Chapter 2

Introduction to IPv6 Architecture

Solutions in this chapter:

- **Understanding the Benefits of IPv6**
- **Comparing IPv6 to IPv4**
- **Examining IPv6 Network Architecture**
- **Upper-Layer Protocol Issues**
- **Understanding ICMPv6**
- **Understanding Neighbor Discovery**

☑ **Summary**

☑ **Solutions Fast Track**

☑ **Frequently Asked Questions**

Introduction

Version 6 of the Internet Protocol (IPv6) has finally arrived in practical form! Although the protocol has been worked on for close to ten years now, official specifications and standards have only recently been finalized, and there are still some aspects of the protocol that are being worked on by the Internet Engineering Task Force (IETF) working groups. The issues that made version 4 of the Internet Protocol (IPv4) inadequate required complex solutions. This has forced designers of the new protocol to work diligently to ensure that the same issues would not be encountered with the new version of the protocol. Members of the Internet community who were responsible for developing the protocol carefully scrutinized each new Request for Comments (RFC) that was developed. For those not familiar with the RFC process, RFCs are documents that detail the protocol specifications so that hardware and software manufacturers will know how to implement the protocol in a standard and agreed-upon manner. Standardization enables each manufacturer and software vendor to follow the same blueprint rather than developing proprietary versions of the protocol. (This was a common problem with earlier networking protocols.)

The realization of a new, scalable protocol in which considerable thought has been given to future expansion is a very exciting concept. Never before has the Internet community seen such a magnitude of effort or planning put into the development of a new protocol, and there has certainly never been a protocol more specifically tailored to the growth of the Internet. The purpose of this book is to help ensure a smooth implementation of the new protocol by focusing on the configuration of IP version 6 on the Cisco IOS, the most common platform that will utilize the new protocol.

Understanding the Benefits of IPv6

Let's discuss some of the benefits of IPv6 in more detail in order to see how this protocol attempts to deal with the Internet and business network problems of today. Some of the main reasons for the development of the new version of the Internet Protocol were the exhaustion of the Class B address space, the growth of the backbone routing table, security issues, IP options size limitation, and routing performance. We'll look at the two main problems solved by IPv6—namely address depletion and routing scalability—in more detail, and then look at some of the added benefits that IPv6 gives to network designers and administrators. The benefits of IPv6 include:

- Increased IP Address Size
- Increased Addressing Hierarchy Support
- Simplified Host Addressing (unified addressing: global, site, local)
- Simplified Autoconfiguration of Addresses (easier readdressing, DHCPv6, Neighbor Discovery instead of ARP broadcasts)
- Improved Scalability of Multicast Routing
- The *Anycast* Address
- Streamlined Header
- Improved Security (security extension headers, integrated data integrity)
- Better Mobility (home agent, *care-of* address, routing extension header)
- Better Performance (aggregation, Neighbor Discovery instead of ARP broadcasts, no fragmentation, no header checksum, flow, priority, integrated QoS)

Increased IP Address Size

IPv6 has 128 bits available for addressing. Let's examine this briefly in order to appreciate the vastness of this degree of address space. 128 bits of address space means that there are 2^{128} different addresses available. The first three bits of 001 are reserved for Globally Routable Unicast addresses. This means that we have 125 bits left to use for addresses (128–3=125), so we have 2^{125} addresses available before Globally Routable Unicast address space is depleted. This is roughly 4.25 E+037 addresses. To put this into perspective, let's compare this to IPv4. In IPv4, we use all address space between 0.0.0.0 and 223.255.255.255 for unicast routing, which is approximately 3.7 E+09 addresses (we will not take into account the addresses delegated as non-routable reserved addresses as defined by RFC 1918). This means that IPv6 has room for 10^{28} more addresses than IPv4!

Clearly, 128 bits provides enough address space to take current Internet trends well into the future. In fact, it may seem like an inexhaustible amount of address space, but we will see when we get into the details of LAN and WAN configuration that this is not quite the case. Still, it is a much larger address space than IPv4. One thing to think of in order to appreciate IPv6 is the number of *networks* that IPv6 can support. An IPv6 address uses the last 64 bits to describe the host ID for a system on a network. In addition, IPv6 actually uses the last 64 bits of the address to distinguish hosts from one another on the same subnet. Whether

using the link-local, site-local, or Globally Routable Unicast address format, the last 64 bits on a machine will remain the same. This is because IPv6 uses the Layer 2 Media Access Control (MAC) address as the host ID for a machine (the Layer 2 MAC address is the address that is burned into all Layer 2 hardware, such as Ethernet cards and other Network Interface Cards). Since MAC addresses are only 48 bits long, each one is padded with a 16-bit prefix, which will be discussed in more detail below. This limits the number of addresses that can be used because there will rarely be 2^{64} addresses in use on a typical Ethernet LAN. Some address space definitely gets wasted. However, if you remove the 64 bits used for host ID and the first three bits used to designate Globally Routable Unicast addresses, you get 2^{61} possible addresses (2.31E+018). So even without using all of the addresses that IPv6 has available, we have the scaling ability to take us well beyond the future of IPv4. Clearly, IPv6 frees up our ability to use addressing efficiently without having to worry about running out of addresses. Please keep in mind that I do not mean to say that address space should be used in a carefree manner; that is how IPv4 came into its current predicament.

Table 2.1 presents a rough comparison of the address sizes of IPv4 and IPv6.

Table 2.1 Address Space Comparison

Specification	IPv4	IPv6
Address Length	32 bits	128 bits
Host Identifier Length	2 - 24 bits	64 bits
Network Identifier Length	7 - 30 bits	61 bits
Maximum Number of Hosts per Subnet	$2^2 – 2^{24}$	2^{64}
Maximum Number of Subnets	$2^7 – 2^{30}$	2^{61}
Maximum Number of Hosts	3.7 E+09	4.25 E+037

Increased Addressing Hierarchy Support

As we learned earlier in the chapter, IPv6 addressing has restructured the means by which address blocks are delegated. IPv4 first used the classful IP assignment rules, then began to assign based on the principles of Classless Inter-Domain Routing (CIDR). IPv6 corrects the de-aggregation problems associated with each of these by splitting the IPv6 address into a set of definite scopes, or boundaries, by which IPv6 addresses are delegated.

The Format Prefix is used to show that an address is Globally Routable Unicast, or another type of address, and is always set to the same value. This allows a routing system to quickly discern whether or not a packet is Globally Routable Unicast or some other type. By obtaining this information quickly, the routing device can more efficiently pass the packet off to routing subsystems for proper handling.

The Top Level Aggregator (TLA) ID is used for two purposes. First, it is used to designate a large block of addresses from which smaller blocks of addresses are carved in order to give downstream connectivity to those who need access to the Internet. Second, it is used to distinguish where a route has come from. If large blocks of address space are given only to Internet service providers, and then in turn delegated to customers, it becomes easier to see which transit network a route has traversed, or from which transit network the route first originated. With IPv4 many addresses were portable, and the numbers authorities were delegating blocks down to small businesses, so it became impossible to know where a route came from without tracing back towards the source of the packet. Now, with IPv6, determining the source of a route is more feasible. Imagine an Internet consisting of 500 Tier 1 providers. If this were the case (and it is most likely not too far off from today, though what makes a provider a Tier 1 provider is very ambiguous), then at the very least a quick search through a text file could tell you where a route originated, based on the TLA ID of the longest match route. It's even possible to obtain software that has this functionality built into it (if the software is properly written to keep a dynamic list as new delegations are assigned).

Let us look at the size of the TLA in more detail. We discussed in prior sections how the address space would be given only to providers and those who needed their own IPv6 space. (The term *needed* is used cautiously here, as it is ambiguous as well—there are currently no set boundaries or requirements with respect to what *need* means in this context.) This way, we are able to sufficiently aggregate prefixes into big blocks at the Internet core, thereby passing fewer routes between routing domains, as well as internally, which will increase the efficiency of the Internet core.

For fun, let us assume that we have the IPv6 address delegation 3D00::B234::/24. Let us further assume that all of our customers have sufficient need for a /48 delegation for their networks. This leaves us with 24 bits of addressing to delegate out—that's a lot of address space! In fact, the number of networks we can support with this scheme is equivalent to the number of *hosts* we could support with a Class A IPv4 address block! You can see that there will be much more

pressure put on Tier 1 service providers to efficiently track the delegation of address space that they make. Currently, a Tier 1 service provider gets addresses in blocks of perhaps /16 or less. If we assume that a service provider today only delegates addresses up to /24, that leaves only 256 delegations (eight bits) that the service provider can make prior to applying for more address space. Most Tier 1 service providers today are required to subnet delegations down to at least /28 in order to qualify for more address space, so this example may not be entirely realistic, but we can still grasp the size of a TLA, which is monumental compared to the size of the assignments that are given today.

NOTE

This amount of address space can present tremendous support infrastructure problems. Service providers that today provide DNS service for their downstream customers will have to think long and hard about the best way to implement these support structures before getting into the IPv6 market.

As we can see from the previous example, Tier 1 service providers will have extremely large address spaces to deal with. This will not only eliminate most of the politics surrounding address delegation and obtaining more address blocks, but will also provide motivation for major support and automation of infrastructure upgrades within an organization. Many service providers today have difficulty upgrading support structure due to the engrained functionality and interdependencies of many support platforms integrated together. IPv6 provides for great challenges and opportunities, not only in network engineering and architecture, but also in IT development and integration. The trick will be making the move from the old world to the new look like a fresh start, rather than a workaround for support.

The Next Level Aggregator (NLA) address block is a block of addresses that are assigned downstream out of a TLA block. We know that these addresses are to be aggregated as much as possible into bigger TLA blocks, when they are exchanged between providers, in the Internet core. Let us look at the benefits of this type of addressing structure from the NLA perspective.

There are two main advantages to getting address space from a provider. The first has to do with individual backbone routing stability. If we are an NLA and wish to provide downstream service to our customers, we will most likely wish

to provide the fullest, most robust service we can to our client base in order to retain current clients and to gain market share. Perhaps we wish to allow customers to connect to us at multiple locations, as we are fairly geographically diverse for a given region, and have rich connectivity upstream to Internet Tier 1 (core) providers. Furthermore, we want to allow our customers to receive a full routing table, should they desire one, if they want to use explicit routes to form their routing policy. Perhaps they wish to load-balance between two connections, using some destinations preferred through one connection, and the rest preferred through the other connection to us. To do this, we have to carry full routes in our backbone so that we can pass them down to our customers. Though an Internet core is usually composed of very modern, robust routing equipment, a Tier 2 provider may not be able to afford to constantly upgrade their backbone in order to keep up with new technology and increased routing table size. Luckily, with IPv6, processing power is not as big a worry as it could be. Because the Internet core is fundamentally aggregated efficiently, we now have a much smaller routing table to maintain. We can provide full routes to a customer, and that set of routes may not be too big for us to handle. So by everyone "playing nice" and following aggregation strategies, we are able to reap the benefits of the core's minimized routing table size in our own network backbone.

