Chapter 7
Tunnels

Tunnels are the heart and soul of virtual private networking. This chapter introduces tunnels by describing basic tunnel operation and analyzing several tunnel topologies from both the virtual network and virtual data-link perspectives. Specific protocols, security, performance, and management topics are discussed in later chapters after we understand simple tunnels.

Tunnels can be created for transporting any protocol through nearly any other protocol. This book focuses on tunneling of IP through IP, which is the common method for virtual private networking in the public Internet. The examples in this book show the virtual network layer as a private IP network such as an enterprise intranet, and the underlying network layer as a public IP network such as the Internet. The concepts are easily applied to other combinations of protocols, even to combinations of connection-less and connection-oriented protocols such as IP over X.25.

7.1 Basic Operation

A *network tunnel* behaves like a data-link between two network nodes, but rather than sending the packets through a physical medium, a tunnel encapsulates them and sends them as data through an underlying network. I refer to the network that uses the tunnel service as the *virtual network*, and the network used by the tunnel as the *underlying network*.

Since a tunnel simulates a data-link in the virtual network, a tunnel can also be referred to as a *virtual data-link*. Although *virtual data-link* is more descriptive, I tend to use *tunnel* because it is shorter and more common in other literature.

To better understand tunneling, let us first look at a conventional link in a conventional IP network. Generally, a data communication network is made up of nodes that are interconnected by a variety of communication mediums. In wide area networks, this medium is typically a transmission line. The service provided by the transmission line is enhanced by the data-link protocol layer to create a data-link service. That service is enhanced by the network protocol layer (IP) to provide a network service. That service is enhanced by successively higher protocol layers until it is ultimately used by an application. (See “OSI Layers” on page 6.)

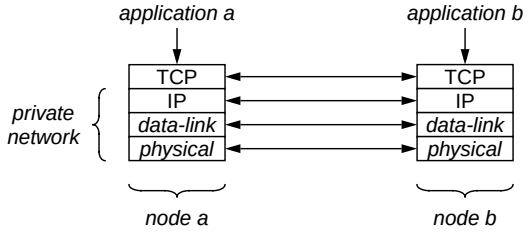

Figure 7.1. Protocol Diagram of a Conventional Data-Link

Figure 7.1 shows the protocol layer relationships of a conventional link. In the diagram, two nodes that are directly connected by a physical data-link form a simple private network. For illustration, an application on each node communicates with the other through a TCP connection.

7.1.1 Generic Tunnel

Now let us compare a conventional data-link to a virtual data-link implemented as an simple tunnel through a public network (such as the Internet). Figure 7.2 shows the same two nodes as Figure 7.1, except that a tunnel has replaced the conventional link. The tunnel protocol layer provides data-link services to the IP network layer just like the data-link protocol layer in the conventional link. However, rather than using the services of a transmission line or other physical communication medium, the tunnel uses the services of an underlying public network.

The tunnel protocol layer operates between two completely separate network layers: the *virtual network*, shown in the diagram as IP_v, and the *underlying network*, shown as IP_u. The two IP layers operate completely independently, having separate network address spaces and separate routing tables. Addresses in one have no meaning in the other. It is not possible for datagrams to pass from one to the other. The underlying network sees the tunnel as an application. It is unaware that the application data stream contains packets for another network. Likewise, the virtual network is unaware that its packets are being carried by another network. The two networks remain completely unaware of each other.

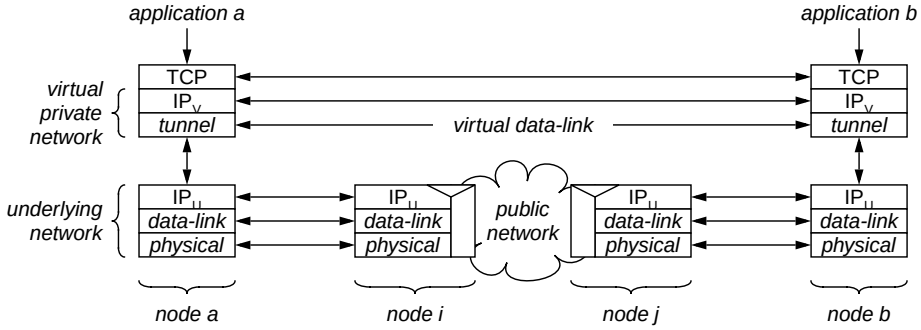

Figure 7.2. Protocol Diagram of a Virtual Data-Link

Having said that, there are some effectively recursive situations where the first network and the underlying network are actually the same network. Section {} will consider such cases, further. For now, we should think of them as separate networks.

