Planning an IP transition

Many enterprise administrators are so comfortable working with IPv4 addresses that they are hesitant to change. Network Address Translation (NAT) and CIDR have been excellent stopgaps to the depletion of the 32-bit IP address space for years, and many would like to see them continue as such. However, the IPv6 transition, long a specter on the horizon, is now approaching at frightening speed, and it is time for administrators not familiar with the new technologies to catch up.

The networking industry—and particularly the Internet—has made huge investments in IPv4 technologies; replacing them with IPv6 has been a gradual process. In fact, it is a gradual process that was supposed to have begun in earnest over 10 years ago. However, many administrators don't replace their IPv4 equipment unless it stops working. Unfortunately, the day when that equipment stops working is approaching rapidly. So, although it might not yet be time to embrace IPv6 exclusively, administrators should have the transition in mind as they design their networks and make their purchasing decisions.

NOTE **IPV4 ADDRESS EXHAUSTION**

The exhaustion of the IANA unallocated address pool occurred on January 31, 2011. One of the RIRs, the Asia Pacific Network Information Center (APNIC), was depleted on April 15, 2011, and the other RIRs are expected to follow.

Enterprise administrators can do as they wish within the enterprise itself. If all the network devices in the organization support IPv6, they can begin to use IPv6 addresses at any time. However, the Internet is still firmly based on IPv4, and will continue to be for several years. Therefore, the transition from IPv4 to IPv6 must be a gradual project that includes a period of support for both IP versions.

Now, and in the immediate future, administrators must work under the assumption that IPv6 traffic over an IPv4 connection. Eventually, the situation will be reversed. Most of the world will be running IPv6 and the remaining IPv4 technologies will have to transmit their older traffic over new links.

Using a dual IP stack

The simplest and most obvious method for transitioning from IPv4 to IPv6 is to run both. This is what all current versions of Windows do, going back as far as Windows Server 2008 and Windows Vista.

By default, these operating systems install both IP versions and use them simultaneously. Even if you had never heard of IPv6 until today, your computers are likely already using it and

have IPv6 link-local addresses that you can see by running the ipconfig /all command.
The network layer implementations in Windows are separate, so you configure them separately. For both IPv4 and IPv6, you can choose to configure the address and other settings manually or use autoconfiguration.
Because Windows supports both IP versions, the computers can communicate with TCP/IP resources running either IPv4 or IPv6. However, an enterprise network includes other devices, most notably routers, which might not yet support IPv6. The Internet is also almost completely based on IPv4.
Beginning immediately, administrators should make sure that any network layer equipment they purchase includes support for IPv6. Failure to do so will almost certainly cost them later.

Tunneling

Right now, there are many network services that are IPv4-only and comparatively few that require IPv6. Those IPv6 services are coming, however.
The DirectAccess remote networking feature in Windows Server 2012 R2 and Windows 8.1 is an example of an IPv6 technology and much of its complexity is due to the need to establish IPv6 connections over the IPv4 Internet.
The primary method for transmitting IPv6 traffic over an IPv4 network is called *tunneling*. Tunneling, in this case, is the process by which a system encapsulates an IPv6 datagram within an IPv4 packet, as shown in Figure 4-4. The system then transmits the IPv4 packet to its destination, with none of the intermediate systems aware of the packet's contents

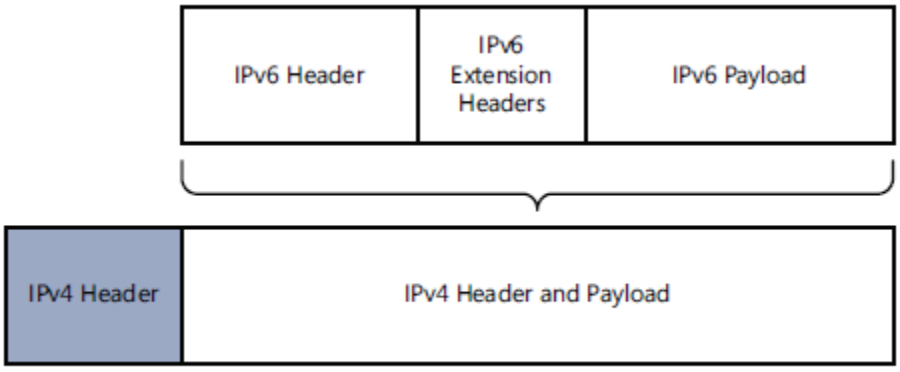

FIGURE 4-4 IPv6 traffic encapsulated inside an IPv4 datagram

Tunneling can work in a variety of configurations, depending on the network infrastructure, including router-to-router, host-to-host, router-to-host, and host-to-router. However, the most common configuration is router-to-router, as in the case of an IPv4-only connection between an IPv6 branch office and an IPv6 home office, as shown in Figure 4-5.

