Internet Draft
<draft-lutchann-ipv6-intro-00.txt>
Expires: November 2001

Nathan Lutchansky
Cornell University
May 2001

# A Technical Introduction to IPv6

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC 2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Abstract

This document attempts to provide an introduction to each of the new features of the Internet Protocol, version 6 (IPv6), including transitional mechanisms for interoperating in a mixed IPv4/IPv6 environment.

## 1. Introduction

After over five years of discussion and development, IPv6 is finally coming into production use. The 6bone, a testbed tunneled IPv6 backbone, has been around for some time and is in experimental use by almost all major backbone providers. Several Internet exchanges are supporting IPv6 natively, including the London, Tokyo, and Chicago exchanges. ISPs in Australia and Japan are even beginning to offer native IPv6 service to their customers.

Although the IPv6 protocol has been designed to minimize incompatibilities during the transition from IPv4, the current Internet Protocol, there are many enhancements and changes of which users and network administrators need to be aware.

Because IPv6 is still mostly an experimental protocol, this document is targeted at individuals having prior experience with TCP/IP who are interested in learning about IPv6 and its current state of development. A solid knowledge of IP addressing, network topology, and local network routing is recommended, as well as a vague understanding of how routing operates on the Internet at large. Readers without experience with these concepts should become familiar with TCP/IP before beginning experimentation with IPv6.

## 2. The History of the Internet

The primary motivation for revamping IP was the realization that the addressing scheme of IPv4 didn't provide enough addresses for all the computers and networks that are on the Internet today.

### 2.1. In the beginning...

When the network was first created, sites on the Internet created network links directly to each other, and large blocks of addresses were assigned to each institution. For example, Stanford, like many of the first Internet sites, was allotted all addresses having a certain first octet of the IP address (36 for Stanford) --- only the first 8 bits of the IP address were needed to know that a packet was destined for the Stanford network. Although it seems wasteful and short-sighted in retrospect, it was actually the most intelligent way to design the network. By aggregating each site behind a large subnet, only one route in every router on the Internet was needed for each institution, regardless of whether the institution had 10 computers, or 16 million.

Three designations were made: "Class A" networks would have an 8-bit address to designate the destination network (like Stanford), "Class B" networks with a 16-bit network address, and "Class C" networks with a 24-bit network address. The 32-bit IP address divided into the network address and the node address, thus Class A networks had 24 bits worth of addresses available for the site topology, and Class C networks had 8 bits available. Larger sites needing more addresses would be assigned a Class A or B block and smaller sites would be assigned a Class C block.

ISPs didn't exist back then, so each site maintained its connection to the Internet by keeping a direct link to another connected institution. The global routing tables only contained one route per institution, which was expected to be a total of a few hundred or thousand routes at most. There was no concern that the address space would be depleted; after all, the addressing scheme allowed for about 60 large institutions with Class A networks and about 65000 smaller institutions with Class B networks. For a network that was only

intended to be used by universities and government researchers in the United States, surely this would be enough!

## 2.2. The Internet today

Obviously, the Internet has grown larger than what the original designers expected. In the U.S. alone, virtually every educational institution, government office, and business has Internet access, as do most homes with a personal computer. Sites no longer have direct connections to each other but instead buy their Internet service from an ISP, who in turn pays for connectivity to a national or international backbone provider. Internet routing is no longer a web of connections, but more of a hierarchy, with the backbones on top, ISPs in the middle, and end users at the bottom. Keeping track of address assignments and routing information on the Internet has become a logistical nightmare.

As a result of the explosive growth of the Internet, sites were no longer allowed to be assigned large blocks of addresses. To allow for more granularity in address assignments, the Class system was replaced with Classless Interdomain Routing (CIDR) which allows address blocks to be assigned on any bit boundary.

## 2.3. Classless Interdomain Routing

One benefit of CIDR is that address blocks can be assigned in sizes that more accurately reflect a site's expected usage, but the more important effect is that the Internet routing architecture can now take advantage of the hierarchical structure of the Internet. Larger address blocks (around 16 bits) are assigned to ISPs, who break down their allocation into smaller blocks (18-24 bits) which they assign to their customers' sites.

As a result, the networks of many customers of an ISP can be aggregated into a single routing entry (a method called "route aggregation") in the global routing tables, since all packets to those networks are routed through that ISP, and only the ISP needs to know more specific routes for each customer. Thus, global routing tables need only contain entries for the address blocks of backbones and large ISPs. In practice, however, multihomed sites (those with service from more than one ISP) each need to have one global route per ISP. Even with CIDR and route aggregation, global routing tables still contain over 100,000 entries!

