
   add device tnip 1
   add device tnip 2
   add device eth-subinterface ethernet0/0 10
   add device eth-subinterface ethernet0/0 20
   add device loopback 1
   add device loopback 2
   set data-link x25 serial0/1
   feature access-lists
; -- Access Lists user configuration --
      access-list 100
         entry 1 default
         entry 1 permit
         entry 1 source address 192.168.212.201 255.255.255.255
;
      exit
;
   exit
;
;
;
;
;
;
;
;
;
   network tnip1
; -- IP Tunnel Net Configuration --
      ip vrf forwarding VRFI1
;
      ip address 10.1.1.1 255.255.255.0
;
;
;
;
;
      enable
      mode gre multipoint
      source ethernet0/0.10
      vrf-encap VRFE1
      no ip icmp redirects
      nhrp enable
      nhrp holdtime 90
      nhrp map multicast dynamic
      nhrp track-status
      encapsulation
; -- GRE Configuration --
         key 2010
      exit
```

```
;
   exit
;
   network tnip2
; -- IP Tunnel Net Configuration --
      ip vrf forwarding VRFI2
;
      ip address 10.1.2.1 255.255.255.0
;
;
;
;
;
      enable
      mode gre multipoint
      source ethernet0/0.20
      vrf-encap VRFE2
      no ip icmp redirects
      nhrp enable
      nhrp holdtime 90
      nhrp map multicast dynamic
      nhrp track-status
      encapsulation
; -- GRE Configuration --
         key 2012
      exit
;
   exit
;
   network loopback1
; -- Loopback interface configuration --
      ip vrf forwarding VRFI1
;
      ip address 192.168.1.1 255.255.255.0
;
;
;
;
;
   exit
;
   network loopback2
; -- Loopback interface configuration --
      ip vrf forwarding VRFI2
;
      ip address 192.168.1.1 255.255.255.0
;
;
;
;
;
   exit
;
   network ethernet0/0.10
; -- Ethernet Subinterface Configuration --
      ip vrf forwarding VRFE1
;
      ip address 192.168.212.201 255.255.254.0
;
;
;
;
;
      encapsulation dot1q 10
;
;
;
```

```
;
   exit
;
   network ethernet0/0.20
; -- Ethernet Subinterface Configuration --
      ip vrf forwarding VRFE2
;
      ip address 192.168.212.201 255.255.254.0
;
;
;
;
;
      encapsulation dot1q 20
;
;
;
;
   exit
;
   event
; -- ELS Config --
      enable trace subsystem TNIP ALL
      enable trace subsystem IKE ALL
      enable trace event IPSEC.031
   exit
;
;
   protocol ip
; -- Internet protocol user configuration --
      ipsec
; -- IPSec user configuration --
         enable
         assign-access-list 100
;
         template 1 isakmp des md5
         template 1 life duration seconds 1d
         template 1 ike natt-version rfc
         template 1 keepalive dpd
;
         template 2 dynamic esp tdes sha1
         template 2 source-address 192.168.212.201
         template 2 encap transport
         template 2 vrf VRFE1
;
         map-template 100 2
         key preshared ip 0.0.0.0 ciphered 0xD0A99FF6F5D7FA37
         advanced renegotiation-time 100
         advanced pkt-dest-isakmp-dest
         advanced dpd no always-send
      exit
;
;
      vrf VRFI1
         route 192.168.2.1 255.255.255.255 10.1.1.2
         route 192.168.3.1 255.255.255.255 10.1.1.3
;
;
      exit
;
;
      vrf VRFI2
         route 192.168.2.1 255.255.255.255 10.1.2.2
         route 192.168.3.1 255.255.255.255 10.1.2.3
;
;
      exit
```

```
;
;
;
   exit
;
;
;
   dump-command-errors
   end
NHRP-VRF-HUB Config$
```

### 4.2.2 Spoke1 Configuration