If we assume the underlying network (IP_u) is a public network such as the Internet, then we naturally consider the virtual network (IP_v) to be a *virtual private network* (VPN). We say *virtual* because it uses a virtual data-link. We say *private* because the VPN traffic is isolated from the public network by the tunnel layer.

7.1.2 IPSEC Tunnel

The privacy of a simple tunnel can be compromised by any entity in the underlying network that is able to examine (snoop) packets. Likewise, the integrity can be compromised by any entity that is able to modify packets. Introducing cryptography into the tunnel protocol protects the privacy and integrity of the VPN against such attacks from the underlying network. Cryptography is the basic difference between simple tunneling and virtual private networking. {} covers tunnel security in greater detail.

In the previous section, we looked at a generic tunnel. In this section, we look at a specific tunnel based on the IP Security protocol (IPSEC, See Chapter 13). The IPSEC protocol is composed of multiple sublayers that use cryptography to provide authentication, integrity and privacy. Figure 7.3 shows a secure tunnel using IPSEC as the tunnel protocol layer. IPSEC tunnels are likely to be the most popular for VPN since the protocol is already reasonably popular for IPv4, and is required to be included in all implementations of IPv6.

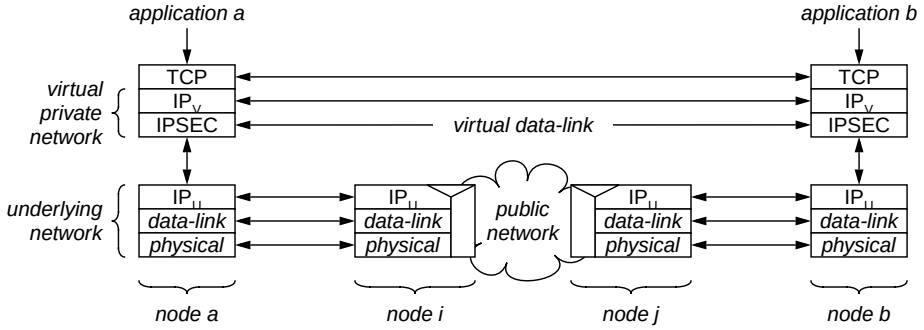

Figure 7.3. Protocol Diagram of an IPSEC Tunnel

7.2 Virtual Network Topology

The topology of a distributed system appears different when viewed from different protocol layers. Each protocol layer hides the topology of the layers below it from view by the layers above it. For example, the TCP transport layer sees a direct logical connection to its peer on the remote end-node, while the IP network layer sees a path that might include several intervening routers. We usually consider topology from the perspective of the network layer. However, tunneling introduces a second network layer, and with it, the potential for confusion.

This section looks at topology from the perspective of the virtual network layer (IP_v in the figures). Following this, Section 7.3 looks at the topology of the virtual data-links, themselves. We could also look at the topology from the perspective of the underlying network layer, but it is irrelevant to VPN design except that it indirectly affects performance and reliability.

The following virtual network topologies are the most basic. They can be used as the building blocks for more complicated topologies, if required, but that is rare. Selecting the topology is one of the most important decisions in VPN design.

7.2.1 End-node to End-node

End-node to end-node (end-to-end) tunneling is the simplest to understand and may provide the greatest protection. In this case, a tunnel directly connects the end-nodes. We have already seen this topology in Figure 7.2.

