

# Security, Safety and the Acceptance of IPv6

George V. Neville-Neil  
Member of The FreeBSD Project

## ABSTRACT

In order for a network protocol to gain acceptance in the broader Internet community it is necessary for that protocol to be deployed on enough systems and used by enough people to show that it is both beneficial and also not a danger to the rest of the network as a whole. Gaining acceptance for a new protocol is therefore a chicken and egg problem, and solving it requires that protocol designers and developers consider the issues of security and safety earlier in the development process. In this paper I will discuss some problems faced in deploying IPv6 that relate to the perceived safety and security of the protocol, and give a specific example from the FreeBSD project in which features of IPv6 were turned off for safety reasons.

## Categories and Subject Descriptors

C.2.0 [Computer Systems Organization]: Computer Communication Networks

## General Terms

Design, Standardization

## Keywords

IPv6, Security

## 1. INTRODUCTION

One of the stated goals of IPv6 has been to provide a set of network protocols which, from the very beginning, have security, including authentication and privacy, as main features[3].

IPv6 also tries to satisfy another, and diametrically opposing goal, which is convenience, as embodied in the now famous “Dentist’s Office” problem. The idea behind the “Dentist’s Office” problem is that a small office should be capable of setting up their systems on a network, with little or no networking knowledge. The features that make setting up

the network easy, specifically link local addresses and the auto-configuration protocols, can also make attacking systems that use these features easier in some, now common, situations.

One complicating factor has been that since the time when the first IPv6 recommendations appeared in 1995, the Internet has grown from 16 million users, with little or no commerce, to over a billion users, with billions of dollars in commerce carried out across the network. Laptops and wireless networks have now become ubiquitous, which means that an attacker need not be physically present in order to mount an attack. Activities such as war driving or using high gain antennas to gain access to a network are now commonplace and must be taken into account when designing and deploying network protocols. These factors all come to bear on how next generation protocols such as IPv6 are designed, built, and ultimately deployed.

One last, but by no means insignificant, factor is that unlike most other protocols, IPv6 is an *infrastructure protocol*. What is meant by an infrastructure protocol is that IPv6 provides a complete system for implementing a packet switched Internet. The problem for IPv6 is that there already is an existing packet switched Internet, based on its predecessor, IPv4, even though several previous dire predictions would have us believe that by now the IPv4 Internet would be both unmanageable and unusable.

This paper uses two terms, safety, and security, to discuss two different ideas. The security of a protocol is a measure of how well the protocol can deliver data without an attacker being able to manipulate the data or to masquerade as someone else in the network. To provide security, that is authentication and encryption of data, IPv6 mandates support for the IPsec family of protocols[11]. The safety of a protocol is a measure of how much risk enabling the software that provides it poses to the rest of the system. Adding a large piece of software to any system is a risk, but those risks need to be mitigated as much as possible before the subsystem is added in. Most of the concerns discussed in this paper relate more to safety than to security.

The easiest way to understand the problems relating to safety and the deployment of IPv6 is with a metaphor. It is as if we have two bridges, one old, and a bit rusty, and one shiny and new, that both go to the same place. The old bridge is crowded and a bit dangerous but people have used it every day for years and know generally what the hazards are. The new bridge is untested but is due to open soon. It is wider, faster and its builders say that it’s far safer than the old bridge. If, on or before the official opening day for

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

IPv6’07, August 31, 2007, Kyoto, Japan.

Copyright 2007 ACM 978-1-59593-790-2/07/0008 ...\$5.00.

the new bridge, a news report appears claiming that the new bridge was built without enough steel, or that cracks have already appeared in the concrete piers that support it, it is unlikely that people are going to want to switch. What will happen is there will be a large amount of bad publicity, investigations and delays. It may take years for people to trust the new bridge enough to use it. It is this same situation in which we find ourselves with IPv6. It has taken many years to convince people to put these protocols into their systems. Routers and end station operating systems had to be upgraded and tested and service providers have to be convinced, or at least see a good reason, to deploy the new protocols. We are no longer in a situation where the Internet could be upgraded by having a “flag day.”[5]

Network protocols can no longer be designed such that safety and security considerations come later. In order for a network protocol to be accepted and therefore deployed in the wider community it must always assume a hostile environment. Any protocol that does not assume a hostile environment will remain balkanized from the rest of the network and therefore will never garner enough usage or users to be accepted into the global Internet.