```
NHRP-VRF-SPOKE-1 Config$sho con
; Showing Menu and Submenus Configuration for access-level 15 ...
; Super Router * * Version 10.8.13-Alfa-NV
   log-command-errors
   no configuration
   set hostname NHRP-VRF-SPOKE-1
   set inactivity-timer disabled
   feature vrf
; -- VRF user configuration --
      vrf VRFI1
      vrf VRFI2
      vrf VRFE1
      vrf VRFE2
   exit
;
   add device tnip 1
   add device tnip 2
   add device eth-subinterface ethernet0/0 10
   add device eth-subinterface ethernet0/0 20
   add device loopback 1
   add device loopback 2
   set data-link x25 serial0/1
   set data-link x25 serial0/2
   feature access-lists
; -- Access Lists user configuration --
      access-list 100
         entry 1 default
         entry 1 permit
         entry 1 source address 192.168.212.202 255.255.255.255
;
      exit
;
   exit
;
;
;
;
;
;
   network tnip1
; -- IP Tunnel Net Configuration --
      ip vrf forwarding VRFI1
;
      ip address 10.1.1.2 255.255.255.0
;
;
;
;
;
      enable
      mode gre multipoint
      source ethernet0/0.10
      vrf-encap VRFE1
      no ip icmp redirects
      nhrp enable
      nhrp holdtime 90
      nhrp map multicast 10.1.1.1
      nhrp map 10.1.1.1 255.255.255.0 192.168.212.201
      nhrp nhs 10.1.1.1
      encapsulation
; -- GRE Configuration --
         key 2010
      exit
;
```

```
   exit
;
   network tnip2
; -- IP Tunnel Net Configuration --
      ip vrf forwarding VRFI2
;
      ip address 10.1.2.2 255.255.255.0
;
;
;
;
;
      enable
      mode gre multipoint
      source ethernet0/0.20
      vrf-encap VRFE2
      no ip icmp redirects
      nhrp enable
      nhrp holdtime 90
      nhrp map multicast 10.1.2.1
      nhrp map 10.1.2.1 255.255.255.0 192.168.212.201
      nhrp nhs 10.1.2.1
      encapsulation
; -- GRE Configuration --
         key 2012
      exit
;
   exit
;
   network loopback1
; -- Loopback interface configuration --
      ip vrf forwarding VRFI1
;
      ip address 192.168.2.1 255.255.255.0
;
;
;
;
;
   exit
;
   network loopback2
; -- Loopback interface configuration --
      ip vrf forwarding VRFI2
;
      ip address 192.168.2.1 255.255.255.0
;
;
;
;
;
   exit
;
   network ethernet0/0.10
; -- Ethernet Subinterface Configuration --
      ip vrf forwarding VRFE1
;
      ip address 192.168.212.202 255.255.254.0
;
;
;
;
;
      encapsulation dot1q 10
;
;
;
```

```
;
   exit
;
   network ethernet0/0.20
; -- Ethernet Subinterface Configuration --
      ip vrf forwarding VRFE2
;
      ip address 192.168.212.202 255.255.254.0
;
;
;
;
;
      encapsulation dot1q 20
;
;
;
;
   exit
;
   event
; -- ELS Config --
      enable trace subsystem TNIP ALL
      enable trace subsystem IKE ALL
      enable trace event IPSEC.031
   exit
;
;
   protocol ip
; -- Internet protocol user configuration --
      ipsec
; -- IPSec user configuration --
         enable
         assign-access-list 100
;
         template 1 isakmp des md5
         template 1 life duration seconds 1d
         template 1 ike natt-version rfc
         template 1 keepalive dpd
;
         template 2 dynamic esp tdes sha1
         template 2 source-address 192.168.212.202
         template 2 encap transport
         template 2 vrf VRFE1
;
         map-template 100 2
         key preshared ip 0.0.0.0 ciphered 0xD0A99FF6F5D7FA37
         advanced renegotiation-time 100
         advanced pkt-dest-isakmp-dest
         advanced dpd no always-send
      exit
;
;
      vrf VRFI1
         route 192.168.1.0 255.255.255.0 10.1.1.1
         route 192.168.3.0 255.255.255.0 10.1.1.1
;
;
      exit
;
;
      vrf VRFI2
         route 192.168.1.0 255.255.255.0 10.1.2.1
         route 192.168.3.0 255.255.255.0 10.1.2.1
;
;
      exit
```