The protection afforded by the tunneling protocol is maximized in an end-to-end topology because the entire connection between the end-nodes is protected. Unfortunately, the presence of an interface between the end-node and the public network (IP_u) introduces

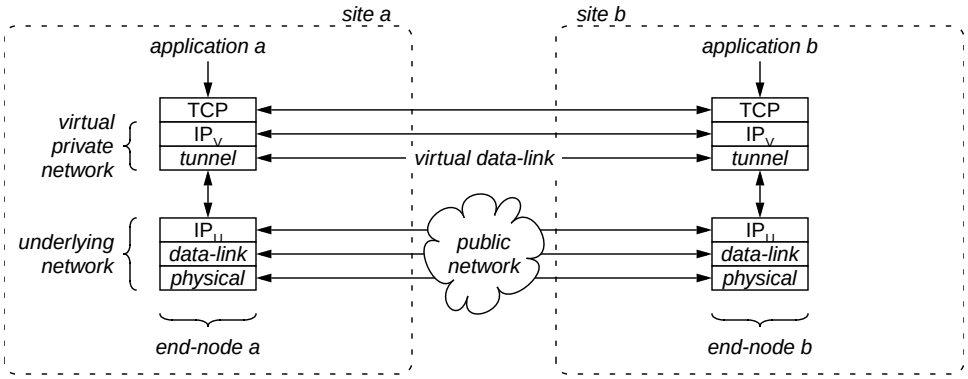

Figure 7.4. End-node to End-node Virtual Data-Link

a new risk by inadvertently making the end-node into a firewall. The problem is exacerbated because users perceive the direct public network connection as a desirable feature. Section 11.1.8 on page 45 covers inadvertent firewalls in more detail.

7.2.2 End-node to Gateway

The end-node to gateway topology changes the end-to-end topology discussed in section 7.2.1 by moving the tunnel endpoint from end-node b to a gateway router. This topology provides a big advantage by allowing many end-nodes to share a single VPN gateway router. The gateway provides a central point for authentication and access control of the tunnels thereby simplifying administration.

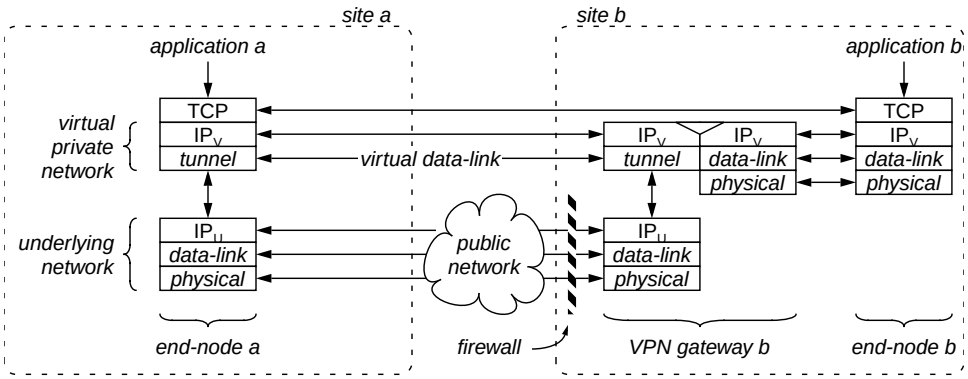

Figure 7.5. End-node to Gateway Virtual Data-Link

If the private network contains a firewall, the VPN gateway should be logically inside the firewall (possibly on the same machine as the firewall). The firewall filters can be set to

permit enciphered tunnel traffic to the gateway without consideration of the traffic's origin. The gateway authenticates the tunnel either directly or as a side effect of cipher key management. The tunnel enciphers traffic between end-node a and the VPN gateway. Traffic between VPN gateway b and end-nodes in site b is not protected.

7.2.3 Gateway to Gateway

The gateway-to-gateway topology terminates each end of the tunnel at a VPN gateway. The subnet between each gateway and its local end-nodes is assumed to be trusted. This topology is an attractive choice for connecting geographically dispersed subnets, such as branch offices, into a private intranet. Figure 7.6 shows the gateway-to-gateway topology.