The challenges faced by those of us who wish to deploy new protocols are both technical and social. Much has been written about the technical challenges of providing a safe and secure set of network protocols, but little has been done about the social challenges. Although this paper is technical in nature some of the social challenges will be addressed as well.

This paper is broken into two major sections, Section 2 which covers the technical problems that were addressed and Section 3 which covers the social problems that are involved in convincing people that a protocol is sufficiently secure to be deployed. A final, smaller Section 4 gives our conclusions and recommendations to people trying to deploy IPv6 as well as future protocol authors.

## 2. A PARTICULAR CASE

In this section I will explain a specific safety concern that was addressed in FreeBSD’s support for IPv6 and how some basic requirements in the IPv6 protocol proved to be considered too risky to leave running as part of a default installation. Although this paper is intended for an IPv6 savvy audience a brief introduction to the specific part of IPv6 that caused the problem is given as well as an introduction to how IPv6 is integrated into FreeBSD.

### 2.1 IPv6 and Link-Local Addresses

One goal of the designers of the IPv6 protocols was to make setting up a system that ran the new protocols easier than it was to set up a system running IPv4. Until the Dynamic Host Configuration Protocol (DHCP) [10] became widely deployed users had to know several pieces of seemingly esoteric information, such as their subnet, their host’s network layer address, and the network layer address of their default router. When most computers were set up by specialists, such as engineers and systems administrators the fact that these pieces of information were esoteric was not a problem, as it was the job of those people setting up the network to be familiar with how it worked.

IPv6 attempted to do away with the user needing to know any of this information by specifically including protocols for the auto-configuration of nodes in the network. A node

can start running without a globally routable address, and talk to its neighbors to learn about the local network and eventually, with the help of a router, automatically boot strap itself into the global Internet.

In the IPv4 network, DHCP uses the relatively crude method of broadcasting packets between the node that is connecting to the network, and the server. Broadcasts are considered wasteful and to be avoided in network protocol design so IPv6 uses a combination of multicast addresses and link local IPv6 addresses assigned to nodes to make it so that broadcast packets were not needed. IPv6 uses link local addresses for auto-address configuration, neighbor discovery, or when there is no router present[4]. A link-local address is formed from some information about the link, a common case for Ethernet devices is to use the network interface’s hardware address to form an EUI-64 Interface ID. A node on a link can send a message to any other node on the same link by sending data to the target node’s link local address. It was the automatic availability of the link local address which was the source of the problem in this case.

### 2.2 FreeBSD and IPv6

FreeBSD is a widely used, open source, Unix operating system, derived from the Berkeley Software Distribution (BSD) operating systems developed at the University of California at Berkeley in the 1970s and 1980s[8]. Since February of 2000, FreeBSD has enabled support for IPv6 as a default feature of the operating system kernel. FreeBSD’s IPv6 code came from the Kame project, a part of the WIDE project[6].

In FreeBSD, and most other modern operating systems including other flavors of BSD, Mac OS/X, and Linux, there are two parts to configuring and running a complex subsystem like IPv6. The first is the support in the operating system, which is provided by software compiled into the kernel or provided by kernel modules which are loaded at system boot time. Kernel subsystems may also have their own, in-kernel, start up code which is executed when the system boots.