While the introduction of CIDR prevented routing tables from becoming unmanageably large and allowed the address space to be assigned in a more conservative manner, it created a different IP address problem.

### 2.4. The pain of renumbering

Since IP address blocks were initially intended to be assigned to sites rather than to ISPs, IP addresses assigned to individual computers and network devices were not expected to change often, if ever. Under CIDR, IP addresses used by a site must be allocated from the block owned by their ISP. This means that if a site switches their Internet service to a different ISP, they cannot continue to use the same IP addresses and must use new addresses assigned by the new ISP. Unfortunately, IPv4 wasn't designed to accommodate IP address changes, and so most TCP/IP stacks need to be reconfigured manually to change the IP address of a device, which often requires restarting the device. Changing ISPs can be a huge administrative headache for many sites.

## 3. Architectural changes in IPv6

It was clear that a radical change needed to take place to ensure future growth of the Internet. IPv6 was designed with the goal of aggressive address aggregation, ideally restricting the global routing tables to contain only around 8,000 entries -- a tenth of today's size -- at the worst. In addition, IPv6 devices are required to automatically configure themselves for new IP addresses (a process called renumbering) for a seamless and low-overhead transition to a new ISP. Procedures for allocating address blocks for new IPv6 addresses are very conservative to reduce wasted space.

### 3.1. Benefits of aggregation

Route aggregation works well when each address can only be reached via one path, but multihomed sites with more than one ISP can be reached by multiple routes. In IPv4, routes to such sites need to be propagated through the global routing tables to ensure optimal, fault-tolerant routing. This practice is a contributor to the large size of the global routing tables and thus has been prohibited in IPv6.

Through aggregation, IPv6 eliminates site-specific routes from global routing tables, so multihomed sites must be supported through other means. Two methods are available for multihomed sites to receive service from all of their providers.

### 3.2. Multihoming in IPv6

A site may chose to designate one of their providers as a "primary" provider, and use only an IP address allocation from that provider. The remaining provider would accept the site's traffic directly from the primary, for load-sharing and to provide an alternate route to

the site in the event of a link outage between the site and the primary. Outbound traffic could be sent from the site to either provider. This method requires that all providers to a multihomed site work together to provide service to the site.

Alternatively, multihomed sites can be assigned an address block from each provider, and every device at the site would have one IP address per provider. For example, if a site obtains service from three ISPs, each device will be assigned three IP addresses to use. Optimal routing is achieved by using a complex set of source address selection rules for each outgoing connection. Fault tolerance is achieved by listing all the IP addresses for servers in DNS records, allowing clients outside the network to attempt connections to each address until successful.

Each method has disadvantages, and multihoming in IPv6 is still a subject of discussion for the IETF.

### 3.3. Other IPv6 features

In addition to enhanced addressing capabilities, IPv6 also incorporates a number of extensions that have been developed for IPv4. One of the more appealing IPv4 extensions now standard in IPv6 is multicasting, which allows a data stream to be sent to the network only once and received by any number of interested hosts. Hopefully, multicasting will enable rich multimedia content to be streamed live over the Internet without the need for the massive bandwidth currently necessary to send one copy of the data to each recipient. Other standard IPv6 features include automatic configuration, a Quality of Service mechanism, and encryption capabilities (with IPSEC). IPv6 also has a capability to attach arbitrary header data to IP packets, allowing for extensions that may be developed in the future.

## 4. Basic IPv6 address format

The IPv6 addressing architecture is described in [RFC-2373].

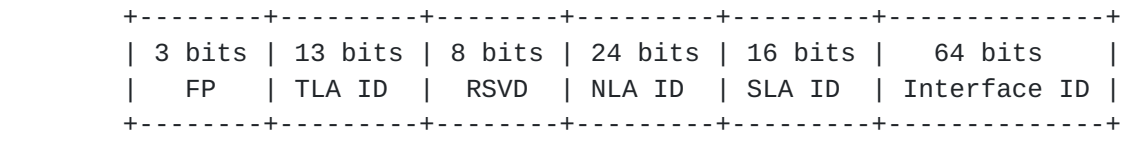
### 4.1. String format of addresses

IPv6 extends the address size from 32 bits to 128 bits. Addresses are represented as hex numbers, with colons between every 16-bit chunk, e.g. 3ffe:23:9091:7e2:1:ff3b:c0a:11a0. Long strings of chunks of value zero can be replaced with a double colon, although only one double-colon can be present in an address. For example, 3ffe:23:9091:0:0:0:0:1 can be written as 3ffe:23:9091::1. When writing IPv6 addresses in URLs, the address must be enclosed in square brackets, as per [RFC-2732]. For example, if the web server