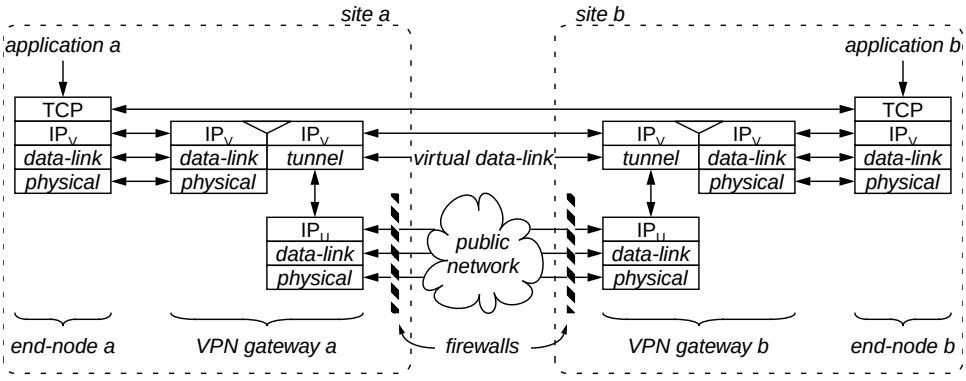

Figure 7.6. Gateway to Gateway Virtual Data-Link

7.2.4 Service-Provider to Gateway

The service-provider to gateway topology is a special case of a gateway-to-gateway topology. The protocol diagram in Figure 7.7 is nearly identical to that shown in Figure 7.6, except that VPN gateway a is now part of the service provided by end-system a's internet service provider (ISP) and is located at the ISP point of presence (POP). End-node a dials into the ISP access router using a modem or some other public switched service. The connection between end-node a and the access router is not protected. When the call is connected to the ISP access router, the router authenticates the caller, looks up the caller's profile in the ISP's database, and creates a tunnel to the VPN gateway specified in the profile. The L2TP and PPTP protocols were designed with this scenario in mind. (See Chapter 14.)

The service-provider to gateway topology is attractive to service providers because it gives them a value added service they can charge a premium for. It is also attractive to law enforcement agencies because they can subpoena the tunnel cipher keys from the service

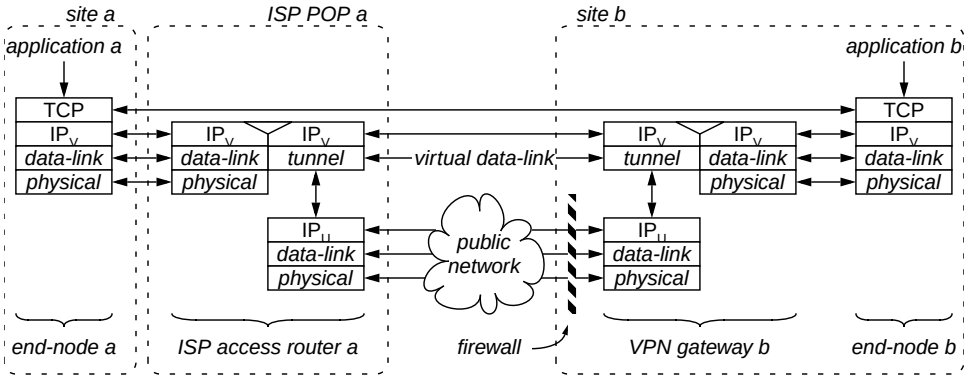

Figure 7.7. Service-Provider to Gateway Virtual Data-Link

provider without tipping off the user. Even beyond the government factor, letting the service provider hold your encryption keys degrades the security of the VPN due to the risk of mismanagement or accidental disclosure. The service provider holding keys for hundreds of companies would be a much bigger target than a single company.

7.2.5 Comparison

Table 7.1 shows a comparison of the four virtual network topologies presented in this section. The comparison assumes site a is a remote user or remote branch office, and site b is a local enterprise intranet. The enterprise intranet, and branch office LAN (if any) are assumed to be relatively trusted.