The second benefit to NLA aggregation has to do with actual route stability of our routes across the Internet core globally. Some background information is needed in order to fully appreciate this point. At the beginning of the Internet's explosion in size, there were times when the Internet was not very stable. BGP speakers would lose routes due to backbone links failing, immature software and the like. Because of this, routes were constantly being advertised and then withdrawn (when the route became unreachable), causing considerably more processing to take place on core routers, which must keep an up-to-date set of full Internet routes at all times. To combat this BGP instability, the concept of *route dampening* emerged. Essentially, route dampening works as follows: every time a route is withdrawn and re-advertised, it is assigned a penalty, which is kept track of at the place of instability (usually an exterior border gateway protocol or eBGP session). The more the route "flaps," the higher the penalty associated with that route. When the penalty associated with the route reaches a certain level, the route is withdrawn and not accepted for advertisement for a given period of time. When this happens, the route is said to be *dampened*. The dampened route must undergo some period of wait time, without flapping more (or the penalty gets even higher) before it can be re-introduced into a router's BGP table. When the route goes long enough without flapping (the penalty decreases with time),

the route is again allowed, and it is inserted back into the router's BGP table and treated just like other routes. Route dampening provided a way for the Internet core to deal with instabilities in a manner that minimized the cost of other crucial processing. Border Gateway Protocol (BGP) version 6 and IPv6 routing will be discussed in detail in a later chapter to better explain this concept.

Configuring & Implementing…

Site Renumbering

Site renumbering is performed when there is a need to replace or redistribute existing addresses. An extreme example of site renumbering can occur when a new service provider is chosen and new global addresses need to be issued. Site renumbering in IPv4 can pose an administrative nightmare, but IPv6 simplifies this process immensely. Here are two ways to implement the site renumbering process:

- If stateful autoconfiguration is in use, the Top Level Aggregator and Next Level Aggregator numbers of the new service provider can be placed in the DHCPv6 service and propagated throughout the site.
- If stateless autoconfiguration is in use, the new network prefixes can be placed in the routers and propagated throughout the site.

The advantage of the IPv6 architecture is that the interface identifier remains the same; only the prefixes are changed. In the case of IPv6 stateful autoconfiguration, the prefixes can be propagated from a central point.

Now that we understand route dampening, we can appreciate the second benefit of aggregation. When our upstream provider aggregates this route for us and only announces the aggregate to their peers, this aggregate will, in all likelihood, remain stable, without respect to the stability of our own network. Because of this, we are virtually certain that another provider will never dampen our routes somewhere else on the Internet. None of the more specific routes that we use need be spread across the Internet core, outside of our own upstream provider. This improved routing stability is a major benefit to aggregation as a

whole, both in IPv4 and in IPv6. The good part is that it is *required* in IPv6—in IPv4 it is only recommended. So now the only place that we need to worry about being dampened is within our own upstream provider's network. Fortunately, because we are paying for our upstream connectivity in most cases, it is substantially easier to get our own upstream provider to help us remove the penalties on dampened routes than it would be with another provider to whom we had no financial obligation. As you can see, in addition to more addresses and smaller routing table sizes, there are more ramifications of IPv6 aggregation schemes than first meet the eye.

The Site Level Aggregator (SLA) enjoys most of the benefits that an NLA does, except for its size: the SLA is usually a network or network provider with a much smaller network. Because of this, a smaller delegation of address space is needed. It retains the values of aggregations in that its routing tables are kept smaller, even when receiving a full Internet routing table from its upstream provider. It also enjoys the benefits of global route stability, in that its upstream provider, whether an NLA or a TLA, aggregates according to the principles of the IPv6 aggregations model.

Simplified Host Addressing

As we studied earlier, the IPv6 model defines 128 bits of address space. The first 64 bits are used for network numbering, and the last 64 bits are used for host numbering. We also remember that the last 64 bits of the host ID are obtained from the MAC address of the host's Network Interface. You may wonder how the 64-bit address is derived from a MAC address, which is classically only 48 bits. In this section we will look into how addresses are derived, and what developments we may see in the future as a result of IPv6's addressing scheme.

By convention, when assigning a host in IPv4, IPv4 will break up the given subnet and assign host addresses based upon the addresses that are available. Also by convention, the first address is normally given to the designated router, and the rest of the addresses get assigned to hosts on that subnet, with the last address in the subnet reserved for the broadcast addresses of that subnet. In IPv6, the situation is somewhat different. With IPv6, we know that the host ID is a 64-bit address that is obtained from the MAC address. Although the MAC addresses of today are typically 48 bits, we need a way to get the host ID to come out to 64 bits. The solution to this problem is to pad the MAC address with some well-defined set of bits that will be known by routing systems on that subnet. Today, we use the strings **0XFF** and **0xFE** (:FF:FE: in IPv6 terms) to pad the MAC address between the company ID and the vendor-supplied ID of the MAC

address. (MAC addresses are delegated in much the same way as IP addresses today, except that companies who make network interface (NIC) cards are given a piece of MAC space, rather than providers being given IPv4 space.) This way, every host will have a 64-bit host ID that is related to their MAC address in the same way. Furthermore, we know that the 64-bit MAC address will be unique on a given network, because every NIC card will have a unique MAC address. This well-defined padding makes it possible to learn the IPv6 addresses (or at least the host IDs) of other machines on the subnet simply by learning the Layer 2 MAC information.

One interesting debate is whether or not MAC addresses will need to become 64 bits in length prior to the widespread deployment of IPv6. If there is a need for MAC addresses to become longer (if all MAC addresses are used), then 64 bits will most likely be the next option for length, as this will supply over 1.8E019 more MAC addresses to use (2^{64}–2^{48}). Moreover, if this comes to be the case, we may simply stop the padding of the MAC address, and use the full 64 bits of the MAC address for the Host ID.

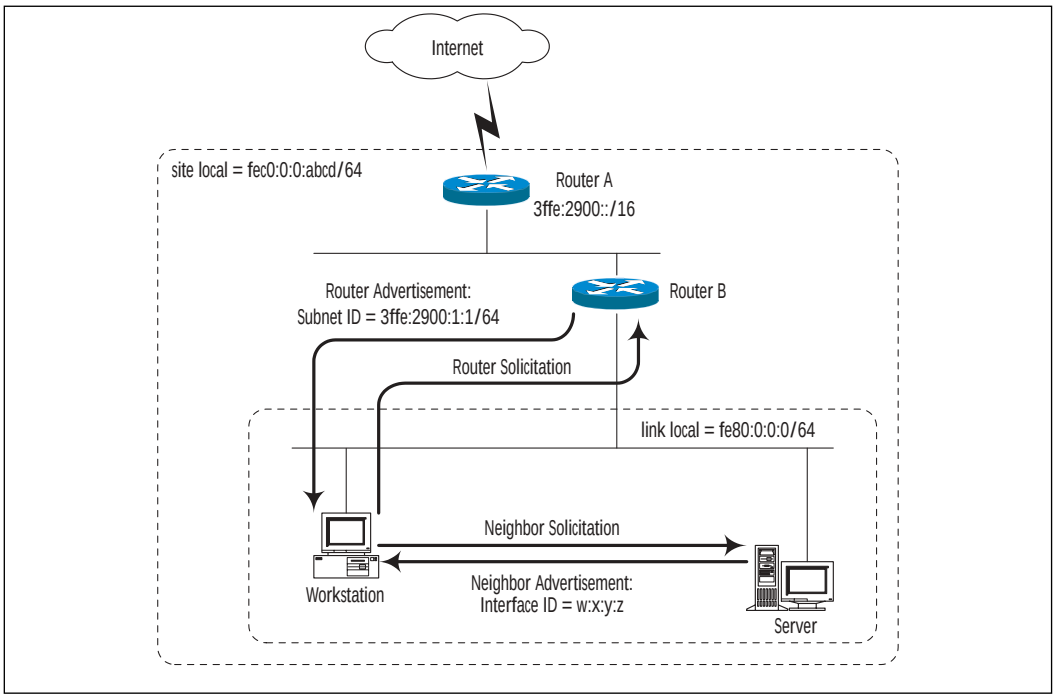
Simpler Autoconfiguration of Addresses

One of IPv6's best perks for administrators is that not only is the Host ID determined prior to configuring an IPv6 speaking machine, but the network on which it resides can be deduced as well, making it possible for autoconfiguration to take place. Before we go into detail about autoconfiguration, a new type of address will have to be brought into our repertoire: the *multicast* address.

A multicast address can be simultaneously assigned to more than one machine. It differs from an anycast address in that anycast packets are routed to the *closest* destination (one of the set of machines with the same address), while multicast packets are routed to *all* machines that are assigned that address. This is fundamentally different than a Globally Routable Unicast address in that more than one host can be numbered with the same address, so the address that a given host is assigned need not necessarily be unique for the scope on which the multicast address is acting. All machines assigned this multicast address are said to be in a *multicast group* whose address is the multicast address they use. Multicast-speaking machines send and receive data from more than one host (every member in that group). This type of addressing and routing has typically been used for 1-to-N or M-to-N type Internet transactions (when one or more people need to get identical information to more than one destination). Multicast provides an efficient means by which to do so.

If we unite the concept of multicast with the concept of the Host ID coming from the hardware on a given machine, we can see how autoconfiguration is possible. When a machine first powers up onto a network and realizes that it is connected and is supposed to speak IPv6, it will send a multicast packet that is well known and has a standard definition out onto the LAN segment to which it is attached. This packet will be destined towards a locally scoped *(see earlier)* multicast address, known as the Solicited Node Multicast address. When the router sees this packet come in, it can reply with the network address from which the machine should be numbered in the payload of the reply packet. The machine receives the packet and, in turn, reads the network number that the router has sent. It then assigns itself an IPv6 address by appending its Host ID (obtained from its MAC address of the interface that it connected to that subnet) to that network number. See Figure 2.1 for a graphical representation of autoconfiguration. Not only does this require no manual intervention by the administrator to configure the machine (though it may or may not involve manual configuration of the router on that subnet), it also ensures that the address is unique. The machine is guaranteed to have a unique address because the network number is assigned uniquely by the router on that network, and the Host ID is unique because the MAC address of the interface by which that machine is attached is provided by the vendor and unique. Furthermore, now that it has a routable address, it can learn the default route that it needs in order to get off of that subnet. Notice the ease of configuration that we now have when we move from one network to another. Not only do we no longer have to manually reconfigure an end-station (and then reboot in most cases), we also no longer have to take time out of our network administrator's busy day for him to delegate an address in order to ensure its uniqueness. Also, the administrator no longer has to keep track of the addresses that he has assigned and which ones are free at any given time! Certainly, this can save a network administrator a great deal of time, not just in the paperwork associated with keeping track of addresses used, but in reconfiguration that must occur in order for a network to be renumbered. Think of the things an administrator could be doing if he isn't constantly being hounded for IP addresses or network numbers! We will go into more detail about the concept of the multicast address and its possibilities in a later section.