```
;
;
;
   exit
;
;
;
   dump-command-errors
   end
NHRP-VRF-SPOKE-1 Config$
```

### 4.2.3 Spoke2 Configuration

```
NHRP-VRF-SPOKE-2 Config$sho con
; Showing Menu and Submenus Configuration for access-level 15 ...
; Super Router * * Version 10.8.13-Alfa
   log-command-errors
   no configuration
   set hostname NHRP-VRF-SPOKE-2
   set inactivity-timer disabled
   feature vrf
; -- VRF user configuration --
      vrf VRFI1
      vrf VRFI2
      vrf VRFE1
      vrf VRFE2
   exit
;
   add device tnip 1
   add device tnip 2
   add device eth-subinterface ethernet0/0 10
   add device eth-subinterface ethernet0/0 20
   add device loopback 1
   add device loopback 2
   set data-link x25 serial0/1
   feature access-lists
; -- Access Lists user configuration --
      access-list 100
         entry 1 default
         entry 1 permit
         entry 1 source address 192.168.212.203 255.255.255.255
;
      exit
;
   exit
;
;
;
;
;
;
;
;
;
;
   network voip2/0
; -- VoIP interface Configuration --
      line 1 interface-type fxo
;
      line 2 interface-type fxo
;
      line 3 interface-type fxo
;
      line 4 interface-type fxo
;
   exit
;
   network tnip1
; -- IP Tunnel Net Configuration --
      ip vrf forwarding VRFI1
;
      ip address 10.1.1.3 255.255.255.0
;
;
;
;
;
```

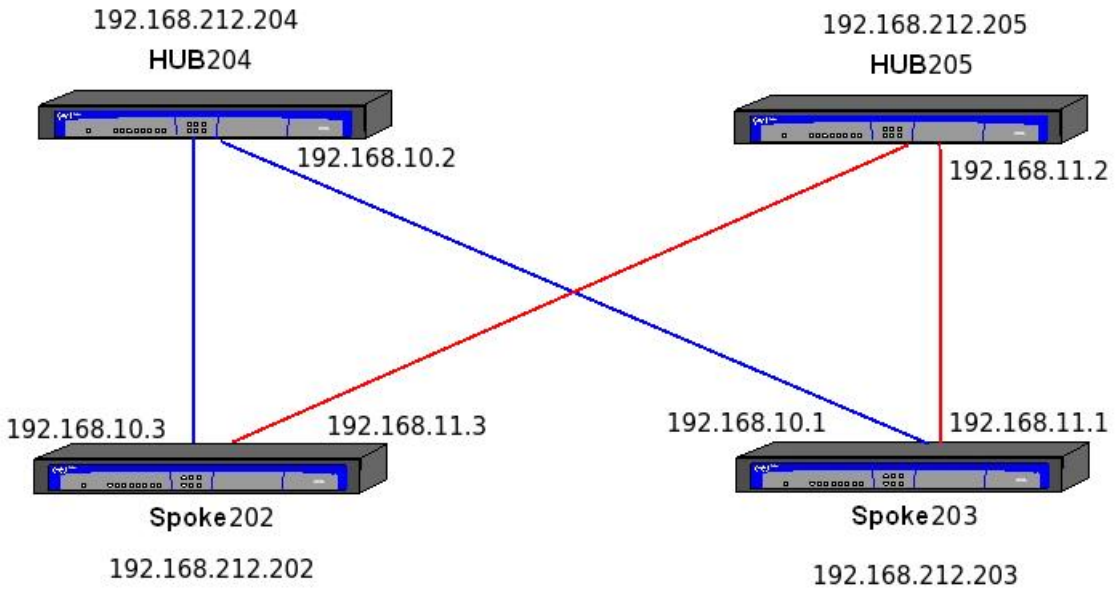
```
      enable
      mode gre multipoint
      source ethernet0/0.10
      vrf-encap VRFE1
      no ip icmp redirects
      nhrp enable
      nhrp holdtime 90
      nhrp map multicast 10.1.1.1
      nhrp map 10.1.1.1 255.255.255.0 192.168.212.201
      nhrp nhs 10.1.1.1
      encapsulation
; -- GRE Configuration -

