IPSECME Working Group F. Detienne
Internet-Draft M. Kumar
Intended status: Standards Track M. Sullenberger
Expires: January 30, 2014 Cisco
July 29, 2013

Flexible Dynamic Mesh VPN
draft-detienne-dmvpn-00

Abstract

The purpose of a Dynamic Mesh VPN (DMVPN) is to allow IPsec/IKE Security Gateways administrators to configure the devices in a partial mesh (often a simple star topology called Hub-Spokes) and let the Security Gateways establish direct protected tunnels called Shortcut Tunnels. These Shortcut Tunnels are dynamically created when traffic flows and are protected by IPsec.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document describes a Dynamic Mesh VPN (DMVPN), in which an initial partial mesh expands to create direct connections called Shortcut Tunnels between endpoints that need to exchange data but are not directly connected in the initial mesh.

In a generic manner, DMVPN topologies initialize as Hub-Spoke networks where Spoke Security Gateway nodes S* connect to Hub Security Gateway nodes H* over a public transport network (such as the Internet) considered insufficiently secure so as to mandate the use of IPsec and IKE. For scalability and redundancy reasons, there may be multiple hubs; the Hubs would then be connected together through the DMVPN. The diagram Figure 1 depicts this situation.

```
DC1      DC2
 |        |
[H1]-----[H2]
```

```
                            |  |     |  |
                          +-+  |     |  +-+
                          |    |     |    |
                         [S1] [S2]  [S3] [S4]
                          |    |     |
                          D1   D2    D3
```

Figure 1: Hub and Spoke, multiple hubs, multiple spokes

Initially, the Security Gateway nodes (S*) are configured to build tunnels secured with IPsec to the Security Gateway node (H*) in a hub and spoke style network (any partial mesh will do, but Hub-Spoke is common and easily understood). This initial network is then used when traffic starts flowing between the protected networks D*. DMVPN uses NHRP as a signaling mechanism over the S*-H* and H*-H* tunnels to trigger the spokes (S*) to discover each other and build dynamic, direct Shortcut Tunnels. The Shortcut Tunnels allow those spokes to communicate directly with each other without forwarding traffic through the hub, essentially creating a dynamic mesh.

The spokes can be either routers or firewalls playing the role of Security Gateways or hosts such as computers, mobile phones,etc. protecting their own traffic. Nodes S1, S2 and S3 above are routers while S4 is a host implementation.

This document describes how NHRP is modified and augmented to allow the rapid creation of dynamic IPsec tunnels between two devices. Throughout this document, we will call these devices participating in the DMVPN "nodes".

In the context of this document, the nodes protect a topologically dispersed Private, Overlay Network address space. The nodes allow the devices in the Overlay Network to communicate securely with each other via GRE tunnels secured by IPsec using dynamic tunnels established between the nodes over the (presumably insecure) Transport network. I.e. the protected tunnel packets are forwarded over this Transport network.

The NBMA Next Hop Resolution Protocol (NHRP) as described in [RFC2332] allows an ingress node to determine the internetworking layer address and NBMA address of an egress node. The servers in such an NBMA network provide the functionality of address resolution based on a cache which contains protocol layer address to NBMA subnetwork layer address resolution information. This can be used to create a virtual network where dynamic virtual circuits can be created on an as needed basis. In this document, we will depart the underlying notion of a centralized NHS.

All data traffic, NHRP frames and other control traffic needed by this DMVPN MUST be protected by IPsec. In order to efficiently support Layer 2 based protocols, all packets and frames MUST be encapsulated in GRE ([RFC2784]) first; the resulting GRE packet then MUST be protected by IPsec. IPsec transport mode MUST be supported while IPsec tunnel mode MAY be used. The usage of a GRE encapsulation protected by IPsec is described in [RFC4301]. Implementations SHOULD strongly link GRE and IPsec SA's through some form of connection latching as described in [RFC5660].

2. Terminology

The NHRP semantic is used throughout this document however some additional terminology is used to better fit to the context.