Table 7.1. Comparison of VPN Topologies

feature	*end-to-end*	*end-to-gateway*	*gateway-to-gateway*	*ISP-to-gateway*
recommended use	high sensitivity[a]	remote access	branch office	low sensitivity
tunnel software[b] required on end-nodes?	yes	remote[c]	no	no
tunnel software[b] required on gateways?	no	local[d]	yes	yes
security relies on 3rd party?	no	no	no	yes[e]
dependence on ISP?	no	no	no	yes[f]

Table 7.1. Comparison of VPN Topologies

feature	*end-to-end*	*end-to-gateway*	*gateway-to-gateway*	*ISP-to-gateway*
Internet segment protected?	yes	yes	yes	public[g]
LAN segment protected?	yes	remote[h]	no	no
scalable to many remote end-nodes?	maybe[i]	maybe[i]	yes	maybe[j]
scalable to many local end-nodes?	maybe[i]	yes	yes	yes

a. Only choice if enterprise intranet and branch office LAN (if any) are not trusted.
b. Includes enciphering software, use and export of which may be restricted by law.
c. Tunnel support needed on *end-node a*.
d. Tunnel support needed on the enterprise firewall or gateway *router b*.
e. ISP manages cipher keys and responsible for half of cipher software.
f. May be difficult to change to a different ISP or use multiple ISPs.
g. VPN is protected from the public part of the Internet, but not from the ISP access routers or terminal servers where the tunnel originates.
h. If any.
i. Depends on scalable key management protocol and software administration.
j. If the remote users share one or two common ISPs.

7.3 Virtual Data-Link Topology

Data-links have one of two basic topologies. They may be point-to-point such as a PPP dialup line or HDLC leased line, or they may be multipoint such as an Ethernet or FDDI ring. The basic distinction is that a node using point-to-point links requires one physical connection for each directly connected remote node, while a node using a multipoint link may use a single physical connection for many directly connected nodes. Link topology can have a substantial effect on cost, manageability and performance. This section compares the effect of configuring tunnels in each topology.

7.3.1 Point-to-point Topology

In the point-to-point topology, the tunnel (virtual data-link) behaves like a point-to-point transmission line, such as an HDLC leased line or PPP dialup line. Point-to-point data-links are characterized by a separate Data-Link Service Access Point (DLSAP) for each remote tunnel endpoint, even though all of the tunnels may share a single underlying

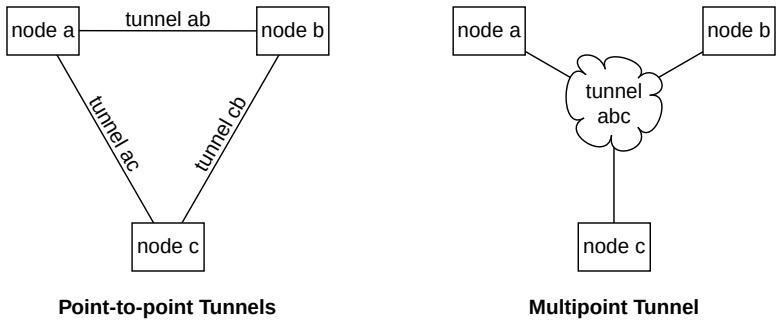

Figure 7.8. Point-to-point vs. Multipoint Topology

network. A DLSAP, which is OSI terminology, (See "OSI Terminology" on page 5.) is also known as a *Data-Link Provider Interface* (DLPI), or simply, *interface*.

Figure 7.9 shows how point-to-point tunnels bind associations in the underlying network to DLSAPs used by the virtual network.