         key 2010
      exit
;
   exit
;
   network tnip2
; -- IP Tunnel Net Configuration --
      ip vrf forwarding VRFI2
;
      ip address 10.1.2.3 255.255.255.0
;
;
;
;
;
      enable
      mode gre multipoint
      source ethernet0/0.20
      vrf-encap VRFE2
      no ip icmp redirects
      nhrp enable
      nhrp holdtime 90
      nhrp map multicast 10.1.2.1
      nhrp map 10.1.2.1 255.255.255.0 192.168.212.201
      nhrp nhs 10.1.2.1
      encapsulation
; -- GRE Configuration --
         key 2012
      exit
;
   exit
;
   network loopback1
; -- Loopback interface configuration --
      ip vrf forwarding VRFI1
;
      ip address 192.168.3.1 255.255.255.0
;
;
;
;
;
   exit
;
   network loopback2
; -- Loopback interface configuration --
      ip vrf forwarding VRFI2
;
      ip address 192.168.3.1 255.255.255.0
;
;
;
;
;
```

```
   exit
;
   network ethernet0/0.10
; -- Ethernet Subinterface Configuration --
      ip vrf forwarding VRFE1
;
      ip address 192.168.212.203 255.255.254.0
;
;
;
;
      encapsulation dot1q 10
;
;
;
;
   exit
;
   network ethernet0/0.20
; -- Ethernet Subinterface Configuration --
      ip vrf forwarding VRFE2
;
      ip address 192.168.212.203 255.255.254.0
;
;
;
;
      encapsulation dot1q 20
;
;
;
;
   exit
;
   event
; -- ELS Config --
      enable trace subsystem TNIP ALL
      enable trace subsystem IKE ALL
      enable trace event IPSEC.031
   exit
;
;
   protocol ip
; -- Internet protocol user configuration --
      ipsec
; -- IPSec user configuration --
         enable
         assign-access-list 100
;
         template 1 isakmp des md5
         template 1 life duration seconds 1d
         template 1 ike natt-version rfc
         template 1 keepalive dpd
;
         template 2 dynamic esp tdes sha1
         template 2 source-address 192.168.212.203
         template 2 encap transport
         template 2 vrf VRFE1
;
         map-template 100 2
         key preshared ip 0.0.0.0 ciphered 0xD0A99FF6F5D7FA37
         advanced renegotiation-time 100
         advanced pkt-dest-isakmp-dest
         advanced dpd no always-send
      exit
```

```
;
;
      vrf VRFI1
         route 192.168.1.0 255.255.255.0 10.1.1.1
         route 192.168.2.0 255.255.255.0 10.1.1.1
;
;
      exit
;
;
      vrf VRFI2
         route 192.168.1.0 255.255.255.0 10.1.2.1
         route 192.168.2.0 255.255.255.0 10.1.2.1
;
;
      exit
;
;
;
   exit
;
;
;
   dump-command-errors
   end
NHRP-VRF-SPOKE-2 Config
```

## 4.3 DMVPN with IPSec tunnel protection: Example

This scenario is made up of four routers. Two routers act as HUBS and the other two as SPOKES. An IP tunnel has been defined from each spoke to each hub, protected by a different IPSec template. Since the source of both IP tunnels is the same, you need to configure the shared keyword in the templates tunnel-protection option.



### 4.3.1 Spoke 203 configuration