- o Protected Network, Private Network: a network hosted by one of the nodes. The protected network IP addresses are those that are resolved by NHRP into an NBMA address.
- o Overlay Network: the entire network composed with the Protected Networks and the IP addresses installed on the Tunnel interfaces instantiating the DMVPN.
- o Transport Network, Public Network: the network transporting the GRE/IPsec packets.
- o Nodes: the devices connected by the DMVPN that implement NHRP, GRE /IPsec and IKE.
- o Ingress Node: The NHRP node that takes data packets from off of the DMVPN and injects them into the DMVPN on either a multi-hop tunnel path (initially) or single hop shortcut tunnel. Also the node that will send an NHRP Resolution Request and receive an NHRP Resolution Reply to build a short-cut tunnel.
- o Egress Node: The NHRP node that extracts data packets from the DMVPN and forwards them off of the DMVPN. Also the node that answers an NHRP Resolution Request and send an NHRP Resolution Reply.
- o Intermediate Node: An NHRP node that is in the middle of multi-hop tunnel path between an Ingress and Egress Node. For the particular data traffic in question the Intermediate node will receive packets from the DMVPN and resend them (hair-pin) them back onto the DMVPN.

Note, a particular node in the DMVPN, may at the same time be an Ingress, Egress and Intermediate node depending on the data traffic flow being looked at.

In general, DMVPN nodes make extensive use of the Local Address Groups (LAG) and Logically Independent Subnets (LIS)models as described in [RFC2332]. A compliant implementation MUST support the LAG model and SHOULD support the LIS model.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Tunnel Types

The tunnels described in this document are of type GRE/IPsec. GRE/IPsec allows a single pair of IPsec SA's to be negotiated between the DMVPN nodes. From an IPsec aggregation standpoint, this means less negotiation, cleaner use of expensive resources and less reprogramming of the data plane by the IKE control plane as additional networks are discovered between any two peers.

In the remainder of this document, GRE and GRE/IPsec will be used interchangeably depending on the focused layer but always imply "GRE protected by IPsec"

Taking advantage of the GRE encapsulation, and while NHRP could be forwarded over IP, the RFC recommended Layer 2 NHRP frames have been retained in order to simplify the security policies (packet filters do not have to be augmented to allow NHRP through, no risk of mistakenly propagating frames where they should not, etc.). Compliant implementations MUST support L2 NHRP frames.

DMVPN can be implemented in a number of ways and this document places no restriction on the actual implementation. This section covers what the authors believe are the important implementation recommendations to construct a scalable implementation.

The authors recommend using a logical interface construct to represent the GRE tunnels. These interfaces are called Tunnel Interfaces or simply Interfaces from here onward.

In the remainder of this document, we will assume the implementation uses point-to-point Tunnel Interfaces; routes to prefixes in the Overlay network are in the Routing Table (aka Routing Information Base). These routes forward traffic toward the tunnel interfaces.

Point-to-Multipoint GRE interfaces (aka multipoint interfaces for short) can also be used. In that case there is by construction only one tunnel source NBMA address and the interface has multiple tunnel endpoints. In this case NHRP registration request and reply messages, [RFC2332], are used to pass the tunnel address to tunnel NBMA address mapping from the NHC (S*) to the NHS (H*). The NHRP registration request and reply MAY be restricted to a single direct tunnel hop between the NHC (S*) and NHS (H*).

For didactic reasons, and an easier understanding of the LAG support, we will use the point-to-point construct to highlight the protocol behavior in the remainder of this document. An implementation can use different models (point-to-point, multipoint, bump in the stack,...) but MUST comply to the external (protocol level) behavior described in this document.

4. Solution Overview

4.1. Initial Connectivity

We assume the following scenario where nodes (S1, S2, H1, H2) depicted in figure Figure 2 supporting GRE, IPsec/IKE and NHRP establish connections instantiated by GRE tunnels. Those GRE tunnels SHOULD be protected by IPsec/IKE. These tunnels will be used to secure all the data traffic as well as the NHRP control frames. In general, routing protocols (and possibly other control protocols) will also run through these tunnels, and therefore also be protected.

```
           DC1
            |
          [H1]
           |  |       ]
         +-+  +-+     ] GRE/IPsec tunnels over Transport network
         |      |     ]
        [S1]   [S2]
         |      |
         D1     D2
```

Figure 2: Hub and Spoke Initial Connectivity

It is assumed that S1, H1 and S2 are connected via a shared Transport network (typically a Public, NBMA network) and there is connectivity between the nodes over that transport network.