WWW.DOM.AIN had the fictional IP address listed above, the site http://www.dom.ain/ would also be accessable at the URL http://[3ffe:23:9091::1]/. Note that the use of IPv6 addresses in URLs and other location identifiers is strongly discouraged, since IP addresses are expected to change often. Domain names should be used for this purpose instead.

## 4.2. Components of an IPv6 address

Like IPv4 addresses, IPv6 addresses are broken into a network component and a node component. IPv6 takes the process one step further and breaks the network component into several pieces, each of which is used for routing at a different level of the Internet. [RFC-2374] defines a global aggregatable address format for IPv6 unicast addresses. Here is the format of the 128 bits of IPv6 addresses, as described by that document:




FP is the 3-bit format prefix, which defines the type of address. Addresses with the FP bits set to 001 will always follow the format described here.

The 13-bit Top-Level Aggregation (TLA) identifier identifies the exchange or backbone provider that routes that address from the top level of the Internet -- so-called "default-free" zones.

The 8-bit Reserved field (RSVD) should always be set to zero for now, at least in global unicast addresses. In the future, these bits will be used to expand the TLA or NLA fields as necessary.

The 24-bit Next-Level Aggregation (NLA) identifier is split into further aggregation fields and a site identifier, as seen fit by each organization assigned a TLA ID. Portions of the NLA space may be suballocated to other organizations by the TLA holder. However, the top 48 bits of an IPv6 address must uniquely and fully identify a single site. This hard restriction on the site prefix boundary provides consistency between sites and allows organizations to switch ISPs without having to reorganize their site's topology.

The 16-bit Site-Level Aggregation (SLA) identifier identifies individual subnets within an organization. Since the 16-bit length of the SLA identifier allows for 65,536 different subnets, it is unlikely that many organizations would need more than a single 48-bit prefix for a site.

The 64-bit Interface ID, unique to each host on a network, can either be set manually or can be automatically derived from the host's hardware address.

### 4.3. Address configuration and assignment

IPv6 has several techniques for configuring the IP addresses of each host. Of course, network administrators can manually assign interface IDs, and while this may be desirable for servers, it quickly becomes tedious when configuring and maintaining workstations. Thus, several methods of automatic configuration are available.

One method of automatic configuration is DHCPv6, which is specified in [DHCPv6], operates very similarly to DHCP in IPv4.

IPv6 also allows for stateless automatic configuration for selecting interface IDs, allowing network devices to configure themselves on the network using only advertisement messages from routers as described in section 6 on Neighbor Discovery. Devices learn the network prefix(es) from the routers on the network, then use a globally-unique interface ID in IEEE EUI-64 format which is usually generated from the hardware address of the network interface hardware. The EUI-64 format is described in detail in [RFC-2373].

### 4.4. Temporary addresses for privacy

Because of privacy concerns involved in using a globally-unique identifier in an IP address, [RFC-3041] suggests the use of a secondary IP address containing a randomly-generated bitstring as the interface ID. Outgoing connections to public services use the random IP address, and incoming connections use the IP address obtained through manual or automatic configuration as described above. At this point, the privacy mechanism is not yet implemented, but it could potentially offer a greater degree of privacy than current IPv4 address-assignment methods depending on the topology of the intervening network.

## 5. Other address conventions

### 5.1. Address prefixes

Prefixes are used to define blocks of network addresses. They are written as an IP address followed by a slash and a prefix length, which indicates the number of bits which identify the network. This is similar to IPv4 CIDR notation. Since a double-colon (::) can be used to replace a number of zeros in an address, prefixes often end in :: to fill the unused address bits in a prefix with zeros. For

example, 3ffe::/16 describes all addresses beginning with the 16-bit string 3ffe and 2001:10::/48 describes all addresses beginning with the 48-bit string 2001:0010:0000.