```
   log-command-errors
   no configuration
   add device tnip 1
   add device tnip 2
   set data-link sync serial0/0
   set data-link sync serial0/1
   set hostname SPOKE203
   set inactivity-timer disabled

   network ethernet0/0
; -- Ethernet Interface User Configuration --
      ip address 192.168.212.203 255.255.252.0
      ip address 10.10.10.3 255.255.255.0 secondary
;
   exit

   network tnip1
; -- IP Tunnel Net Configuration --
      description "tunnel with HUB204"
;
      ip address 110.168.10.1 255.255.255.0
;
      mode gre multipoint
      source ethernet0/0
      protection-ipsec
      nhrp enable
      nhrp authentication test
      nhrp holdtime 300
      nhrp map multicast 192.168.212.204
      nhrp map 192.168.10.2 255.255.255.255 192.168.212.204
      nhrp nhs 192.168.10.2
      nhrp track-status
      encapsulation
; -- GRE Configuration --
         key 10000
      exit
;
   exit
;
   network tnip2
; -- IP Tunnel Net Configuration --
      description "tunnel with HUB205"
;
      ip address 192.168.11.1 255.255.255.0
;
      mode gre multipoint
      source ethernet0/0
      protection-ipsec
      nhrp enable
      nhrp authentication test
      nhrp holdtime 300
      nhrp map multicast 192.168.212.205
      nhrp map 192.168.11.2 255.255.255.255 192.168.212.205
      nhrp nhs 192.168.11.2
      encapsulation
; -- GRE Configuration --
         key 10001
      exit
;
   exit
;
   protocol ip
; -- Internet protocol user configuration --
      ipsec
```

```
; -- IPSec user configuration --
        enable
;
        template 1 isakmp tdes sha1
        template 1 ike natt-version rfc
        template 1 ike group two
        template 1 keepalive dpd
        template 1 send-original-pkt
;
        template 2 dynamic esp tdes sha1
        template 2 encap transport
        template 2 tunnel-protection tnip2 shared
;
        template 3 dynamic esp tdes sha1
        template 3 encap transport
        template 3 tunnel-protection tnip1 shared
;
        key preshared ip 0.0.0.0 ciphered 0xE51A70141339BFED
        advanced dpd no always-send
     exit
;
   exit
;
;
   dump-command-errors
   end
```

### 4.3.2 Hub 205 configuration

```
   log-command-errors
   no configuration
   add device tnip 1
   set data-link sync serial0/0
   set hostname HUB205
   set inactivity-timer disabled
;
   network ethernet0/0
; -- Ethernet Interface User Configuration --
      ip address 192.168.212.205 255.255.252.0
;
   exit
;
;
;
   network tnip1
; -- IP Tunnel Net Configuration --
      ip address 192.168.11.2 255.255.255.0
;
      mode gre multipoint
      source ethernet0/0
      protection-ipsec
      nhrp enable
      nhrp authentication test
      nhrp holdtime 300
      nhrp map multicast dynamic
      nhrp track-status
      encapsulation
; -- GRE Configuration --
         key 10001
      exit
;
   exit
;
   protocol ip
; -- Internet protocol user configuration --

      ipsec
; -- IPSec user configuration --
         enable
;
         template 1 isakmp tdes sha1
         template 1 ike natt-version rfc
         template 1 ike group two
         template 1 keepalive dpd
;
         template 2 dynamic esp tdes sha1
         template 2 encap transport
         template 2 tunnel-protection tnip1
;
         key preshared ip 0.0.0.0 ciphered 0xE51A70141339BFED
         advanced dpd no always-send
      exit
;
   exit
;
;
;
;
   dump-command-errors
   end
```