Figure 2.1 Autoconfiguration Mechanism



Improved Scalability of Multicast Routing

Now that we have studied unicast addressing, looked at the primary advantages of multicast addressing, and discovered the potential of routing table size scalability in the Internet with IPv6, let's take a moment to discuss the multicast address in a little more detail. Multicast servers are perhaps the most misunderstood technology of the present day. We'll start by examining the concept of multicasting in general.

In the beginning, the Internet was primarily a research network upon which research data flowed between one university and another. This was not big business, so congestion problems were tolerated, and the data that people were sending was not dated because it didn't need to be received in real-time. Today, by contrast, businesses and consumers are using the Internet for a vast array of applications. More and more, we are seeing different types of media going over the Internet, whether it's stock quotes, phone calls, or even our favorite TV channel. We see the need for media to arrive quickly and to be sent to an increased audience. Even things like newsgroups are getting information out to millions of people each day. This 1-to-N transmission trend is bringing about a need for a new type of traffic sending, in which one person can send a piece of

data to many people. In the past, if we wanted to send a piece of data to ten friends, we would simply make ten copies of that data and send them to each person one at a time. However, as this type of transmission gains popularity, a scaling problem takes place. For instance, say we have a video or radio show that we wish to send out over the Internet. If we want to send this media to 10,000 people, all of whom want to see or hear the show in a fashion as close to real-time as possible, we have to make sure that our upstream bandwidth is sufficient to handle up to 10,000 times the data rate of one transmission. That requires us to spend much more money purchasing the upstream bandwidth to satisfy our client base (our viewers or listeners). Fortunately, the concept of multicast was conceived some time ago, and has been in a testing phase for quite a while. The idea of taking one piece of data and efficiently sending it to many interested parties at once becomes a complex routing problem, especially if we become caught in the unicast paradigm we are all used to. The concept of multicast addresses this problem.

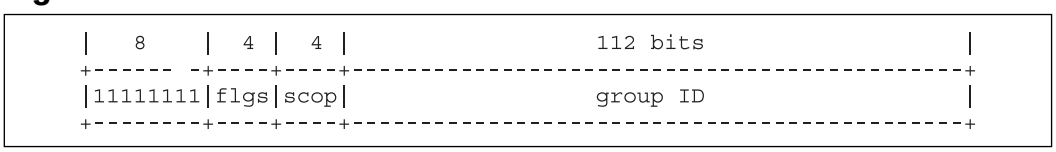
In a multicast situation, we have a 1-to-N (or M-to-N) relationship between the source and the destination. Instead of using a unicast address to indicate that we are interested in receiving a given multicast feed, we use a multicast address. In IPv4, a multicast address is usually referred to as a *group address*. This group address, when applied to a machine or to an application on that machine, signifies that we are interested in listening to any data that is sent to that address. In IPv4, the address range of 224.0.0.0 to 239.255.255.255 is used to designate multicast group addresses. When someone wants to receive multicast feeds, they (temporarily or permanently, depending on the situation) assign themselves that address, and effectively listen for packets coming along that have that multicast address listed as a destination.

The routing for multicast becomes rather complex and is beyond the scope of this book, but you are encouraged to read more about it. Good information can be found either on the Multicast Forum home page at www.ipmulticast.com, or in the IETF working groups regarding multicast. Some of the IETF working groups you may wish to check out include the Mbone Deployment working group (www.ietf.org/html.charters/mboned-charter.html) and the Inter-Domain Multicast Routing working group (www.ietf.org/html.charters/idmr-charter.html). There are also a number of protocol-specific working groups currently active within the IETF, including the Multicast Source Discovery Protocol working group (MSDP) and the Protocol Independent Multicast working group (PIM). I leave it up to you to learn as much as you wish about the current updates to multicast routing in general. For the purposes of this book, let us assume that multicast

works, and that it will help save us bandwidth (since it allows us to send only one stream out of our Internet connection, and the backbone will reproduce it as seen fit to make sure it gets to all the interested destinations).

One particularly good application of multicast is in a corporate network. Perhaps a memo needs to be sent to all employees' workstations at once, or a live videoconference from the CEO needs to be sent over the corporate network to all employees. In a corporate network, we want to save as much money as possible on bandwidth while maintaining an efficient routing structure. Multicast buys us just that. However, in most cases, we only want multicast information (streams) to get to the places that are supposed to see them. We do not want the whole Internet to hear our CEO talk about our newest secret initiative to take over our competition! For this, IPv6 has the concept of multicast scoping built into it. With IPv6, we can designate certain multicast streams to be routed only within a certain area, and never to allow packets to get out of that area for fear of who may see them. This scoping will be well known and understood by all routing entities in order to ensure, through minimal configuration, that multicast data and multicast routes do not get outside the edges of the routing domain for which they are meant to exist. Figure 2.2 presents the multicast addressing format in a little more detail.

Figure 2.2 IPv6 Multicast Address Format



As we can see, the multicast addressing architecture is a little different than that of the Globally Routable Unicast addressing format. Notice that the first eight bits are all set to one, which will allow a routing device to know immediately that the packet is multicast in nature, and subject to the special handling associated with this packet type. The next four bits are used for flags. Currently, the first three bits in the *flags* field are reserved and undefined, so they should always be set to zero. (You will find that some implementers of protocols will use these bits fallaciously for some sort of proprietary signaling. This is fine until the bits get standardized to something in the future, at which point incompatibilities will arise.) The fourth bit is known as the *T bit* (see RFC 2373), and is used to decide whether the multicast address is a permanently assigned address (also called *well-known*) or a temporary assignment (also known as *transient*). So this

field will tell us if the multicast address being used is one that is standard (perhaps a group address used to contact all nodes within a given routing domain, for example) or a temporarily assigned address (perhaps the Monday night football game broadcast over the Internet). The next field is the one we are interested in here: the *scope* field will determine how far the multicast packet can go, in what areas of a routing domain the packet can travel, and the group address that can be advertised. The *scope* field takes the values shown in Table 2.2.

Table 2.2 Scope Definitions

Scope Field Value	Definition
0	Reserved
1	Node-local scope
2	Link-local scope
3	(unassigned)
4	(unassigned)
5	Site-local scope
6	(unassigned)
7	(unassigned)
8	Organization-local scope
9	(unassigned)
A	(unassigned)
B	(unassigned)
C	(unassigned)
D	(unassigned)
E	Global scope
F	Reserved

Depending on how we assign our multicast address, we can control how far the multicast packets will travel, and how widely the routing announcements associated with that multicast group will be advertised. For instance, if you would like to advertise a multicast session of the fish tank in your office, and you would like the whole world to see it, you would assign a scope of E (1110 in binary). However, if you want to set up a multicast group so that you and your coworkers can have a video conference over the corporate network, you would want to make sure to give the address a scope of 5 (0101 in binary), or 2 (0010) if

everyone involved is on the same LAN as you. This makes life a little easier in terms of controlling how far information gets propagated. Now, instead of relying on a Network Administrator to apply filters at the borders of each routing domain, we can rely on software (which is generally not susceptible to the same sort of random changes that networks are) to keep our traffic within the scope we want. This allows for privacy at a level that is much easier to implement. This is another benefit of IPv6: Not only are multicast boundaries well defined, they are also easy to maintain.

The Anycast Address

IPv6 defines a new type of address, known as the *anycast* address. Although this form of address is deployed in a limited fashion in IPv4, IPv6 integrates this address type into its operations, which improves routing efficiency. In this section we will look into some of the characteristics of the anycast address in detail and discuss some of the interesting applications of the anycast address in the IPv6 Internet of the future.

An anycast address is an IPv6 address that is assigned to a group of one or more hosts, all of which serve a common purpose or function. When packets are sent to the IPv6 anycast address, routing will dictate which member of the group receives the packet via the machine closest to the source, as determined by the IGP of the network in question. (IGP is the Interior Gateway Protocol: the routing protocol you use in your routing domain; for example, RIP, EIGRP, or IS-IS,) In this way, it becomes possible to disperse functionality geographically across your network in a way that improves efficiency in two ways. This differs fundamentally from the multicast address. Although both the anycast and the multicast addresses are assigned to more than one host, the anycast address serves for data transmissions that are 1-to-1, whereas multicast addressing is used when a data transmission to multiple destinations is required. Let us look at the two primary benefits of the anycast addressing scheme.

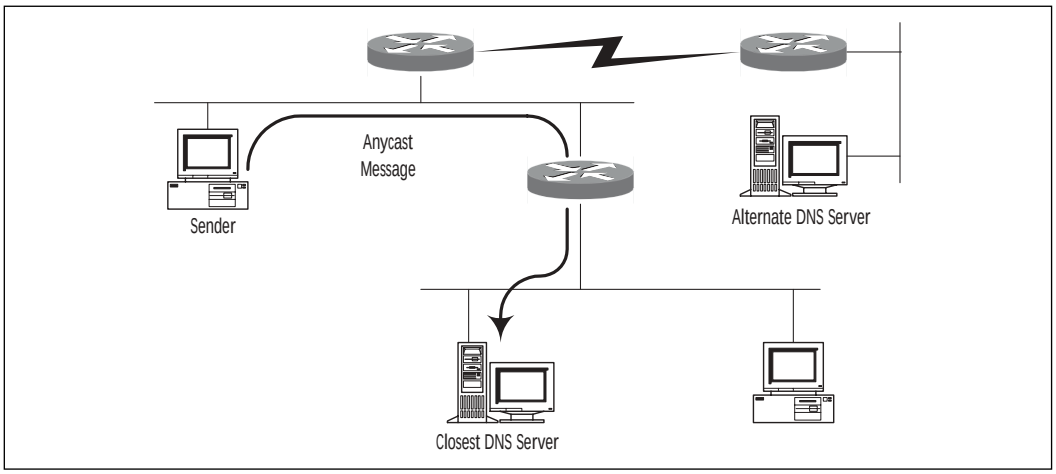
First, if you are going to the closest machine in a group and it does not matter which group member you exchange information with, you will usually save time by communicating with the closest (IGP-wise) group member. Second, communicating with the closest anycast group member saves bandwidth, because the distance a packet has to travel is, in most cases, minimized. So not only can anycast save you time, but it can also save you money (by using less bandwidth).

The anycast address does not have its own set of bits to define it; instead, anycast addressing is derived from either scoped or Globally Routable Unicast

addresses. From the point of an IPv6-speaking machine, the anycast address is no different than a unicast address. The only difference is that there may be other machines that are also numbered with the same scope of unicast address within the same region for which that scope is defined (for instance, you may have more than one machine with a site-local anycast address within a given site).