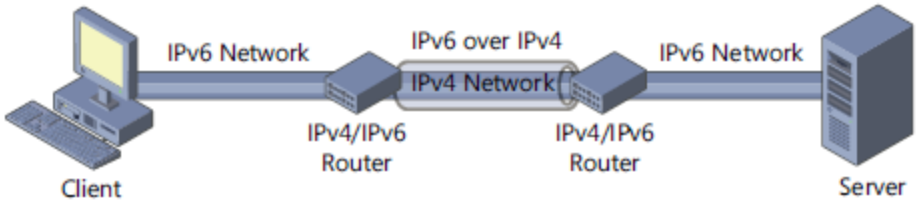

FIGURE 4-5 Two IPv6 networks connected by an IPv4 tunnel

The two routers support both IPv4 and IPv6 and the local networks at each site use IPv6. However, the link connecting the two sites is IPv4-only. By creating a tunnel between the routers in the two offices, they can exchange IPv6 traffic as needed by using their IPv4 interfaces. Computers at either site can send IPv6 traffic to the other site and the routers are responsible for encapsulating the IPv6 data in IPv4 packets for the trip through the tunnel.

Windows supports several different tunneling methods, both manual and automatic, as described in the following sections.

CONFIGURING TUNNELS MANUALLY

It is possible to manually create semipermanent tunnels that carry IPv6 traffic through an IPv4-only network. When a computer running Windows Server 2012 R2 or Windows 8.1 is functioning as one end of the tunnel, you can use the following command:

```
netsh interface ipv6 add v6v4tunnel “interface” localaddress remoteaddress
```

In this command, interface is a friendly name you want to assign to the tunnel you are creating; localaddress and remoteaddress are the IPv4 addresses forming the two ends of the tunnel. An example of an actual command would be as follows:

```
netsh interface ipv6 add v6v4tunnel “tunnel” 206.73.118.19 157.54.206.43
```

CONFIGURING TUNNELS AUTOMATICALLY

There are also a number of mechanisms that automatically create tunnels over IPv4 connections. These are technologies designed to be temporary solutions during the transition from IPv4 to IPv6. All of them include a mechanism for expressing an IPv4 address in the IPv6 format. The IPv4-to-IPv6 transition technologies that Windows supports are described in the following sections.

6TO4

The 6to4 mechanism essentially incorporates the IPv4 connections in a network into the IPv6 infrastructure by defining a method for expressing IPv4 addresses in IPv6 format and encapsulating IPv6 traffic into IPv4 packets.

ISATAP

Intra-Site Automatic Tunnel Addressing Protocol (ISATAP) is an automatic tunneling protocol used by the Windows workstation operating systems that emulates an IPv6 link by using an IPv4 network.

ISATAP also converts IPv4 addresses into the IPv6 link-layer address format, but it uses a different method than 6to4. ISATAP does not support multicasting, so it cannot locate routers in the usual manner by using the Neighbor Discovery protocol. Instead, the system compiles a potential routers list (PRL) by using DNS queries and sends Router Discovery messages to them on a regular basis by using Internet Control Message Protocol version 6 (ICMPv6).

TEREDO

To use 6to4 tunneling, both endpoints of the tunnel must have registered IPv4 addresses. However, on many networks, the system that would function as the endpoint is located behind a NAT router, and therefore has an unregistered address. In such a case, the only registered address available is assigned to the NAT router itself, and unless the router supports 6to4 (which many don't), it is impossible to establish the tunnel.

Teredo is a mechanism that addresses this shortcoming by enabling devices behind non-IPv6 NAT routers to function as tunnel endpoints. To do this, Teredo encapsulates IPv6 packets within transport-layer User Datagram Protocol (UDP) datagrams rather than networklayer

IPv4 datagrams, as 6to4 does.

For a Teredo client to function as a tunnel endpoint, it must have access to a Teredo server, with which it exchanges Router Solicitation messages and Router Advertisement messages to determine whether the client is located behind a NAT router.

To initiate communications, a Teredo client exchanges null packets called *bubbles* with the desired destination, using the Teredo servers at each end as intermediaries. The function of the bubble messages is to create mappings for both computers in each other's NAT routers.