The nodes possess multiple interfaces; each of which has a dedicated IP address:

- o a public interface IntPub connected to the transport network; IP address: Pub{node}
- o one or several tunnel interface Tunnel0,1,.. (GRE/IPsec) connecting to peers; IP address: Tun{i}{node}
- o a private interface IntPriv facing the private network of the node; IP address: Priv{node}

e.g. node S1 owns the following addresses: PubS1, TunS1 and PrivS1

The networks D1, D2, DC1 and also the tunnel address Tun{i} can and are presumed to be private in the sense that their address space is kept independent from the transport network address space. Together, they form the Overlay network. For the transport network, the address family is either IPv4 or IPv6. In the context of this document, for the overlay network, the address family is IPv4 and/or IPv6.

Initially, nodes S1 and S2 create a connection to node H1. Optionally, S1 and S2 MAY register to H1 via NHRP. Typically the GRE tunnels between S* and H1 will be protected by IPsec. A compliant implementation MUST support IPsec protected GRE tunnels and SHOULD support unprotected GRE tunnels.

At the end of this section, a dynamic tunnel will be set up between S1 and S2 and traffic will flow directly through S1 and S2 without going through H1.

4.2. Initial Routing Table Status

In the context of this document, the authors make no assumption about how the routing tables are initially populated but one can assume that routing protocols exchange information between H1 and S1 and between H1 and S2.

In this diagram, we assume each node has routes (summarized or specific) for networks D1, D2, DC1 which are IP networks. We assume the summary prefix SUM to encompass all the private networks depicted on this diagram. We assume the communication between those networks needs to be protected and therefore, the routes point to tunnels. I.e. S1 knows a route summarizing all the Overlay subnets and this route points to the GRE/IPsec tunnel leading to H1. Note, the the summary prefix is a network design choice and it can be replaced by a prefix summary manifold or individual non-summarized routes.

Example 1: Node S1 has the following routing table:

o TunH1 => Tunnel0
o SUM => TunH1 on Tunnel0
o 0.0.0.0/0 => IntPub
o D1 => IntPriv

Example 2: Node H1 has the following routing table:

o TunS1 => Tunnel1
o TunS2 => Tunnel2
o D1 => TunS1 on Tunnel1
o D2 => TunS2 on Tunnel2

- o 0.0.0.0/0 => IntPub
- o DC1 => IntPriv

The exact format of the routing table is implementation dependent but the node discovery principle MUST be enforced and the implementation MUST be compatible with an implementation using the routing tables outlined above.

This document does not specify how the routes are installed but it can be assumed that the routes (1) and (2) in the tables above are exchanged between S* and H* nodes after the S*-H* connections have been duly authenticated. In a DMVPN solution, it is typical that the routes are exchanged by a route exchange protocol (e.g. BGP) or are installed statically (usually a mix of both). It is important that routing updates be filtered in order to prevent a node from advertising improper routes to another node. This filtering is out of the scope of this document as most routing protocol implementations are already capable of such filtering. In order to meet these criteria, an implementation SHOULD offer identity-based policies to filter those routes on a per peer basis.

When a device Ds on network D1 needs to connect to a device Dd on network D2

- o a data packet ip(Ds, Dd) is sent and reaches S1 on IntPriv
- o the data packet is routed by S1 via Tunnel0 toward H1; S1 encapsulates, protects and forwards this packet out IntPub via the transport network to H1
- o H1 receives the protected packet on IntPub; H1 decrypts and decapsulates this packet; the resulting data packet looks to the IP stack on H1 as if it arrived on interface Tunnel1
- o the data packet is routed by H1 via Tunnel2 toward S2; H1 encapsulates, protects and forwards this out IntPub via the transport network to S2
- o S2 receives the protected packet on IntPub; S2 decrypts and decapsulates this packet; the resulting data packet looks to the IP stack as if it arrived on interface Tunnel0
- o S2 routes the data packet out of its IntPriv interface to the destination Dd

4.3. Indirection Notification

Considering the packet flow seen in {previous section}. When H1 (Intermediate Node) receives a packet from the ingress node S1 and forwards it to the Next Node S2, it technically re-injects the packet back into the DMVPN.

At this point H1 SHOULD an Indirection Notification message to S1. The Indirection Notification is a dedicated NHRP message indicating the ingress node that it sent an IP packet that had to be forwarded via the intermediate node to another node. The Indirection Notification MUST contain the first 64 bytes of the clear text IP packet that was forwarded to the next node. The exact format of this message is detailed in the section [PACKET_FORMAT].

The Indirection Notification MUST be sent back to the ingress node through the same GRE/IPsec tunnel upon which the hair-pinned IP packet was received and MUST be rate limited.

This message is a hint that a direct tunnel SHOULD be built between the end-nodes, bypassing intermediate nodes. This tunnel is called a "Shortcut Tunnel".