Now that we understand the differences between anycast and multicast addresses, let's look into some possible uses of the anycast address. One application that anycast can help with is DNS (Domain Name Service). If we were to offer DNS to many people or customers, as in the case of most Tier 1 service providers today, we would need to build our DNS in a way that could handle a large number of queries from all parties for which we provide the service. Because of this, it is often more efficient to deploy multiple DNS servers and spread them out geographically. This will allow for fail-over if one DNS server becomes unreachable due to network failures, and it will also allow us to distribute the load of our DNS service between these servers. However, we do not want to make our customers assign too many different IP addresses for DNS servers to point to for resolution, as most people only use one or two. Also, we want some way for one or two IP addresses to be used for all of our geographically diverse service, for the fail-over reason just stated. One way to do this would be to assign each DNS server that has identical configuration and authoritative information the *same* IP address. If we then inject routes to each of these DNS servers into our backbone routing table, when someone queries our DNS the request will be sent to the DNS server that is geographically closest. This will allow us to split up the load among multiple DNS servers and to avoid too much backhauling of DNS queries across our backbone. So by this method of deployment, we are saving both time for our customers (DNS servers are close, so their data transmission takes less time), and money for ourselves (bandwidth = money for service providers). Because DNS is User Datagram Protocol (UDP)-based rather than TCP-based, transactions between DNS servers and end-stations are quick and short, and don't need to be kept track of with sequencing, error checking, etc. When we want to resolve a host name, a packet is sent to the DNS server requesting the address associated with a given Internet Domain Name and a response is sent back with the answer. This makes the anycast addressing model viable for this type of application. An illustration of anycast in use is provided in Figure 2.3. You can read more on this specific type of deployment at www.globecom.net/ietf/draft/draft-catalone-rockell-hadns.00.txt.

Figure 2.3 Anycast Message



Designing & Planning…

Which Applications Are Good Candidates for Anycast?

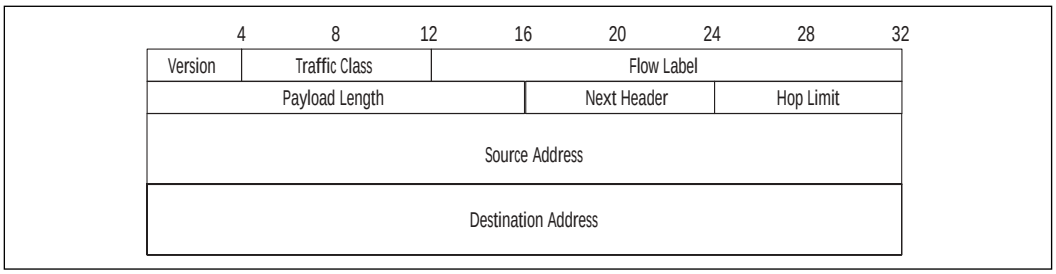
Some applications may not be well suited to anycast deployment. For instance, TCP-based applications using anycast addressing for deployment will not provide the fail-over capabilities that the previous example provides. When we are using a UDP-based application, there is no sequencing information to keep track of. With TCP, we run into a problem: when a network problem occurs and users are in the middle of a TCP session with an anycast machine, the TCP sequencing will be all wrong when the traffic gets rerouted to the next-closest anycast server. In the case of Web traffic, which is largely TCP-based, the user would need to reload the Web page (at least), to get to where he or she was when the failure occurred on the network. Other applications could have even more disastrous consequences associated with rerouting, so careful consideration should be given to which types of services are supplied using an anycast model.

Streamlined Header

The new IPv6 header is simpler and more streamlined than the IPv4 header. The new header has only six fields and two addresses, while an IPv4 header contains

ten fixed fields, two addresses, and a variable-length *options* field. Figure 2.4 illustrates the format of an IPv6 header.

Figure 2.4 IPv6 Header

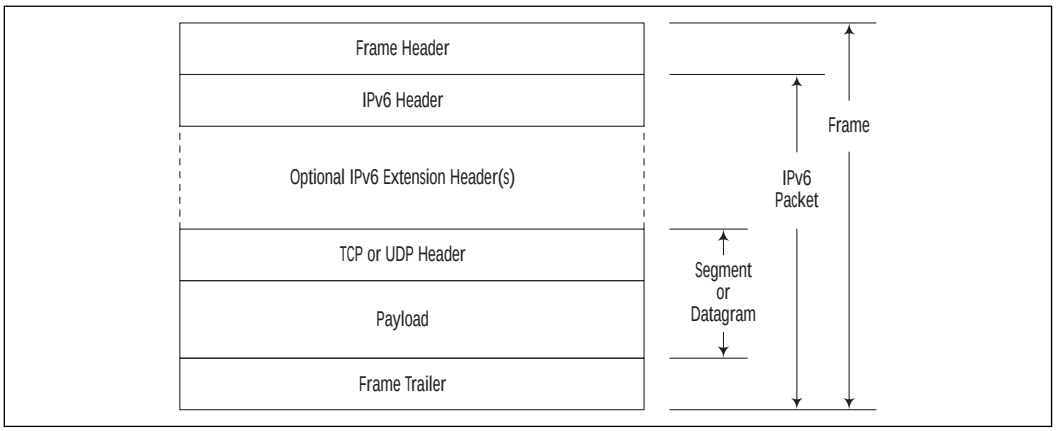


The IPv6 header provides the following simplifications:

- **Simplified Format** The IPv6 header has a fixed format with fewer fields. The variable-length *options* field has been eliminated. Other IPv4 fields have been eliminated or changed to optional extension headers. Not all of the optional extension headers need to be processed by each node—many of the extension headers are intended only for processing by host nodes. This simplified format reduces the protocol overhead of IPv6 and allows for greater flexibility.
- **No Header Checksum** The IPv4 *checksum* field has been eliminated. This field was included in IPv4 because early networks used slow and unreliable links, thus computing the checksum at each hop was necessary for ensuring data integrity. Today's network links are fast and highly reliable, so only hosts need to compute the checksum, not routers.
- **No Hop-by-Hop Fragmentation Procedure** In IPv4, routers fragmented packets that were too large for transmission on the outgoing interface. This added significantly to IPv4's processing overhead. In IPv6, only the host may fragment a packet. To aid the host, IPv6 includes a function that finds the maximum transmission unit (MTU) size from source to destination.

Figure 2.5 illustrates a transmission frame of a TCP segment (or UDP datagram) using IPv6.

Figure 2.5 Transmission Frame with IPv6



Security

One goal of IPv6 designers was to support interoperable encryption-based security. IPv6 integrates security into its architecture by introducing two optional extension headers: the Authentication Header (AH) and the Encrypted Security Payload (ESP) header. These two headers can be used together or separately to support many types of security functions.

- **Authentication Header (AH)** The heart of the Authentication Header is the *integrity check value (ICV)* field. The ICV is computed by the source and computed again by the destination for verification. This procedure provides both connectionless integrity and data origin authentication. Connectionless integrity detects modifications to the payload. Data origin authentication verifies the identity of the source of the data. The AH also contains a *sequence number* field that can be used to detect packet replay attacks, which tie up receiving system resources. By examining the sequence numbers, we can spot the arrival of duplicate IP packets.
- **Encrypted Security Payload (ESP) Header** IPv6 can provide confidentiality by encrypting the payload. The IPV6 ESP header contains a *security parameter index (SPI)* field that refers to a security association telling the destination how the payload is encrypted. ESP headers may be used end-to-end or for tunneling. When tunneling, the original IPv6 header and payload are both encrypted and jacketed by outer IPv6 and ESP headers. Near the destination, a security gateway strips away the

outer headers and decrypts the original header and payload. This encapsulation provides limited traffic flow confidentiality because a traffic analyzer may see the outer headers but not the inner encrypted header and payload.

Mobility

A growing number of Internet users work while traveling. This places primary importance on IPv6's ability to support mobile hosts, such as laptops. IPv6 introduces four concepts that are key to its support of mobile computing:

- Home Address
- Care-of Address
- Binding
- Home Agent

In IPv6, mobile hosts are identified by a home address, regardless of where they are attached at the moment. When a mobile host changes from one subnet to another, it must acquire a *care-of* address through the autoconfiguration process. The association between the home address and the care-of address is called a binding. When the mobile host acquires a care-of address, it notifies its home agent with a Binding Update message. The home agent maintains a mapping between home addresses and care-of addresses, called a *binding cache*.
A mobile host can be reached by sending a packet to its home address. If the mobile host is not connected to its home network, the home agent will forward the packet to the mobile host via its care-of address. The mobile host will then send a Binding Update message to the source node. The source node will update its binding cache and send subsequent packets directly to the mobile node via its care-of address. Only the first packet exchanged between a source node and a mobile host passes through the home agent. All subsequent packets are passed directly between the source node and the mobile host. This redirection function of IPv6 ensures scalability when supporting mobility.

The proposed use of IPv6 for cell phones may provide an example of the intended operation of a mobile host. When an IPv6 cell phone is activated to receive calls, it acquires a care-of address from the cell it is in. The cell phone notifies its home agent, which is maintained by the service provider. Calls received by the home agent are forwarded to the cell phone via the care-of address, and the cell phone sends a Binding Update message to the originating

phone. The originating phone or the originating service provider updates its binding cache and sends subsequent packets directly to the cell phone via its care-of address.

If the cell phone moves into another cell, another care-of address is acquired. Binding Update messages are sent to the home agent and to the originating phone or service provider. Subsequent packets are sent directly to the cell phone via the new care-of address. Previous care-of addresses may be maintained in the binding caches to allow the cell phone to receive packets from a previous cell if warranted by changes in signal strength.

Performance

The IPv6 architecture provides advantages in network performance and scalability. These advantages include:

- **Reduced Address Translation Overhead** To overcome limitations in its address space, IPv4 permits the use of private addresses that are restricted for use only within the private network. Network address translation must be used to map private addresses to a limited pool of public addresses. This address translation represents network performance overhead. In IPv6, address translation to overcome address space limitations is unnecessary.
- **Reduced Routing Overhead** Many IPv4 address blocks (such as Class C address blocks) are allocated to users without respect to aggregation. This results in disjointed subnets, each requiring separate routing table entries. They cannot be aggregated and be represented as a single network. This greatly increases routing table sizes and routing performance overhead. In contrast, the IPv6 addresses are allocated via service providers to encourage an addressing hierarchy that reduces routing overhead.
- **Increased Route Stability** In IPv4, route flapping results when an unreliable link is repeatedly withdrawn and re-advertised. The advertising and processing of these routing changes poses a burden to the Internet backbone. In IPv6, a single provider can aggregate the routes of many networks and allow route flapping to be isolated to that provider's network. Routing changes need only to be advertised between peer routers in a provider's network.