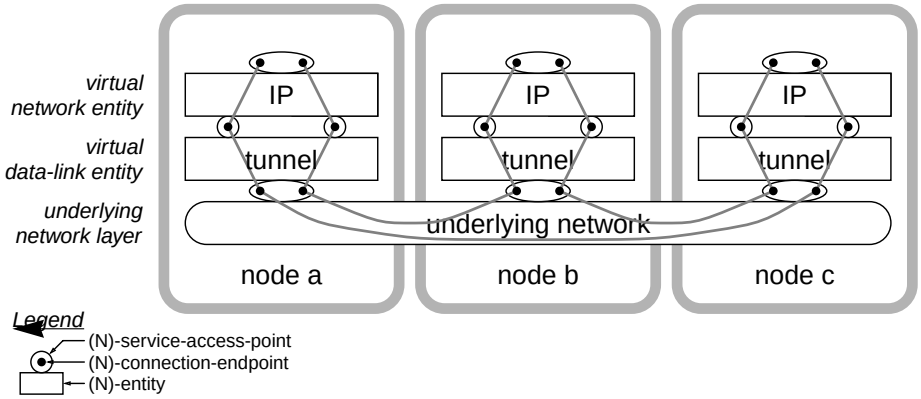

Figure 7.9. Example of Interface Binding for Point-to-Point Tunnels

A point-to-point virtual data-link maps each underlying network connection endpoint (see Figure 7.9) into a separate DLSAP (interface). In an IP network, the network entity maps each data-link interface to one or more network addresses. Most modern IP implementations permit unnumbered interfaces, in which case more than one interface effectively shares one network address. Otherwise, a separate network address is required for each tunnel endpoint. A subnet of n nodes fully interconnected by point-to-point tunnels requires a minimum of $n^2 - n$ DLSAPs (interfaces), $n - 1$ per node.

Configuring a subnet of point-to-point tunnels requires each node to keep a table having one row for each of its neighbors in the subnet, each row containing three parameters as follows:

1. the virtual network address of the local tunnel endpoint

2. the virtual network address of the remote tunnel endpoint

3. the underlying network address of the remote tunnel endpoint

This table is different for every node in the subnet so the nodes may not share a common configuration file. The configuration complexity is 3 parameters/row $\times$ $(n-1)$ rows/table $\times$ n tables = $3(n^2-n)$ parameters.

7.3.2 Multipoint Topology

Multipoint tunnels behave like a data-link layer based on a multipoint transmission line. *Multipoint* must not be confused with *multicast*. When a frame is transmitted on a multipoint link, the sender specifies which of the nodes is to receive the frame. An Ethernet is a multipoint data-link implemented using multicast (broadcast). When a frame is transmitted on Ethernet, all nodes receive it, but only the one to whom it is addressed delivers it to its data-link provider interface. (Ethernet also supports multicast and broadcast, but its the unicast operation that we are concerned with.)

Multipoint is the most natural topology for tunnels based on a connection-less underlying network. If the underlying network service is connection-oriented, then the tunnel layer would have to simulate a multipoint topology. Simulating multipoint may seem like unnecessary complexity considering that the virtual network is capable of operating over point-to-point data-links, but doing so may significantly reduce routing overhead. The tunnel layer can manage routing more efficiently since it already knows the topology whereas the network layer must discover it by exchanging routing messages.

A multipoint tunnel maps each underlying network (see Figure 7.10) into a data-link service access point (interface). In an IP network, the network entity then maps each data-link interface to one or more network addresses. A subnet of n nodes interconnected by a multipoint tunnel requires a minimum of n DLSAPs (interfaces), 1 per node, and n virtual network addresses.

Configuring a multipoint virtual link requires a table having one row for each tunnel endpoint node in the subnet, each row having two parameters as follows:

1. the virtual network address of the node

2. the underlying network address of the node

This table is identical for every node in the subnet so the nodes may share a common configuration file. The configuration complexity is 2 parameters/row $\times$ n rows/table $\times$ 1 table = $2n$ parameters.