### 5.2. Non-global addresses

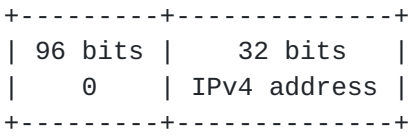
Two prefixes are set aside for link-local and site-local addresses. Link-Local Addresses, in prefix fe80::/64, are valid and unique only on the local network directly connected to each interface card (usually the local Ethernet segment) and are never routed. They are automatically assigned to every interface and are primarily used to obtain configuration information. Site-Local Addresses, in prefix fec0::/48, may be used however a site sees fit, and the IP addresses may be assigned to networks just like any global address allocation. Site-local addresses are never routed onto the Internet.

### 5.3. The loopback address

The special address ::1 is the loopback address, which always points to the local machine. It is analogous to the 127.0.0.1 address in IPv4.

### 5.4. IPv4-compatible addresses

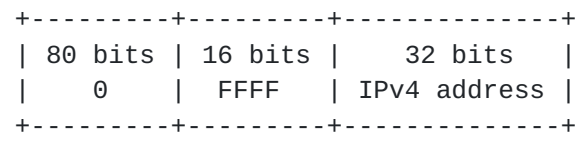
Every IPv4 address can be written in IPv6 notation by appending 96 bits of zeros to the beginning of the address, extending the address size to 128 bits, as shown here:




These addresses are IPv4-compatible IPv6 addresses, formed by appending a double-colon to the beginning of a standard IPv4 address, like ::10.9.8.1. (This syntax is used only for convenience, and the example address is equivalent to ::a09:801.) These addresses are used in IPv4 tunneling mechanisms as described in section 7 on Transition Mechanisms.

### 5.5. IPv4-mapped addresses

[RFC-2373] also delineates the notion of IPv4-mapped addresses, which allow the addresses of IPv4-only hosts to be represented in IPv6 notation. The format of these addresses is 80 bits of zeros, followed by 16 bits of ones, followed by the 32-bit IPv4 address.




Such addresses are written like ::ffff:192.168.0.1. IPv4-mapped addresses should NEVER be used in IPv6 headers, nor entered in DNS records. They exist so that TCP/IP stacks which support both IPv4 and IPv6 can store the addresses of all hosts in an IPv6 address field, even hosts that only have IPv4 addresses.

IPv4-compatible addresses are distinct from IPv6-mapped addresses. IPv4-compatible addresses are used to represent hosts with IPv6 capability but are only reachable by IPv4, so an IPv6-in-IPv4 tunneling mechanism must be used to communicate with these hosts using IPv6. IPv4-mapped addresses are used internally inside a network device to address hosts that only have IPv4 capability, and thus these addresses are only useful to devices with both IPv4 and IPv6 connectivity. The utility of IPv4-mapped addresses will be discussed in section 10 about Programming and the API.

### 5.6. Multicast and Anycast

The address format above describes unicast addresses -- that is, addresses that specify a destination of exactly one distinct network interface. In addition, two other types of destinations may be specified for IPv6 packets. Multicast, as described earlier in this document, is treated much the same way as IPv4. Multicast group addresses are assigned to organizations from the prefix ff00::/8. Anycast is a special mode of unicast that allows a number of nodes to listen to the same IP address with the expectation that a packet sent to that address will arrive at exactly one of the listening nodes. Anycast addresses are allocated by each organization from their unicast address allocation.

## 6. The Neighbor Discovery Protocol

The mechanism used by IPv6 to automatically configure hosts new to a network is the Neighbor Discovery (ND) protocol, defined in [RFC-2461]. ND is a subset of ICMP, and provides functionality necessary for a host to automatically discover parameters about the local subnet (local network number, router addresses) and about other hosts on the network and how to reach them (hardware addresses of neighbors, duplicate address detection). It subsumes the functionality of the IPv4 ARP protocol and provides some of the functionality of address configuration protocols such as BOOTP and DHCP. In fact, a network device connecting to a network for the first time can learn all parameters necessary to function solely

through ND information.

Since multicasting has replaced broadcasting in IPv6, ND packets destined for some or all hosts on the local network are sent to a link-local multicast address. The address ff02::1 reaches all devices on a local network, and the address ff02::2 reaches all routers on the local network. Certain ND packets may also be destined for a single host, in which case they are unicast to that host's IP address.

Both routers and hosts advertise their presence using router advertisement and neighbor advertisement messages, respectively. Advertisement messages are sent to all hosts periodically, or to a single host when requested by a router solicitation or neighbor solicitation message.