- **Reduced Broadcasts** IPv4's Address Resolution Protocol (ARP) uses broadcasts to map link-layer addresses with network-layer addresses. IPv6 uses Neighbor Discovery to perform a similar function during the auto-configuration process without the use of ARP broadcasts.
- **Scoped Multicasts** In IPv6, a multicast address contains a *scope* field that can restrict multicast packets to the node, the link, or the organization. In IPv4, implementing similar restrictions requires the implementation of filters and private address spaces.
- **Streamlined Header** In contrast to the 12 fixed-length fields and one variable-length field in the IPv4 header, the streamlined IPv6 header has only eight fixed-length fields. To implement extended functions, extension headers can be used that need not be checked by intermediate routers. This streamlined header architecture lowers network overhead.
- **No Intermediate Node Fragmentation** In IPv4, when an intermediate node or router receives a packet that is too large to be forwarded, the router may fragment the packet. This costly function is not supported in IPv6. Instead, only the source node will perform packet fragmentation. To assist the source node, IPv6 provides a *Path MTU Discovery* function to determine the MTU size for the path from source to destination.
- **No Header Checksum** The IPv4 header provides a *checksum* field to allow for error detection at each hop in the network. To eliminate the overhead of checksum processing by each hop, IPv6 eliminates the *header checksum* field. While this may cause erroneous packets to be forwarded, the reliability of today's network links reduces that probability. Checksum verification is already performed at the source and destination by upper-layer processes such as TCP and UDP. In IPv6, checksum processing is solely the responsibility of the source and destination. This greatly reduces network overhead.

Comparing IPv6 to IPv4

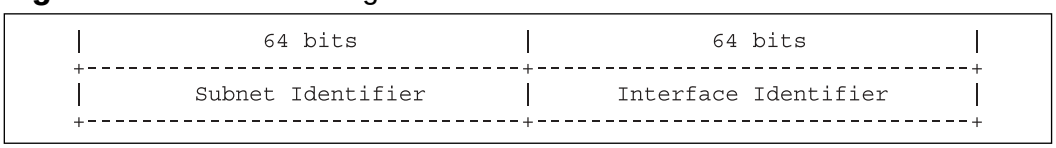
IPv6 clearly differs from IPv4 is many significant ways. Let's examine the most significant differences between the two versions of the protocol. This will allow those who have worked with IPv4 to easily recognize the major differences with the new protocol version. The most significant differences are:

- Streamlined Header Format
- Flow Label
- 128-bit Network Addresses
- Elimination of Header Checksum
- Fragmentation Only by Source Host
- Extension Headers
- Built-in Security

Addressing Structure

The IPv6 unicast address is 128 bits long and is comprised of a subnet prefix and an interface identifier. For Aggregatable Global Unicast addresses, both the prefix and the interface ID are 64 bits long, as illustrated in Figure 2.6. The subnet prefix is the network number assigned to the link. The interface identifier is derived from the node's Media Access Control (MAC) address. During IPv6 address autoconfiguration, the host node supplies its own interface ID from a ROM and queries the local router or DHCPv6 server for a subnet prefix.

Figure 2.6 IPv6 Addressing Format



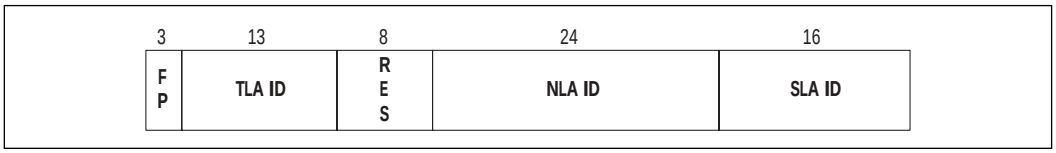
Today's MAC addresses are only 48-bits long so 16 bits in the interface ID is reserved. The IEEE has proposed a MAC address that is 64 bits long, known as EUI-64.

In contrast, the IPv4 addresses are only 32 bits long. They are also comprised of a subnet number and a host number. The IPv4 addresses are not derived from MAC addresses. Network administrators assign both the subnet numbers and the host numbers.

Address Administration

The 64-bit subnet prefix of an IPv6 Aggregatable Global Unicast address is further subdivided into five fields. These five fields are illustrated in Figure 2.7. The first field is the *format prefix (FP)* field, which identifies an Aggregatable Global Unicast address with a binary value of 001. The third field is reserved for future use.

Figure 2.7 Subnet Prefix of Aggregatable Global Unicast Address



Two fields, the *TLA ID* and the *NLA ID*, are key to understanding IPv6's support for an aggregatable addressing hierarchy. The TLA ID is the Top-Level Aggregation Identifier. IPv6 global addresses will be assigned to service providers or TLA organizations. The TLA organizations will in turn allocate addressing space to the Next-Level Aggregation (NLA) organizations. This hierarchical method of allocating address space encourages address aggregation to reduce the size of core routing tables.

In contrast, IPv4 addresses are commonly allocated by CIDR blocks. Each CIDR block is comprised of one or more Class C addresses. Each Class C block can address approximately 254 devices. Unfortunately, CIDR blocks allocated to different organizations cannot be easily aggregated, and each CIDR block may require a separate routing table entry in a core router. The issuance of CIDR blocks has caused explosive growth in the size of core routing tables.

To local network administrators, the most important field may be the *Site-Level Aggregation (SLA) Identifier*. Unlike the TLA ID and the NLA ID, the SLA ID is not usually delegated to a downstream organization with a pre-assigned value. Per RFC 2374, the SLA ID allows an organization to define its own local subnets and addressing hierarchy. The 16 bits provided by the SLA ID for subnet identifiers can support 65,535 subnets, enough for all but the largest organizations. To support even larger networks, a downstream organization may request that a lower-order portion of the NLA ID be delegated.

Herein lies a key advantage of the IPv6 addressing structure. A network administrator need not worry about allocating addresses for both subnet and host identifiers as he must in IPv4. In IPv4, both subnet and host identifiers often come from the same limited pool of available addresses. An IPv4 network administrator often "borrows" host identifier fields to number his subnets. In IPv6, the top half of the IPv6 address structure provides enough addressing space for subnet identifiers. In a separate and virtually inexhaustible field, the IPv6 host identifiers are uniquely generated for each host by a separate autoconfiguration process.

IPv6 eases the network administrator's burden in another way: Aggregatable Global Unicast addresses do not require address translation when used to access external networks such as the Internet. In IPv4, private address spaces are used

when global addresses are unavailable. These private addresses must be translated to a limited set of global addresses when accessing external networks. IPv4 address translation schemes include Network Address Translation (NAT) and Port Address Translation (PAT). IPv6 virtually eliminates the need for address translation as a means of accessing external networks.

Table 2.3 illustrates the reduced address administration burden placed upon IPv6 network administrators.

Table 2.3 Address Administration Comparison

Address Administration Issues	IPv4 Private Class A Block	IPv6 Aggregatable Global Unicast
Address Length	32 bits	128 bits
Length of Pre-assigned Upstream Fields	8 bits	48 bits
Length of Delegated Addressing Fields	24 bits	80 bits
Host Identifier Length	24 – subnet bits	64 bits
Subnet Identifier Length	24 – host bits	16 bits (SLA ID)
Allocate host addresses for subnet identifiers	Yes	No
Determine subnet identifiers	Yes	Yes
Determine host identifiers	Yes	No
Address Translation Required (NAT/PAT)	Yes	No

Header Comparison

IPv6 provides a more streamlined header than that of IPv4. Five fields are eliminated, including the variable-length IPv4 *options* field. Removal of the variable-length field and other fields permits the IPv6 header to have a fixed format of 40 bytes in length. A comparison of the two types of headers is summarized in Table 2.4.

Table 2.4 Header Comparison

Header	IPv4	IPv6
Header Format	Variable	Fixed
Header Fields	13	8
Header Length	20-60 bytes	40 bytes

Continued

Table 2.4 Continued

Header	IPv4	IPv6
Address Length	32 bits	128 bits
Header Checksum	Yes	No
Fragmentation Fields	Yes	No
Extension Headers	No	Yes

To provide for additional options, IPv6 defines the following extension headers, which will be discussed in detail in the next chapter:

- Hop-by-Hop Options header
- Destination Options header
- Routing header
- Fragment header
- Authentication header
- Encapsulating Security Payload header

Feature Comparison

The IPv6 architecture contains integrated features that are not contained in IPv4. Table 2.5 contrasts the features of IPv4 and IPv6.

Table 2.5 Feature Chart

Feature	IPv4	IPv6
Anycast Address	No	Yes
Multicast Scoping	No	Yes
Security Support	No	Yes
Mobility Support	No	Yes
Autoconfiguration	No	Yes
Router Discovery	No	Neighbor Discovery
Multicast Membership	IGMP	Multicast Listener Discovery
Router Fragmentation	Yes	Source only

Anycast addresses and multicast scoping are not found in IPv4. While security is a function of upper-layer protocols in IPv4, IPv6 provides integrated security support through the use of authentication and encryption headers. For mobility, IPv6 provides the functions of the home agent, care-of addresses, and binding caches.

IPv6 provides Plug and Play functionality. A host address and prefix length can be autoconfigured rather than manually entered, and routers and neighbors can be discovered. IPv4 contains no built-in discovery function—it requires that basic IP parameters be manually configured or obtained from external sources, such as DNS servers.

For multicast membership support, IPv4 relies on augmentations such as Internet Group Management Protocol (IGMP). In contrast, IPv6 has a built-in Multicast Listener Discovery (MLD) protocol. MLD allows a router to determine which of its ports contain multicast listeners and to add or prune multicast transmissions appropriately.

IPv4 allows routers to fragment packets, which causes greater network overhead. IPv6 only permits the source node to fragment a packet. It provides a *Path MTU Discovery* function that allows the source to fragment packets efficiently. This eliminates the need for network devices to further fragment a packet along the path to its destination.

Examining IPv6 Network Architecture

The Internet has become a victim of its own success. The worldwide success of the Internet has resulted in an explosion of new users and applications, and this growth has placed new demands upon the network infrastructure and upon network administrators. In the early 1990s, the IETF began to design IPv6 with the objective of improving IPv4 to meet these new demands. The areas targeted for improvement included:

- **Address Space Depletion** The depletion of IP addresses has been foreseen for many years and many patches and extensions have been added to IPv4 to alleviate and postpone the looming crisis. Among those extensions are variable-length subnet masking (VLSM), Classless Inter-Domain Routing (CIDR), Network Address Translation (NAT), Port Address Translation (PAT), and private address spaces. IPv6 introduces a large unified addressing structure that will make many of these complex extensions obsolete.

- **Network Performance** The Internet has outgrown many features in IPv4 that now encumber network performance. Among these features are header checksums, MTU size, and packet fragmentation. IPv6 is streamlined to reduce protocol overhead.
- **Security** IPv4 was not designed with security in mind—security was considered the responsibility of higher layers in the Open Systems Interconnect (OSI) model. IPv6 provides integrated security support for encryption and authentication.
- **Plug and Play** Configuring nodes in an IPv4 network has always been complicated. Many configuration tasks are manually intensive and not practical in large networks. A case in point is the renumbering of a network when a new Internet service provider is selected. The growth of mobile computing has also added to the workload of network administrators. IPv6's autoconfiguration facilities take us a step closer to true "Plug and Play" computing.