Compliant implementations MUST be able to send and accept the Indirection Notification, however implementations MUST continue to accept traffic over the spoke-hub-spoke path during spoke-spoke path establishment (Shortcut Tunnel).

When a node receives such a notification, it MUST perform the following:

- parse and accept the message
- extract the source address of the original protected IP packet from the 64 bytes available
- perform a route lookup on this source address

  * If the routing to this source address is also via the DMVPN network upon which it received the Indirect Notification then this node is an intermediate node on the tunnel path from the ingress node (injection point) to the egress node (extraction point). In this case this intermediate node MUST silently drop the Indirect Notification that it received. Note that if the node is an intermediate node, it is likely that it has generated and sent an Indirect Notification about this same protected IP packet to its tunnel neighbor on the tunnel path back towards the ingress node (injection point). This is correct behavior.
- if the previous step did succeed, extract the destination IP address (Dd) of the original protected IP packet from the 64 bytes available.

The ingress node MAY also extract additional information from those 64 bytes such as the protocol type, port numbers etc.

In steady state, Indirection Notifications MUST be accepted and processed as above from any trusted peer with which the node has a direct connection.

4.4. Node Discovery via Resolution Request

After processing the information in the Indirection Notify, the ingress node local policy SHOULD determine whether a shortcut tunnel needs to be established. Assuming the local policy requests a shortcut tunnel, the ingress node MUST emit a Resolution Request for the destination IP address Dd.

More specifically, the NHRP Resolution Request emitted by S1 to resolve Dd will contain the following fields:

- Fixed Header
  - ar$op.version = 1
  - ar$op.type = 1
- Common Header (Mandatory Header)
  - Source NBMA Address = PubS1
  - Source Protocol Address = TunS1
  - Destination Protocol Address = Dd

The resolution request is routed by S1 to H1 over the GRE/IPsec tunnel. If an intermediate node has a valid (authoritative) NHRP mapping in its cache, it MAY respond. An intermediate node SHOULD NOT answer Resolution Requests in any other case.

Note that a Resolution Request can be voluntarily emitted by Security Gateway and is not strictly limited to a response to the Indirection Notify message. Such cases and policies are out of the scope of the document.

The sending of Resolution Requests by a ingress node MUST be rate limited.

4.5. Resolution Request Forwarding

The Resolution Request can be sent by S1 to an explicit or implicit next-hop-server. In the explicit scenario, the NHS is defined in the node configuration. In the implicit case, the node can infer the NHS to use. Similarly, an intermediate node that cannot answer a Resolution Request SHOULD forward the Resolution Request to an implicit or explicit NHS in the same manner unless local policy forbids resolution forwarding between Spokes. There can be an undetermined number of intermediate node.

A DMVPN compliant implementation MUST be able to infer the NHS from its routing table in the following way:

- o the address Dd to be resolved is looked up in the routing table (other parameters can be considered by the ingress node but these will not be available to intermediate nodes)
- o the best route for Dd is selected (longest prefix match)
  - * if several routes match (same prefix length) only the routes pointing to a DMVPN Tunnel interface are kept. This SHOULD NOT occur in practice.
- o if the best route found points to a DMVPN Tunnel interface, the next-hop address MUST be used as NHS
- o if the best route found does not point to a DMVPN Tunnel interface the forwarding of the packet stops and the matching prefix P and prefix len (Plen) is kept temporarily. Very often, P/Plen == D2/D2len (this is the case in the diagram used in this document) but this may not always be true depending on the structure of the networks protected by S2. The associated prefix length (Plen) is also preserved.

If the Resolution Request forwarding stops at the ingress node (at emission), the Resolution Request process MUST be stopped with an error for address Dd. If the lookup succeeds, the next-hop's NBMA address is used as destination address of the GRE encapsulation. Before forwarding, each intermediate node MUST add a Forward Transit Extension record to the NHRP Resolution Request.

Any intermediate nodes SHOULD NOT cache any information while forwarding Resolution Requests. In the case an intermediate node implementation caches information, it MUST NOT assume that other intermediate nodes will also cache that information.

Thanks to the forwarding model described in this document and due to the absence of intermediate caching, Server Cache Synchronization is not needed and even recommended against. Therefore, a DMVPN compliant implementation MUST NOT rely on such a synchronization which would have adverse effects on the scalability of the entire system.