Router advertisement messages contain information about the local network, such as local network number and maximum MTU, as well as information about the routes that the router is capable of reaching.

Neighbor advertisement messages are primarily used to announce the hardware address of a particular host and to detect duplicate IP addresses.

## 7. Transition mechanisms

Until IPv6 becomes natively available everywhere, sites will have to use transition mechanisms to allow their hosts to communicate with other sites that also IPv6 but are only reachable via an IPv4-only network. Currently, the most common tunneling protocol is called SIT (for Simple IP Transition) which tunnels IPv6 inside IPv4 packets to allow IPv6-aware sites to communicate over the IPv4-only Internet. A number of different mechanisms use SIT as a transport for both manually-configured and automatically-configured tunnels, many of which are described in [RFC-2893].

### 7.1. The 6bone

The initial testbed for the IPv6 protocol is called the 6bone. Sites experimenting with IPv6 have set up manually-configured SIT tunnels to each other, creating the first world-wide IPv6 "network". The TLA 3ffe::/16 has been assigned to sites on the 6bone by [RFC-2471]. Single users interested in participating in the 6bone can visit any one of a number of "tunnel brokers" that will automatically set up a static SIT tunnel. Sites needing to connect more than one device or devices without direct IPv4 access should contact another 6bone site near them to configure a tunnel and obtain an IP allocation inside the 6bone TLA. Lately, however, other transition mechanisms -- ones

that allow 6bone access (unicast-only) without having a direct 6bone tunnel -- have become popular.

### 7.2. Direct tunnelling

Hosts with public IPv4 addresses have, of course, IPv4-compatible IPv6 addresses as described in section 5.4. With minimal configuration, these hosts can easily communicate with other hosts having IPv4-compatible addresses, provided the hosts involved can route IPv4 packets to each other. Communication is done using automatic SIT tunnels. Multicast isn't supported by this method.

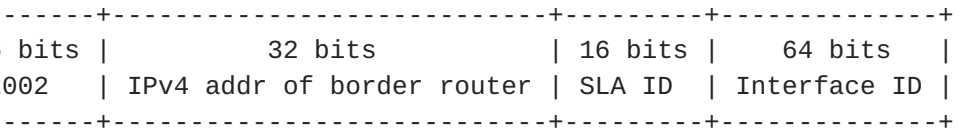
### 7.3. 6to4

The TLA 2002::/16 has been reserved for a protocol known as 6to4, specified in [RFC-3056], which is another method for creating automatic IPv6 tunnels over IPv4 with SIT. By embedding the IPv4 address of a 6to4 router into the NLA and reserved fields, an entire /48 network can be addressed behind every public IPv4 address. Sites with just one static IPv4 address for their border router can assign IPv6 addresses to all their network devices without needing native IPv6 ISP service, manually-configured tunnels, or an official IPv6 address allocation. Routers utilizing 6to4 can route packets to 6to4 addresses by extracting the IPv4 address of the border router from the destination address, then tunneling the packet using SIT to the 6to4 router at the destination site, which will forward the packet to the local destination via native IPv6.

Unfortunately, since public routing of 6to4 addresses is done by the IPv4 Internet, multicast isn't supported, but work is underway to provide such support.

#### 7.3.1. 6to4 addressing

This is the format for 6to4 addresses:




To find the 6to4 prefix for a particular IPv4 address, convert the address to hex and append it to the 2002 TLA. For example, the address 10.9.8.7 in hex is 0x0a090807, so the 6to4 prefix routed via this address is 2002:a09:807::/48. The SLA and Interface IDs can be allocated within a 6to4 prefix in the same manner as in any other /48 site address block.

### 7.3.1. Relaying 6to4 to the 6bone

Although sites utilizing 6to4 for IPv6 connectivity can automatically communicate with other 6to4 sites, some method is needed for 6to4 border routers to route to sites on the 6bone or native IPv6 links. Currently this is accomplished by configuring a tunnel from the border 6to4 router to a public 6to4 gateway router, all of which are listed on the 6bone website [6BONE]. [6to4ANYCAST] specifies an "IPv4 anycast address" which would automatically be routed to the nearest public 6to4 gateway, allowing all 6to4 border routers to tunnel non-6to4 IPv6 packets to a single address, greatly simplifying 6to4 configuration and ensuring optimal routing for tunneled IPv6 packets. As of this revision of this document, the specification is pending IANA approval.