IPv6 Communication Fundamentals

In the following sections, we will examine in further detail how communication occurs between the devices on a network and how IPv6 facilitates that communication. We will examine communication between hosts of the same subnet as well as host and router communication between subnets.

Intra-Subnet Communications

How many of us have cringed at the prospect of connecting our computers or laptops to a network for the first time? A computer must be configured to be able to communicate on a network. Likewise, a network administrator must configure his network devices to facilitate communication between hosts. Nodes at both ends of each network link must have compatible configurations. Many patches have been added to IPv4 to make the configuration process less manually intensive.

IPv6 is designed for Plug and Play. Support for autoconfiguration is built into IPv6. We will first examine stateless autoconfiguration and its role in intra-subnet communications. Later, we will discuss stateful autoconfiguration and its role in inter-subnet communications.

The keys to understanding intra-subnet communications are the following concepts:

- Stateless Autoconfiguration
- Link-Local Address
- Link-Local Prefix
- Interface Identifier
- Neighbor Solicitation message
- Neighbor Advertisement message
- Neighbor Cache

Let's imagine a subnet in a dentist's office. The subnet is comprised of a few workstations and printers. It has no routers, no connections to the Internet, and no servers to assist in the configuration process. A host on such a subnet must configure its own IPv6 address with a process known as *stateless autoconfiguration*.

When a workstation is connected to a port on the subnet, the workstation automatically configures a tentative address, known as the link-local address. This address is formed using the hardware address of the workstation's network interface. The link-local address configured by the workstation is 128 bits long and is comprised of a local-link prefix and the workstation's interface identifier. The local-link prefix is an all-zero network identifier prefaced with the hex digits, FE8. The interface ID, also known as the Media Access Control (MAC) address, resides in a ROM on the interface hardware. Today's MAC addresses are 48 bits long, but new specifications will support 64-bit MAC addresses. A typical local-link address takes the following form, where the x's indicate the 64-bit interface ID:

```
FE80:0:0:0:xxxx:xxxx:xxxx:xxxx.
```

To ensure a unique address, the workstation will send a special Neighbor Solicitation message to the newly configured address and wait one second for a reply. If no Neighbor Advertisement message is returned, the new link-local address is assumed to be unique. (Later, we will see that the Neighbor Solicitation and Neighbor Advertisement messages are also used for other functions that are part of the IPv6 Neighbor Discovery protocol.)

After verifying the uniqueness of the link-local address, the next phase is to query for neighboring routers on the network. In our example of a subnet in a dentist's office, no routers will be found. The workstation is now ready to begin communications with its neighbors.

To communicate with a destination host on the same subnet, the workstation must discover the destination's interface identifier. To do so, the workstation uses the functions provided by the IPv6 Neighbor Discovery protocol. The workstation

sends a Neighbor Solicitation message to the destination and the interface identifier is returned in a Neighbor Advertisement message. This interface ID is placed in a header before the IPv6 header and transmitted on the subnet. The workstation then adds an entry to its Neighbor Cache. The entry contains the destination's IPv6 address, its interface identifier, a pointer to packets pending transmission, and a flag indicating whether or not the destination is a router. This cache will be used for future transmissions instead of sending another solicitation message.

The link-local addresses cannot be used for communications outside the local subnet. For inter-subnet communications, site-local addresses or global addresses must be used in conjunction with routers. We will discuss inter-subnet communications in the next section.

Inter-Subnet Communications

Suppose that in the previous example, our workstation discovered that a router *did* exist on the subnet. How would the autoconfiguration process differ, and how would the workstation communicate with hosts on different subnets? To discuss inter-subnet communications, we will expound on the stateless autoconfiguration process and introduce the following concepts:

- Neighbor Discovery
- Site-Local Address
- Subnet Identifier
- Router Solicitation Message
- Router Advertisement Message
- Default Router List Cache
- Destination Cache
- Prefix List Cache
- Redirect Message
- Path MTU Discovery

During and after autoconfiguration, the workstation relies heavily on the IPv6 Neighbor Discovery protocol. The Neighbor Discovery protocol allows nodes on the same subnet to discover each other and to find routers for use as the next hop towards a destination on another subnet. The Neighbor Discovery protocol replaces the IPv4 Address Resolution Protocol (ARP), the IPv4 default gateway process, and the IPv4 redirect process.

During the autoconfiguration process, after the workstation generates a unique link-local address, it queries for a router. The workstation sends a Router Solicitation message and a router responds with a Router Advertisement message.

The presence of a router indicates that there may be other subnets connected to the router. Each subnet must have its own subnet identifier, because routing is dependent on unique subnet numbers. Host identifiers are not used for making routing decisions. The workstation address must now have a unique subnet identifier. The link-local address with its zero subnet ID is not sufficient for inter-subnet communications.

To support stateless autoconfiguration, the Router Advertisement contains a subnet identifier. Router Advertisements for each router interface contain a different subnet identifier. This identifier will be concatenated with the interface identifier to form the workstation's IPv6 address.

The workstation will discard its tentative link-local address and configure a new address that is known as the *site-local* address. The site-local address contains a 16-bit subnet ID and has the following format, where the x's indicate the 64-bit interface ID:

```
FEC0:0:0:<subnet ID>:xxxx:xxxx:xxxx:xxxx.
```

The workstation will use information from the Router Advertisement to update its caches. The subnet ID is added to the workstation's Prefix List cache. This cache will be used to determine if an address is on the workstation's subnet (on-link) or not (off-link). The router's information will be added to the Neighbor cache and Destination cache. If the router can be used as a default router, an entry will be added to the Default Router List cache.

When the workstation is ready to send a packet to a destination host, it queries the Prefix List to determine whether the destination's IPv6 address is on-link or off-link. If the destination host is off-link, the packet will be transmitted to the next hop, which is the router in the Default Router List. The workstation will then update its Destination cache with an entry for the destination host and its next hop address. If the default router selected is not the optimal next hop to the destination, the router will send a Redirect message to the source workstation with the new recommended next hop router for the destination. The workstation will then update its Destination cache with the new next hop for the destination.

The caches are maintained by each IPv6 host and are queried before solicitation messages are transmitted. The caches reduce the number of solicitation and advertisement messages that need to be sent. The caches are periodically purged of expired information, and they are constantly updated.

To facilitate inter-subnet communications, IPv6 provides another useful service, Path MTU Discovery. IPv6 does not allow routers to fragment packets that are too large to be forwarded through the next hop link or interface; only the source node may fragment a packet. Using IPv6's Path MTU Discovery service, a source node can determine the largest packet that may be sent to the destination. With this information, the source node can appropriately resize its packets prior to transmission.

Site-local addresses can only be used for communications within the site. For communications beyond the site, global addresses must be assigned with a more scalable autoconfiguration procedure.

Internetwork Communications

In stateless autoconfiguration, each node is responsible for configuring its own address and caches using its interface identifier and information provided by the distributed Neighbor Discovery protocol. In small networks, stateless autoconfiguration is advantageous for its simplicity and ease of use. Its disadvantages include relying on multicast discovery mechanisms, using address space inefficiently, and lacking in security and control over policy and access.

To facilitate communications in larger and more complex internetworks, it may be desirable to manage the autoconfiguration process using a procedure known as *stateful autoconfiguration*. During our discussion of this process, we will introduce the following concepts:

- Stateful Autoconfiguration
- Dynamic Host Configuration Protocol Version 6 (DHCPv6)
- DHCPv6 Client, Relay, Agent, Server

Stateful autoconfiguration relies on servers to provide the bulk of the configuration information, including the network information required for obtaining an Aggregatable Global Unicast address. These servers are known as Dynamic Host Configuration Protocol version 6 (DHCPv6) servers. From a network administrator's point of view, stateful autoconfiguration is more complex than stateless autoconfiguration because it requires that configuration information be entered into a DHCPv6 database. On the other hand, stateful autoconfiguration provides greater scalability when administering to large networks.

Stateful autoconfiguration can be used simultaneously with stateless autoconfiguration. For example, a node may follow the stateless procedure upon startup to obtain a link-local address. After obtaining the link-local address, it may use

stateful autoconfiguration to interact with and obtain additional information from a DHCPv6 server.

To obtain configuration information, a workstation first locates a DHCPv6 server by issuing a DHCP Solicit message or by listening for a DHCPv6 Advertisement. The workstation then issues a unicast DHCPv6 Request. If a DHCPv6 server is not on the local subnet, then a DHCPv6 Relay or Agent will forward the request to a server on behalf of the workstation. The server will respond with a DHCPv6 Reply that contains configuration information for the workstation.

The use of a DHCPv6 service has several advantages:

- **Control** The DHCPv6 service controls the distribution and assignment of addresses from a central control point.
- **Aggregation** Through the thoughtful distribution of addresses, an addressing hierarchy can be built to ensure address aggregation.
- **Renumbering** When a new Internet service provider is chosen to replace the old provider, new addresses can more easily be distributed with the DHCPv6 service.
- **Security** A host registration system can be enforced with the DHCPv6 service. This registration system can selectively provide network services to registered hosts and deny access to unregistered hosts.

Designing & Planning…

Autoconfiguration

Stateless and stateful autoconfiguration can coexist in IPv6. When designing and planning for the IPv6 numbering of a site, it is often desirable to plan for both autoconfiguration methods. We'll begin by classifying various types of nodes based on their need to access the site and the Internet:

- Workstations that do not need to communicate beyond their local subnet may use stateless autoconfiguration to obtain link-local addresses.
- Workstations that need to communicate throughout the site but don't need to access the Internet can use stateless

Continued

autoconfiguration to obtain site-local addresses from their local routers.

- Workstations that need access to the Internet can obtain aggregatable global unicast addresses from a DHCPv6 server. DHCPv6 allows address distribution to be controlled from a central point.

Upper-Layer Protocol Issues

In general, the layered architecture shields the upper-layer protocols from changes in the network layers. However, there are several issues need to be addressed. Upper layer protocols that compute checksums over packets must account for changes in IPv6, including the use of 128-bit addresses, having a final destination instead of intermediate ones when the Routing header is used, and so on.

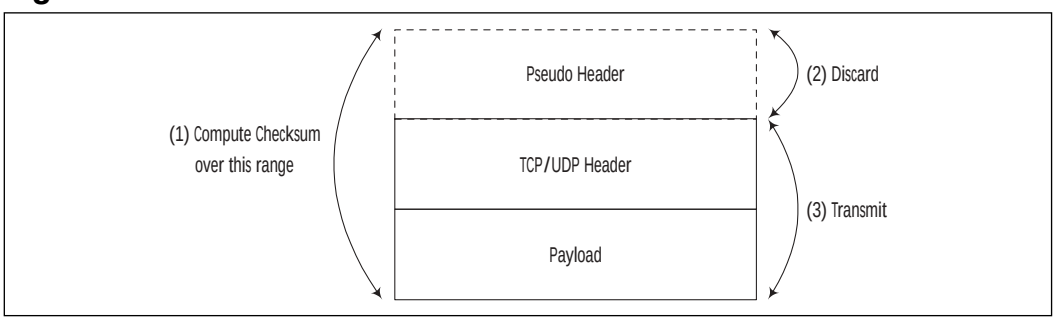
As discussed earlier, the *Time To Live* field (which behaves differently than its original definition) has been renamed the *hop limit*. Any upper-layer protocol that relies on the original meaning of the Time To Live may have to make necessary adjustments. The maximum upper-layer payload size also needs to be adjusted to reflect the length of the IPv6 header (40 bytes).