If the TTL of the request drops to zero or the current node finds itself on a Forward Transit Extension record then the NHRP Resolution Request MUST be dropped and an NHRP error message sent to the source.

When the Resolution Request eventually reaches a node where the route(s) to the destination would take it out through a non-DMVPN interface, the Resolution Request process MUST be stopped and this node becomes the egress node. The egress node is typically (by

virtue of network design) the topologically closest node to the resolved address Dd.

The egress node must then prepare itself for replying with a Resolution Reply.

4.6. Egress node NHRP cache and Tunnel Creation

When a node declares itself an egress node while attempting to forward a Resolution Request, it MUST evaluate the need for establishing a shortcut tunnel according to a user policy. Note that an implementation is not mandated to support a user policy but then the implicit policy MUST request the shortcut establishment. If policies are supported, one of the possible policies MUST be shortcut establishment.

If a shortcut is required, the egress node MUST perform the following operations:

- o the source NBMA address (PubS1) is extracted from the NHRP Resolution Request
- o if a GRE/IPsec tunnel already exists between PubS2 and PubS1, this tunnel is selected (assuming interface TunnelX)
- o otherwise, a new GRE shortcut tunnel is created between PubS2 and PubS1 (assuming interface TunnelX); the GRE tunnel SHOULD be protected by IPsec and the SA's immediately negotiated by IKE
- o an NHRP cache entry is created for TunS1 => PubS1. The entry SHOULD NOT remain in the cache for more than the specified Hold Time (from the NHRP Resolution Request). This NHRP cache entry may be 'refreshed' for another hold time period prior to expiry by receipt of another matching NHRP Resolution Request or by sending an NHRP Resolution Request and receiving an NHRP Resolution Reply.
- o a route is inserted into the RIB: TunS1/32 => PubS1 on TunnelX (assuming IPv4)

Regardless how the shortcut tunnel is created a node SHOULD NOT try to establish more than one tunnel with a remote node. If there are other tunnels not managed by DMVPN, the tunnel selectors (source, destination, tunnel key) MUST NOT interfere with the DMVPN shortcut tunnels.

If a tunnel has to be created and SA's established, a node SHOULD wait for the tunnel to be in place before proceeding with further operations. Regardless of how those operations are timed in the implementation, a node SHOULD avoid dropping data packets during the cache and SA installation. The order of operations SHOULD ensure continuous forwarding.

4.7. Resolution Reply format and processing

After the operations described in the previous section are completed, a Resolution Reply MUST be emitted by the egress node. Instead of strictly answering with just the host address being looked up, the Reply will contain the entire prefix (P/Plen) that was found during the stopped Resolution Request forwarding phase.

The Resolution Reply main fields MUST be populated as follows:

- o Fixed Header
    * ar$op.version = 1
    * ar$op.type = 2
- o Common Header (Mandatory Header)
    * Source NBMA Address = PubS1
    * Source Protocol Address = TunS1
    * Destination Protocol Address = Dd
- o CIE-1
    * Prefix-len = Plen
    * Client NBMA Address = PubS2
    * Client Protocol Address = TunS2

The Destination Protocol address remains the address being resolved (Dd) while the CIE actually contains the remainder of the response (Plen via NBMA PubS2, Protocol TunS2). The Resolution Reply MUST be forwarded to the ingress node S1 either through the shortcut tunnel or via the Hub.

If the address family of the resolved address Dd is IPv6, the Resolution Reply SHOULD be augmented with a second CIE containing the egress node's link local address.

If a node decides to block the resolution process, it MAY simply drop the Resolution Request or avoid sending a Resolution Reply. A node MAY also send a NACK Resolution Reply.

When the Resolution Reply is received by the ingress node, a new tunnel TunnelY MUST be created pointing to PubS2 if one does not already exist (which depends on whether the Resolution Reply was routed via the Hub(s) or directly on the shortcut tunnel). The ingress node MUST process the reply in the following way:

- o Validate that this Resolution Reply corresponds to a Request emitted by S1. If not, issue an error and stop processing the Reply.

o An NHRP Cache entry is created for TunS2 => PubS2
o Two routes are added to the routing table:

  * TunS2 => TunnelY
  * P/Plen => TunS2 on TunnelY

Though implementations may be entirely different, a compliant implementation MUST exhibit a functional behavior strictly equivalent to the one described above. I.e. IP packets MUST eventually be forwarded according to the above implementation.

DMVPN compliant implementations MUST support providing and receiving aggregated address resolution information.