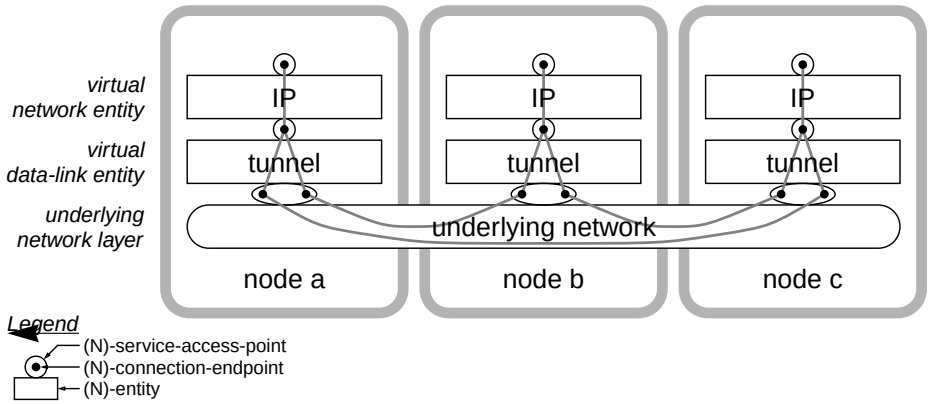

Figure 7.10. Example of Interface Binding for a Multipoint Tunnel

7.3.3 Comparison

Point-to-point topology, if the nodes are fully connected, requires a large number of interfaces ($n^2 - n$). Unless unnumbered interfaces are used, point-to-point topology will use an equally large number of network addresses. Using that many network addresses is usually unacceptable considering the increasing scarcity of IPv4 addresses. Increasing the number of interfaces generally means increasing routing overhead. In addition to these resource issues which impact performance, the additional configuration complexity may also impact reliability by increasing the probability of configuration errors. Use of the point-to- point topology in a design therefore precludes its use in large fully connected subnets. The definition of a *large* subnet depends on the available bandwidth and the definition of acceptable performance and reliability.

Table 7.2. Comparison of Virtual Data-link Topologies

	point-to-point	***multipoint***
number of data-link interfaces	$n^2 - n$	n
routing overhead (bits/second)	$(16/3)(n^4 - 2n^3 + n^2)$	$(16/3)(n^2 - n)$
configuration complexity	$3(n^2 - n)$	$2n$

The point-to-point topology, by explicitly specifying each underlying network connection in the configuration table, allows greater administrative control. The network administrator may omit selected underlying network connections from the table creating a non-uniform topology that may better suit policy. A side effect of constraining the topology is to reduce the resource requirements allowing larger subnets to be feasible using point-to-point tunnels. The constraints must be periodically reviewed and possibly adjusted to track changing policies and usage patterns creating a significant engineering burden. Contractual agreements, tariffs, cost and characteristics of the underlying network, government regulations, security policy and intra-company cost accounting are some examples of situations which may dictate constraining the subnet topology.

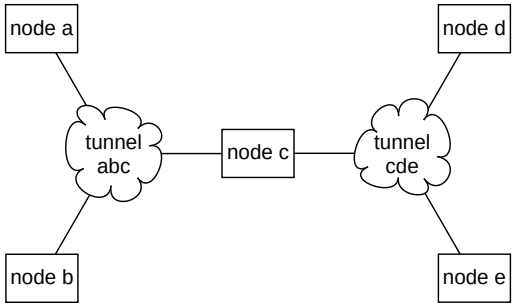

Figure 7.11. Node with Two Multipoint Tunnels

The multipoint topology requires fewer resources in large subnets. It is simpler to configure, and all nodes in a subnet may share a single configuration file. The multipoint topology does not inherently possess the configuration flexibility of the point-to-point topology that allows manual constraint of the subnet topology. That behavior might be recaptured in some cases by using more than one multipoint tunnel on the same underlying network. The effect is to bear the extra overhead of more tunnels only when it is necessary to achieve the desired constraints. Figures 7.11 and 7.12 show an example of one node with two multipoint tunnels.

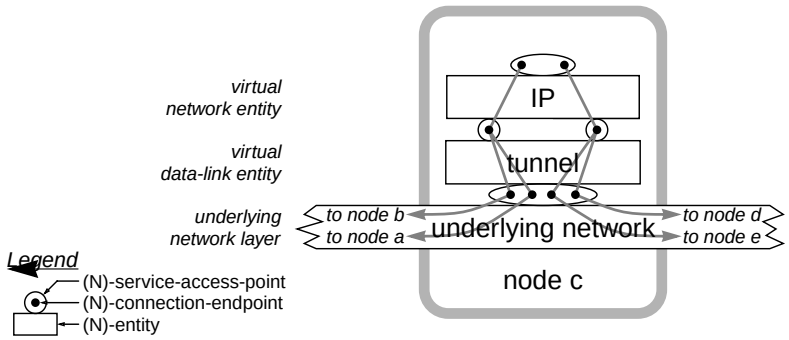

Figure 7.12. Example of Interface Binding for Node with Two Multipoint Tunnels