- **Upper-Layer Checksums** Currently, an upper-layer transport protocol such as TCP and UDP will concatenate a pseudo header before its payload when computing the transport layer checksum. This pseudo header contains the source and destination IPv4 addresses. For IPv6, the pseudo header must be expanded to include the larger addresses, the upper-layer packet length and the *next header* field. The checksum is computed over the IPv6 pseudo header, the TCP or UDP header, and the TCP or UDP payload. Figure 2.8 illustrates the checksum being calculated over an IPv6 pseudo header.
- **Maximum Packet Lifetimes** The IPv4 header has a *Time To Live* field that is used to determine when a packet can be discarded if it has not reached its destination. In contains either a hop count or a time in seconds. IPv6 has renamed this field as the *Hop Limit*, and the *time in seconds* measurement is no longer supported. Applications using this field for time data must be revised.
- **Maximum Upper-Layer Payload Size** The nominal IPv6 header is 40 bytes long. The nominal IPv4 header is 20 bytes long. Replacing the

IPv4 header with a longer IPv6 header will result in larger packet sizes, which may have upper-layer consequences.

- **Routing Headers & Security** The IPv6 routing header extension contains the intermediate nodes that the packet must traverse on the way to its destination. When a packet with a routing header is received by the destination, it should not assume that the reverse path to the source is appropriate. In fact, responding along the reverse path may facilitate certain types of security breaches.
- **Domain Name System (DNS)** The DNS is a distributed database system that defines a hierarchical naming convention for host nodes and maps these host names to IP addresses. For example, www.syngress.com maps to IPv4 address 216.238.176.55. IPv6 enhancements to DNS include new record types with the 128-bit IPv6 address and a new service that can return a hostname when given its IPv6 address.
- **Application Programming Interface (API)** Application programs written for IPv4 must be converted to use APIs written for IPv6. The application programs must contain new data structures for the longer IPv6 addresses. Functions that manipulate IPv4 addresses must be substituted with functions that can manipulate IPv6 addresses.

Figure 2.8 TCP/UDP Checksum Calculation



Understanding ICMPv6

The Internet Control Message Protocol version 6 (ICMPv6) is a key part of the IPv6 architecture. ICMPv6 performs the feedback functions necessary to ensure the smooth operation of IPv6 processes. These functions include:

- Packet Processing Error Reporting
- Diagnostics
- Neighbor Discovery
- Multicast Membership Reporting

ICMPv6 is streamlined to delete ICMPv4 functions that are no longer used, and it combines the functions of three separate IPv4 protocols: ICMPv4, Internet Group Membership Protocol (IGMP), and Address Resolution Protocol (ARP). ICMPv6 messages can be divided into *error messages* and *information messages*.

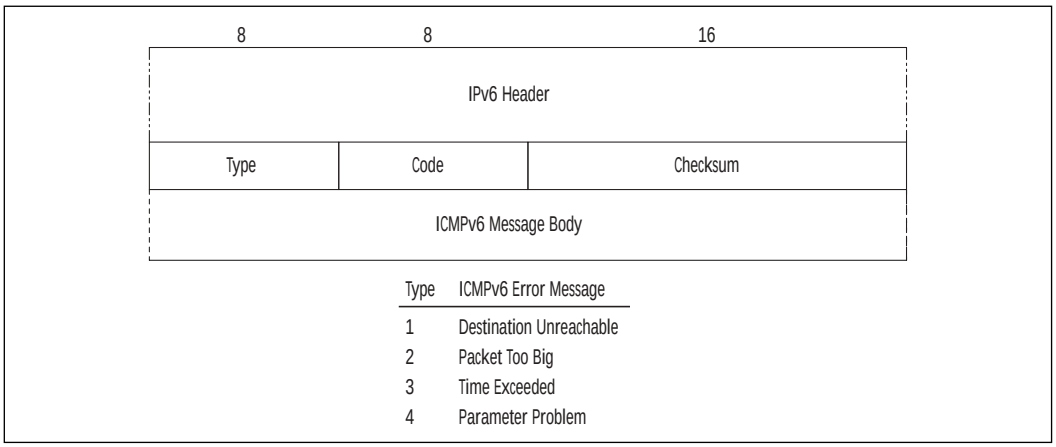
Error Messages

ICMPv6 issues error messages pertain to packet processing. These error messages include:

- **Destination Unreachable** The source host or a router generates this message when a packet cannot be delivered for any reason other than congestion.
- **Packet Too Big** The router generates this message when a packet cannot be forwarded because it is larger than the MTU of the next hop link. The MTU of the next link is returned in the message. This message is used by the Path MTU Discovery function.
- **Time Exceeded** The IPv6 header contains a Hop Limit that is decremented by each router that forwards the packet. When this hop limit reaches zero, the router discards the packet and returns a Time Exceeded message to the source node. This function is the basis of the traceroute function; it traces the route to a destination by sending packets towards the destination with incremental hop limits. The Time Exceeded messages returned are used to identify the routers along the path.
- **Parameter Problem** When a problem with some part of an IPv6 header keeps a router from successfully processing the packet, the packet is discarded and the router returns a Parameter Problem message to the source node.

Each ICMPv6 error message includes three fixed-length fields plus a variable-length message body. Figure 2.9 illustrates the format of the error message.

Figure 2.9 ICMPv6 Error Message



Type	ICMPv6 Error Message
1	Destination Unreachable
2	Packet Too Big
3	Time Exceeded
4	Parameter Problem

Informational Messages

ICMPv6 can also report informational messages. These include:

- **Diagnostic Messages** Diagnostic messages include the echo request and reply. When a destination receives the echo request, it returns a reply. This echo request and reply are used to implement the ping diagnostic function. The ping function is important for determining whether or not a particular destination is connected to the same network as the source.
- **Multicast Listener Discovery (MLD) Messages** These messages enable a router to discover neighboring nodes that wish to receive multicast packets, and what multicast addresses are of interest to them. A router transmits an MLD Query to learn whether a multicast address (or addresses) has listeners on a link. Nodes respond affirmatively with an MLD Report message. When a node ceases to listen to a multicast address, it sends an MLD Done message.
- **Neighbor Discovery Messages** ICMPv6 provides messages necessary for the Neighbor Discovery protocol. These messages include router solicitation and advertisement, neighbor solicitation and advertisement, and redirect. These messages include information options such as the source link-layer address, the target link-layer address, prefix information, the redirected header, and the MTU size.

ICMPv6 informational messages have the same format as the ICMPv6 error message displayed in Figure 2.9. The *type* field values for informational messages range from 128 to 255. Table 2.6 shows some of the common *type* fields for ICMPv6 informational messages.

Table 2.6 ICMPv6 Informational Messages

Type Field Value	ICMPv6 Informational Message
128	Echo Request
129	Echo Reply
130	Multicast Listener Query
131	Multicast Listener Report
132	Multicast Listener Done
133	Router Solicitation
134	Router Advertisement
135	Neighbor Solicitation
136	Neighbor Advertisement
137	Redirect

Understanding Neighbor Discovery

IPv6's Neighbor Discovery protocol is used to obtain information that facilitates the packet forwarding process. The information gathered by the Neighbor Discovery protocol can be used for:

- Next Hop Determination
- Address Resolution
- Prefix Discovery
- Parameter Discovery
- Redirection

Five ICMPv6 messages are used in the Neighbor Discovery protocol. We will discuss Neighbor Discovery with respect to these five messages.

Router Solicitation and Advertisement

During the autoconfiguration process, after the workstation generates a unique link-local address, it queries for a router. The workstation sends a Router Solicitation message and listens for a Router Advertisement message.

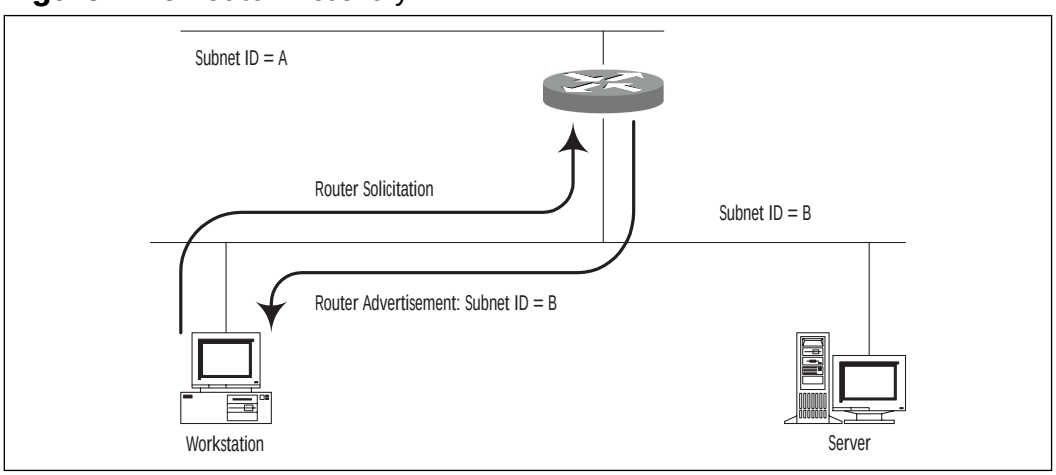
The presence of a router indicates that there may be other subnets connected to the router. Each subnet must have its own subnet identifier because routing is dependent on unique subnet numbers. Host identifiers are not used to make routing decisions. The workstation address must now have a unique subnet identifier. The link-local address, with its zero subnet ID, is not sufficient for inter-subnet communications.

The Router Advertisement contains a network number or prefix. The prefix may contain an aggregatable global unicast prefix or simply a subnet identifier. Router Advertisements for each router interface contain different prefixes. This prefix will be concatenated with the interface identifier to form the workstation's IPv6 address.

The workstation uses information from the Router Advertisement to update its caches. The subnet ID is added to the workstation's Prefix List cache. This cache will be used to determine if an address is on the workstation's subnet (on-link) or not (off-net). The router's information will be added to the Neighbor cache and the Destination cache. If the router can be used as a default router, an entry will be added to the Default Router List cache.

Figure 2.10 illustrates a workstation during the autoconfiguration process. The workstation solicits the local router and receives the subnet identifier it needs to complete its host IPv6 address.