4.8. From Hub and Spoke to Dynamic Mesh

At the end of the resolution process, the overlay topology will be as follows:

```
           DC1
            |
          [H1]
          |  |        ]
        +-+  +-+      ] GRE/IPsec tunnels over Transport network
        |      |      ]
       [S1]===[S2]
        |      |
        D1     D2
```

Shortcut tunnel established

Where the tunnel depicted with = is a GRE/IPsec shortcut tunnel created by NHRP. The Routing Table on S1 will now look as follows:

o TunH1 => Tunnel0
o SUM => TunH1 on Tunnel0
o 0.0.0.0/0 => IntPub
o D1 => IntPriv
o TunS2 => TunnelY
o P/Plen => TunS2 on TunnelY

It is easy to see that traffic from D1 to D2 will follow the shortcut path under the assumption that P == D2 or D2 is a subnet included in P.

The tunnels between S* and H* are actually tunnels created automatically to bootstrap the DMVPN. In practice the initial topology will be a static star (aka Hub and Spoke) topology between S* and H* that will evolve into a dynamic mesh between the nodes S*.

From the spokes (S*) standpoint, the bootstrap tunnels can be established with a node H1 statically defined or discovered by DNS. The problem of finding the initial hubs in a DMVPN is not different than finding regular hubs in a traditional Hub and Spoke network.

For scalability reasons, it is expected that the NHRP Indirection/ Resolution is the only way by which routes are exchanged between S* nodes. While this does not fall in the context of this document, it is worth mentioning that actual implementations SHOULD NOT establish a routing protocol adjacency directly over the shortcut tunnels.

4.9. Remote Access Clients

The specification in this document allows a node to not protect any private network. I.e. in a degenerate case, it MUST be possible for a node S1 to not have a D1 network attached to it. Instead, S1 only owns a PubS1? and TunS1? address. This would typically the case of a remote access client (PC, mobile device,...) that only has a tunnel address and an NBMA address.

```
           DC1
            |
          [H1]
          |  |       ]
        +-+  +-+     ] GRE/IPsec tunnels over Transport network
        |      |     ]
       [S1]===[S2]
               |
               D2
```

Remote Access Client

On the diagram above, S1 is actually a simple PC or mobile node that is not protecting any other network other than its own tunnel address.

These nodes may fully participate in a DMVPN network, including building spoke-spoke tunnels as long as they support GRE, NHRP, IPsec /IKE, and have a way to separate tunneled traffic (virtual interfaces) and be able to update a local routing table to associate networks with different next-hops out either their IntTun (data traffic going over the tunnel) or (IntPub) (tunnel packets themselves and/or non-tunneled data traffic). They may not need to run a routing protocol.

4.10. Node mutual authentication

Nodes authenticate each other using the IKE protocol, while they attempt to establish a tunnel. Because the system is by nature extremely distributed, it is recommended to use X.509 certificates for authentication. Internet Public Key Infrastructure is described in [RFC5280]

The structured names and various fields in the certificate can be useful for filtering undesired connectivity in large administrative domains or when two domains are being partially merged. It is indeed easy for a system administrator to define filters to prevent connectivity between nodes that are not supposed to communicate directly (e.g. filtering based on the O or OU fields).

Though nodes may be blocked from building a direct tunnel by the above means they may or may not be allowed to communicate via a spoke-hub-spoke path. Allowing or blocking communication via the spoke-hub-spoke path is outside the scope of this document.

5. Packet Formats

As described in [RFC2332], an NHRP packet consists of a fixed part, a mandatory part and an extensions part. The Fixed Part is common to all NHRP packet types. The Mandatory Part MUST be present, but varies depending on packet type. The Extensions Part also varies depending on packet type, and need not be present. This section describes the packet format of the new messages introduced as well as extensions to the existing packet types.

5.1. NHRP Traffic Indication

The fixed part of an NHRP Traffic Indication packet picks itself directly from the standard NHRP fixed part and all fields pick up the same meaning as in [RFC2332] unless otherwise explicitly stated.