## 7.4. Other transition mechanisms

There are other tunneling methods in development that allow more flexibility and some support for multicast. One of these is 6-over-4 (described in [RFC-2529]), but it requires IPv4 multicast support on the underlying network, which isn't available at most sites.

# 8. TCP and UDP

TCP and UDP are transport protocols that work on top of the IP layer. Thus, when the IP layer is upgraded (say, from IPv4 to IPv6,) the TCP and UDP layers should need only minor changes. And indeed this is true; they continue to operate almost exactly the same way under IPv6 as under IPv4, with only a few modifications to accomodate the new IP layer. Services like HTTP and SMTP still operate on ports 80 and 25, respectively, using almost exactly the same socket API calls as they did before. As a result, most programs need few changes to become IPv6-compatible, and very little consideration needs to be applied to retain IPv4 compatibility as well.

With most IPv6-compatible TCP/IP stacks, the programmer has the option of allowing a server to listen for connections on a port from both IPv4 addresses and IPv6 addresses simultaneously. Networking code can differentiate between incoming connections by checking whether the peer's address is an IPv4-mapped address or a real IPv6 address.

Specific changes to the BSD socket API are covered in section 10 on Programming and the API.

# 9. DNS

Several new DNS record types have been created by [RFC-1886] for IPv6

addresses. Just as IPv4 addresses use A records for forward resolution (name-to-address) and PTR records for reverse resolution (address-to-name), IPv6 records have AAAA records for forward resolution and PTR records for reverse resolution. These records are defined by [RFC-1886]. AAAA records contain a full IPv6 address as the data value. PTR records are keyed by reversing the IPv6 address, putting periods between every hex digit, and appending .IP6.INT.

For example, if the IPv4/IPv6 server SERVER1.DOM.AIM with IPv6 address 3ffe:821:9982:14::1 and IPv4 address 192.168.0.1 was the authoritative DNS server and mail server for the domain DOM.AIN, the forward and reverse lookup zone files would be similar to those shown below. Note that both A and AAAA records are present to support both hosts using IPv4 and IPv6.

```
; partial zone file for DOM.AIN
DOM.AIN.                          NS         SERVER1.DOM.AIN.
DOM.AIN.                          MX    10   SERVER1.DOM.AIN.
SERVER1                           AAAA       3ffe:0821:9982:14::1
SERVER1                           A          192.168.0.1

; partial zone file for 2.8.9.9.1.2.8.0.e.f.f.3.IP6.INT
1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.4.1.0.0   PTR    SERVER1.DOM.AIN.
```

[RFC-2874] defines new record types, A6 and DNAME, that support renumbering through indirection, but these new types are not implemented in many IPv6 stacks and there is some discussion about whether use of these records would jeopardize the stability of the DNS system.

## 10. Programming and the API

The BSD Sockets API is the standard IP network programming API used on Unix and other systems. [RFC-2553] specifies extensions that are made to existing sockets APIs in order to accommodate the use of IPv6.

New constants, PF_INET6 and AF_INET6, have been defined to identify the IPv6 protocol family and address family in calls to socket() and for use in sockaddr structs. A new form of the sockaddr struct, sockaddr_in6, has been defined to store address, port, and option information for IPv6 socket endpoints. New functionality for providing name and address resolution (primarily via DNS) has been defined also. The gethostbyname() function has been extended to allow resolution of IPv6 addresses, but its use should be deprecated in favor of a newer, protocol-independent function getaddrinfo(), which is defined in the POSIX 1003.1g specification and borrowed by [RFC-2553].

Programs that exclusively use PF_INET6 sockets can still communicate with IPv4 hosts, provided that the program runs on a system that supports both protocols. Connections to IPv4-mapped addresses from IPv6 sockets will be made using IPv4, even though the sockets can be created and manipulated with IPv6-specific API calls. To extend the compatibility to address resolution, the gethostbyname() and getaddrinfo() functions are capable of returning addresses of IPv4-only hosts as IPv4-mapped addresses, allowing programs to remain completely compatible with IPv4 hosts when exclusively using the new IPv6 API.

## 11. Data security and IPsec

As in IPv4, IP-level data security is accomplished in IPv6 using IPsec. Unlike in IPv4, however, IPv6 defines IPsec as a required feature of the IP stack.

The goal of IPsec, as in most data security systems, is to provide confidentiality to prevent packets from being viewed except by the receiving host, and to provide authentication and integrity to guarentee that the data in a packet is authentic and from the correct sender.