Figure 2.10 Router Discovery

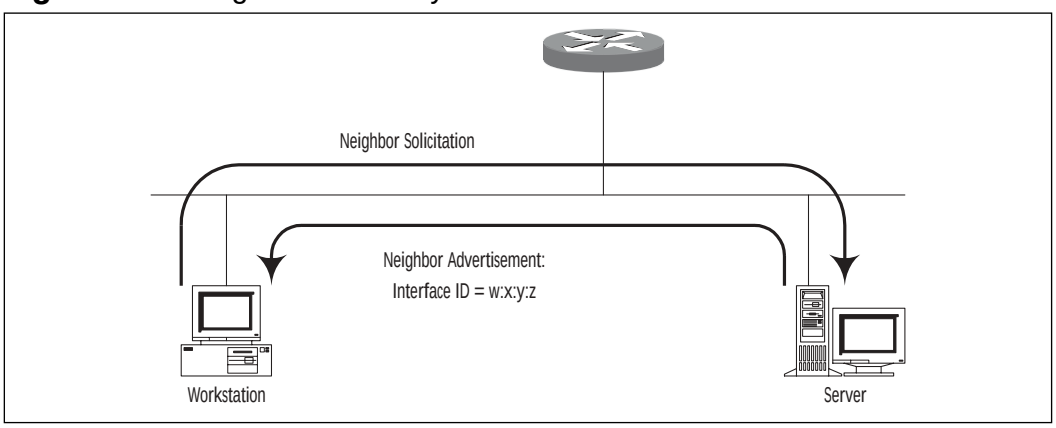


Neighbor Solicitation & Advertisement

To communicate with a destination host on the same subnet, the workstation must discover the destination's interface identifier. To do so, the workstation uses the functions provided by the IPv6 Neighbor Discovery protocol. The workstation sends a Neighbor Solicitation message to the destination, and the interface identifier is returned in a Neighbor Advertisement message. This interface ID is placed in a header before the IPv6 header and transmitted on the subnet. The workstation then adds an entry to its Neighbor Cache containing the destination's IPv6 address and interface identifier, a pointer to packets pending transmission, and a flag indicating whether the destination is a router. This cache will be used for future transmissions (instead of sending duplicate solicitation messages).

Figure 2.11 illustrates how Neighbor Solicitation and Advertisement messages play a key role in the Neighbor Discovery process.

Figure 2.11 Neighbor Discovery



Redirect Message

Routers issue the Redirect message to inform other nodes of a better first hop to the destination. A node can be redirected to another router on the same link. Here's how redirection works in the packet processing process:

When the workstation is ready to send a packet to a destination host, it queries the Prefix List to determine whether the destination's IPv6 address is on-link or off-link. If the destination host is off-link, the packet will be transmitted the next hop, which is the router in the Default Router List. The workstation will then update its Destination cache with an entry for the destination host and its next hop address. If the default router selected is not the optimal next hop to

the destination, the router will send a Redirect message to the source workstation with the new recommended next hop router for the destination. The workstation will then update its Destination Cache with the new next hop for the destination.

Message Options

Neighbor Discovery messages may contain additional information options. These options include:

- **Source Link-Layer Address Option** This option contains the link-layer address of the *source* of the message. It is used in Router Solicitation, Router Advertisement, and Neighbor Solicitation messages.
- **Target Link-Layer Address Option** This option contains the link-layer address of the *target* of the message. It is used in Neighbor Advertisement and Redirect messages.
- **Prefix Information Option** This option contains prefixes for address autoconfiguration. It is used in Router Advertisements.
- **Redirected Header Option** This option contains all or part of the packet that is being redirected. It is used in Redirect messages.
- **MTU Option** This option contains the MTU size of the link. It is used in Router Advertisements.

Summary

IP and its extensions have withstood the test of time over the last three decades, during which we saw an explosion of new users and new applications. Today, its success has prompted its re-examination. The Internet Engineering Task Force (IETF) has designed the next version of IP to meet the growing needs of the Internet. IPv6 is much more than simply an extension of IPv4 with a larger address space—its architecture was designed to provide easier administration and greater performance, security and mobility.

IPv6 takes us closer to Plug and Play computing and greatly eases the burden put upon network administrators. IPv6's stateless autoconfiguration allows subnet communications literally "out of the box." The Neighbor Discovery protocol automatically solicits and caches information needed for packet processing. DHCPv6 uses IPv6's hierarchical addressing structure to allow even the daunting task of site renumbering to proceed with relative ease. ICMPv6 provides an effective set of diagnostic and information-gathering functions.

IPv6's Plug and Play support extends to mobile computing: Mobile hosts acquire care-of addresses as they travel, and home agents bind the care-of addresses to home addresses to ensure that mobile users retain connectivity away from home. Redirection allows a source host to communicate directly with a mobile host.

IPv6 provides integrated security support with two extension headers. The Authentication Header's *Integrity Check Value (ICV)* field supports both connectionless integrity and data origin authentication. The *sequence number* field can be used to detect packet replay attacks. Encrypted payloads can be transmitted with the Encrypted Security Payload (ESP) Header. The header's *SPI* field contains a security association that tells the destination how the payload is encrypted. ESP headers may be used end-to-end or for tunneling.

The address space of IPv6 is not only larger than IPv4's address space; it provides an aggregation hierarchy to simplify address administration and to reduce core routing table sizes. The use of MAC addresses to form the host portion of the address increases the probability of uniqueness during autoconfiguration.

IPv6's design enhances network performance. Its larger address space eliminates the need for network and port address translation, reducing overhead. Core routing overhead is significantly reduced by the ability to aggregate addresses and reduces the size of routing tables. Aggregation enhances core router stability by allowing route flapping to be isolated to a provider's network. IPv6's streamlined header architecture eliminates header checksums and router fragmentation.

Scoped multicast traffic and the introduction of anycast addresses further reduce multicast traffic.

The layered network model generally protects upper-layer protocols from an expensive transition to IPv6. Obvious transitional issues related to the upper-layer protocols, which were discussed earlier, are solvable.

This author's hope is that this book will help you to understand the solutions and benefits provided by IPv6 and encourage you to explore IPv6 further.

Solutions Fast Track

Understanding the Benefits of IPv6

- ☑ The IP address size is greatly increased.
- ☑ A "top-down" allocation plan supports addressing hierarchies, enabling greater address aggregation.
- ☑ Host addressing is simplified by using the MAC address to form the host portion of the IP address.
- ☑ Plug and Play functionality is supported by simpler autoconfiguration of IP addresses.
- ☑ Multicast routing is scoped to provide greater scalability.
- ☑ Anycast addresses increase routing efficiency.
- ☑ Streamlined headers lower routing overhead.
- ☑ Security is built in to support IP security at a lower level in the OSI stack.
- ☑ Mobility is enhanced with greater scalability and ease.
- ☑ Performance is enhanced with greater efficiency and streamlining.

Comparing IPv6 to IPv4

- ☑ IPv6 enlarges the addressing structure without address translation and private address spaces.
- ☑ IPv6 address administration is "top-down" to allow for greater address aggregation.

- ☑ IPv6 headers are simpler and more efficient thanks to a fixed size and fewer fields.
- ☑ IPv6 features not included in IPv4 include enhanced support for multicasting, security, mobility, and discovery.

Examining IPv6 Network Architecture

- ☑ The impending address space depletion provided the initial impetus for developing IPv6.
- ☑ A streamlined architecture improves network performance.
- ☑ Security on the IP level is built into IPv6.
- ☑ Plug and Play has become a reality with IPv6.
- ☑ Intra-subnet communications can be autoconfigured in a Plug and Play fashion.
- ☑ Inter-subnet communications can be set up by the stateless autoconfiguration function.
- ☑ Internetwork communications can be set up by stateful autoconfiguration.

Upper-Layer Protocol Issues

- ☑ Upper-layer checksums need to be computed over a larger IPv6 pseudoheader.
- ☑ The maximum packet lifetime is expressed as an IPv6 Hop Limit, not a Time To Live.
- ☑ Maximum upper-layer payload sizes may need revision to allow for the larger IPv6 header.
- ☑ The IPv6 Routing Header must be used with care to avoid the security risks of source routing.
- ☑ The Domain Name Service (DNS) is being enhanced to support IPv6.
- ☑ Application Programming Interfaces (APIs) are being developed to support the larger IPv6 addresses.

Understanding ICMPv6

- ☑ There are four types of IPv6 error messages:
  - Destination Unreachable messages arise when a packet cannot be delivered for any reason other than congestion.
  - Packet Too Big messages arise when a packet exceeds the MTU size of the outgoing interface.
  - Time Exceeded messages arise when a packet exceeds its Hop Limit.
  - Parameter Problem messages arise when a router cannot process an IPv6 header.
- ☑ There are three types of Informational Messages.
  - Diagnostic messages include the echo request and reply messages.
  - Multicast Listener Discovery (MLD) messages are used to discover multicast listeners and the multicast addresses that are of interest to them.
  - Neighbor Discovery messages support the Neighbor Discovery protocol.

Understanding Neighbor Discovery

- ☑ Router Solicitation and Advertisement messages are used to discover local routers and to advertise router and subnet information.
- ☑ Neighbor Solicitation and Advertisement messages are used to discover neighbors and to advertise their addresses.
- ☑ A Redirect message is used to inform a node of a better first hop to a destination.
- ☑ Message Options are used to include additional information in Neighbor Discovery messages.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form.

Q: Under IPv6, if my organization were to change network service providers, would we have to renumber our host IP addresses?

A: No, the 64-bit host portion of your IPv6 address will remain the same. Only the 64-bit network number of your IPv6 address will change. The network portion of your IPv6 address is provided by a host configuration server. The server's network numbering program will change, but the host numbering will remain the same.

Q: What is the core set of RFCs specifying IPv6 header and extension headers?

A: Newer RFCs may render these obsolete, but most of the information in this chapter is based on the following RFCs:

- **RFC 2374** An IPv6 Aggregatable Global Unicast Address Format
- **RFC 2460** Internet Protocol, Version 6 (IPv6) Specification
- **RFC 2402** IP Authentication Header
- **RFC 2406** IP Encapsulating Security Payload Header

Q: What is the implementation status of IPv6?

A: IPv6 is currently being developed for many host systems and routers, including 3Com, Cisco Systems, Digital, IBM and Microsoft. A preview version of Windows 2000 that supports IPv6 can be downloaded from this URL: http://msdn.microsoft.com/downloads/sdks/platform/tpipv6.asp.

Q: How should my organization transition from IPv4 to IPv6?

A: Strategies have been devised to ease the difficulty of transitioning from IPv4 to IPv6. These strategies involve dual stack approaches and tunneling approaches to provide simultaneous support for both IPv4 and IPv6 during

the transition phase. There is ongoing research aimed at making this transition as easy as possible.

Q: What happened to IPv5?

A: Version 5 was assigned to an experimental protocol, the *Streams Protocol*, known as ST2. ST2 was designed to carry real-time traffic in parallel with IP.

Q: Will IPv6 be more complicated to implement than IPv4?

A: IPv6 was designed with autoconfiguration and Neighbor Discovery functions that will make it much easier to install and implement.