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

```
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |            ar$afn             |          ar$pro.type          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                          ar$pro.snap                          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |  ar$pro.snap  |   ar$hopcnt   |            ar$pktsz           |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |           ar$chksum           |            ar$extoff          |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | ar$op.version |   ar$op.type  |    ar$shtl    |    ar$sstl    |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Traffic Indication Fixed Header

o ar$op.type With ar$op.version = 1, this is an NHRP packet. Further, [RFC2332] uses the numbers 1-7 for standard NHRP messages. When ar$op.type = 8, this indicates a traffic indication packet.

The mandatory part of the NHRP Traffic Indication packet is slightly different from the NHRP Resolution/Registration/Purge Request/Reply packets and bears a much closer resemblance with the mandatory part of NHRP Error Indication packet. The mandatory part of an NHRP Traffic Indication has the following format

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      | Src Proto Len | Dst Proto Len |           unused              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |           Traffic Code        |           unused              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |            Source NBMA Address (variable length)              |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |          Source NBMA Subaddress (variable length)             |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |          Source Protocol Address (variable length)            |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |       Destination  Protocol Address (variable length)         |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |       Contents of Data Packet in traffic (variable length)    |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: Traffic Indication Mandatory Part

o Src Proto Len: This field holds the length in octets of the Source Protocol Address.

o Dst Proto Len: This field holds the length in octets of the Destination Protocol Address.
o Traffic Code: A code indicating the type of traffic indication message, chosen from the following list

   * 0: NHRP Traffic Redirect/Indirection message.This indirection is an indication,to the receiver, of the possible existence of a 'better' path in the NBMA network.
o Source NBMA Address: The Source NBMA address field is the address of the station which generated the traffic indication.
o Source NBMA SubAddress: The Source NBMA subaddress field is the address of the station generated the traffic indication. If the field's length as specified in ar$sstl is 0 then no storage is allocated for this address at all.
o Source Protocol Address: This is the protocol address of the station which issued the Traffic Indication packet.
o Destination Protocol Address: This is the destination IP address from the packet which triggered the sending of this Traffic Indication message.

Note that unlike NHRP Resolution/Registration/Purge messages, Traffic Indication message doesn't have a request/reply pair nor does it contain any CIE though it may contain extension records.

6. Security Considerations

The use of NHRP and its protocol extensions described in this document do not open a direct security hole. The peers are duly authenticated with each other by IKE and the traffic is protected by IPsec. The only risk may come from inside the network itself; this is not different from static meshes.

Implementers must be diligent in offering all the control and data plane filtering options that an administrator would need to secure the communication inside the overlay network.

7. IANA Considerations

The following values are used experimentally:

o The ar$op.type value of 8 representing Traffic Indication
o Traffic Code value of 0 indicating a Traffic Indirection message.

Full standardization would require official IANA numbers to be assigned.

8. Match against ADVPN requirements

This section compares the adequacy of DMVPN to the requirement list stated in [ADVPNreq].

8.1. Requirement 1

A new spoke in a DMVPN does not require changes on a hub to which it is connected other than authentication and authorization state which are dynamically handled. No state is required on other hubs because addresses are passed between hubs using NHRP and IKEv2. This requirement is one of the basic features of DMVPN.

8.2. Requirement 2

NHRP is used to distribute dynamic peer NBMA and Overlay addresses. These addresses will be redistributed or rediscovered upon any address change. This requirement is one of the basic features of DMVPN. Practical implementation and deployments already exist that take advantage of this mechanism.

8.3. Requirement 3

DMVPN requires minimal configuration in order to configure protocols running over IPsec tunnels. The tunnels are latched to their crypto socket according to [RFC5660]. The routing protocols or other feature do not even need to be aware of the IPsec layer nor does IPsec need to be aware of the actual traffic it carries. Practical implementation and deployments already exist.

8.4. Requirement 4

Spokes can talk directly to each other if and only if the Hub and Spoke policies allow it. Sections Section 4.6 and Section 4.5 explicitly mention places where such policies should be applied. Practical implementation and deployments already exist that exhibit this form of restriction.

8.5. Requirement 5