### 11.1. AH and ESP

[RFC-2401], which gives an overview of IPsec, divides IPsec into two main components: the Authentication Header (AH) defined in [RFC-2402], and the Encapsulating Security Payload (ESP) defined in [RFC-2406]. The AH is used to digitally sign packets, and the ESP is used to encrypt and optionally sign packets. Although both AH and ESP allow any cryptographic algorithms to be used, the IPsec standard requires that all implementations support HMAC-MD5 for AH and DES for ESP.

### 11.2. Modes of use

IPsec can be used in either host-to-host mode or gateway-to-gateway mode. Host-to-host mode is intended for use between two hosts that need to secure communications between themselves. Gateway-to-gateway mode is used by border routers to secure communications between specific networks. Usually this is used to create a Virtual Private Network (VPN) between branch offices of an organization over an insecure network.

ESP has two distinct modes of operation to support host-to-host and gateway-to-gateway communications. Transport Mode is used between hosts and only encrypts the transport data and payload of each packet, leaving the IP headers unencrypted. Tunnel Mode is used by

gateways and encrypts the whole packet, including the IP header, then sends out the encrypted packet as the payload of a new packet with the source and destination addresses of the gateways. Tunnel mode reveals less information about the packet, but can only be used to encrypt communications between cooperative gateways. Transport mode only hides the payload of each packet, leaving the headers exposed, but can be used between arbitrary hosts without depending on the network topology.

### 11.3. IPsec on the Internet

The eventual goal of IPsec is the wide availability of opportunistic encryption, in which hosts which have never communicated before are able to automatically obtain encryption keys for each other and use ESP and AH to secure the data that they exchange. Although many implementations of IPsec under both IPv4 and IPv6 are available for many platforms, there is not yet a secure public key infrastructure to distribute keys across the Internet. Thus, while IPsec is used to create VPNs within an organization and to connect isolated hosts to a secure network, opportunistic encryption is still not feasible today.

## 12. Quality of Service in IPv6

[To be completed]

## 13. Mobility support in IPv6

[To be completed]

## Author's Address

Nathan Lutchansky
P.O. Box 33164
Juneau, AK 99803-3164

Phone: (907) 780-4670
EMail: lutchann@litech.org

## References

6BONE The 6bone Website, http://www.6bone.net/.

6to4ANYCAST Huitema, C., "An anycast prefix for 6to4 relay routers", Work in Progress.

DHCPv6 Bound, J., Carney, M., Perkins, C. and R. Droms, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", Work in Progress.

RFC-1886 Thomson, S. and C. Huitema, "DNS Extensions to support IP version 6", RFC 1886, December 1995.

RFC-2373 Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 2373, July 1998.

RFC-2374 Hinden, R., O'Dell, M. and S. Deering, "An IPv6 Aggregatable Global Unicast Address Format", RFC 2374, July 1998.

RFC-2401 Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

RFC-2402 Kent, S. and R. Atkinson, "IP Authentication Header", RFC 2402, November 1998.

RFC-2406 Kent, S. and R. Atkinson, "IP Encapsulating Security Protocol (ESP)", RFC 2406, November 1998.

RFC-2461 Narten, T., Nordmark, E. and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, December 1998.

RFC-2471 Hinden, R., Fink, R. and J. Postel, "IPv6 Testing Address Allocation", RFC 2471, December 1998.

RFC-2529 Carpenter, B. and C. Jung, "Transmission of IPv6 over IPv4 Domains without Explicit Tunnels", RFC 2529, March 1999.

RFC-2553 Gilligan, R., Thomson, S., Bound, J., and W. Stevens, "Basic Socket Interface Extensions for IPv6", RFC 2553, March 1999.

RFC-2732 Hinden, R., Carpenter B. and L. Masinter, "Format for Literal IPv6 Addresses in URL's", RFC 2732, December 1999.

RFC-2874 Crawford, M. and C. Huitema, "DNS Extensions to Support IPv6 Address Aggregation and Renumbering", RFC 2874, July 2000.

RFC-2893 Gilligan, R. and E. Nordmark, "Transition Mechanisms for IPv6 Hosts and Routers", RFC 2893, August 2000.

RFC-3041 Narten, T. and R. Draves, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 3041, January 2001.

RFC-3056 Carpenter, B. and K. Moore, "Connection of IPv6 Domains via IPv4 Clouds", RFC 3056, February 2001.

Full Copyright Statement