Each DMVPN peer has unique authentication credentials and uses them for each peer connection. The credentials do not need to be shared or pre-shared unless the administrator allows it which is out of the scope of this document. To this effect, DMVPN makes great use of certificates as a strong authentication mechanism. Cross-domain authentication is made possible by PKI should the security gateways belong to different PKI domains. Practical implementation and deployments already exist that take advantage of this mechanism.

8.6. Requirement 6

DMVPN Gateways are free to roam. The only requirement is that Spokes update their peers with their new NBMA IP address should it change. Implementations MAY choose to update their peers via MOBIKE but MUST support a re-registration and re-discovery. Roaming across hubs require that the new hub learns the prefixes behind the branch which is what DMVPN does by construction. For supporting roaming hubs changing their NBMA IP address, Hubs' DNS record MUST be updated (the mechanism is not covered in this document) and Spokes MUST be able to resolve a Hub NBMA address by DNS. Practical implementation and deployments already exist.

8.7. Requirement 7

Handoffs are possible and can be initiated by a Hub or a Spoke. At any point in time, a Spoke may create multiple simultaneous connections to several Hubs and change its routing policies to send or receive traffic via any of the active tunnels. If a Hub wishes to offload a connection to another Hub, the Hub can do so by using an IKE REDIRECT as explained in [RFC5685]. Those handoffs are optional and left at the discretion of the implementer. Partial practical implementation and deployments already exist and more get developed on an ad-hoc basis without breaking protocol-level compatibility.

8.8. Requirement 8

DMVPN support gateways behind NAT boxes through the IKEv2 NAT Traversal Exchange. Practical implementation and deployments already exist.

8.9. Requirement 9

Changes of SA are reportable and manageable. This document does not define a MIB nor imposes message formats or protocols (Syslog, Traps,...). All tables such as NHRP, IPsec SA's and routing table are MIB manageable. The creation of IKE sessions triggers messages and NHRP can be instrumented to log and report any necessary event. Practical implementation and deployments already take advantage of those facilities.

8.10. Requirement 10

With an appropriate PKI authorization structure, DMVPN can support allied and federated environments. Practical implementation and deployments already exist.

8.11. Requirement 11

DMVPN supports star, full mesh, or a partial mesh topologies. The protocol stack exposed here can be applied to all known scenarios. Implementers are free to cover and support the adequate use cases. Practical deployments of all those topologies already exist.

8.12. Requirement 12

DMVPN can distribute multicast traffic by taking advantage of protocols such as PIM, IGMP and MSDP. Practical implementation and deployments already exist.

8.13. Requirement 13

DMVPN allows monitoring and logging. All topology changes, connections and disconnections are logged and can be monitored. The DMVPN solution explained in this document does not preclude any form of logging or monitoring and additional monitoring points can be added without impacting interoperability. Practical deployments already exist that take advantage of those facilities.

8.14. Requirement 14

L3VPNs are supported over IPsec/GRE tunnels. The main advantage of a GRE tunnel protected by IPsec is that L2 frames do not need any additional IP encapsulation which means that L2 frames can be natively transported over DMVPN. Practical L3VPN implementation and deployments already exist.

8.15. Requirement 15

DMVPN supports per-peer QoS between Spoke or Hub or between Spokes. The QoS implementation is out of the scope of this document. Practical implementation and deployments already exist.

8.16. Requirement 16

DMVPN allows multiple resiliency mechanisms and no device, Spoke or Hub is a single point of failure by protocol design. Multiple encrypted tunnels can be established between Spokes and Hubs or Hubs can be configured as redundant entities allowing failover. Practical such deployments already exist.

9. Acknowldegements

The authors would like to thank Brian Weis, Mark Comeadow and Mark Jackson from Cisco for their help in publishing and reviewing this

document. We would also like to acknowledge the historical DMVPN team, in particular Jan Vilhuber and Pratima Sethi.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2332] Luciani, J., Katz, D., Piscitello, D., Cole, B., and N. Doraswamy, "NBMA Next Hop Resolution Protocol (NHRP)", RFC 2332, April 1998.

[RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 2784, March 2000.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5660] Williams, N., "IPsec Channels: Connection Latching", RFC 5660, October 2009.

[RFC5685] Devarapalli, V. and K. Weniger, "Redirect Mechanism for the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5685, November 2009.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

10.2. Informative References

[ADVPNreq]
Hanna, S., "Auto Discovery VPN Problem Statement and Requirements", June 2013, <http://tools.ietf.org/html/draft-ietf-ipsecme-p2p-vpn-problem-07.txt>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

Authors' Addresses

Frederic Detienne
Cisco
De Kleetlaan 7
Diegem 1831
Belgium

Email: fd@cisco.com

Manish Kumar
Cisco
Mail Stop BGL14/G/
SEZ Unit, Cessna Business Park
Varthur Hobli, Sarjapur Marathalli Outer Ring Road
Bangalore, Karnataka 560 103
India

Email: manishkr@cisco.com

Mike Sullenberger
Cisco
Mail Stop SJCK/3/1
225 W. Tasman Drive
San Jose, California 95134
United States

Email: mls@cisco.com