Support for IPv6 is provided in FreeBSD by a set of components that are compiled into the kernel when it is built. When we say that “support for IPv6 is enabled by default” this means that the IPv6 option, “INET6”, is part of the generic kernel configuration file (GENERIC), and that all shipping FreeBSD kernels, such as those installed from CDROM, DVD or built from source will have IPv6 software built in to them unless the user specifically removes the option and builds and installs a new kernel.

The second part of configuring a complex subsystem in FreeBSD is achieved using the `sysctl(3)`, sys control, library which sets and gets values in various parts of the kernel. Kernel variables can be modified at boot time as well as while the kernel is running. Certain special values can only be set when the kernel starts and cannot be modified in a running kernel. When the system boots, a set of scripts is run from the `/etc/` directory in order to configure the system correctly. The `rc` scripts do things such as bring up network interfaces, configure and start network protocols, and do other general housekeeping chores necessary to bring the system to a usable state. The start up scripts use the `sysctl(3)` library to configure the kernel as it boots and comes up to a usable state.

### 2.3 Report...

On September 3rd, 2006 the FreeBSD security team received an email from a developer pointing out the following facts:

- IPv6 is enabled by default in all FreeBSD kernels
- An IPv6 link-local address is assigned to each interface brought up on the system.
- Users configuring firewalls without knowing their system had IPv6 would not block IPv6 correctly leading to ports such as telnet being left open.

Running the `ifconfig(8)` program on a default system would show that there was at least one IPv6 address assigned to the interface even when the rest of IPv6 was not configured.<sup>1</sup>

Figure 1 clearly shows an IPv6 link local address assigned to the Ethernet interface named `ed0`. Although a link local address was assigned, most of the rest of the IPv6 setup was not complete at this point. In order to fully participate in an IPv6 network FreeBSD required that the `ipv6_enable` configuration variable be set in the system's `/etc/rc.conf` file. Without the `ipv6_enable` variable being set FreeBSD could do only very limited IPv6 communication, in effect the system had all the vulnerabilities of running IPv6 without any of the benefits.

Packets addressed to link-local addresses are not supposed to be forwarded off link by routers, so how would it be possible to exploit this particular situation? The answer comes from the first complicating factor laid out in Section 1, the wide deployment of wireless technology. By 2006, when this report was originally sent to the security team, all laptops sold included support for 802.11b, at the very least, and most included support for both 802.11b and g wireless networking.

The 802.11 family of protocols attempts to make wireless networking look as much like an Ethernet as possible, in order to leverage the work done in the last 30 years on wired networks. To an end node an 802.11 network interface looks very much like an Ethernet interface. An interface in an 802.11 network has a 6 byte hardware address, called an ether address, the network itself acts as a shared broadcast medium, which means that any node that is joined to the network can, with the proper software, see all unencrypted packets between all the nodes connected to the same wireless access point.

When a computer joins a wireless network the local link is defined as the network to which it has joined, also known as the Service Set Identifier or SSID. Internet Cafe's, airports, conferences, and now even whole cities deploy completely open or partially open wireless networks. A completely open network is just what it sounds like, a network that does not require the user to have a key to access it, or to which the key is well known, as at a conference. A partially open network, which is common in airports and other public places where users can be charged for network access, allows anyone to connect to the network but requires the user to authenticate in some way in order to access the global Internet. In both of these cases, fully open and partially open, a node that

<sup>1</sup>The examples given in this paper are all run in a virtual environment and hence contain no routable IPv4 or IPv6 addresses. The safety issues are only present when systems are connected to a real network.

connects to the network with a link local address enabled is open to attack from other nodes on the same wireless network. How would this attack proceed?

The piece of information that the attacker needs in this scenario is the node's link-local address. When a link local address is assigned in FreeBSD some packets are transmitted by the node even though it has not actually finished configuring IPv6. The node needs to join multicast groups and also perform Duplicate Address Detection (DAD) if DAD is enabled. When a node joins the IPv4 network it gets its address by using DHCP, and the DHCP traffic is transmitted in the clear. The attacker need only sniff packets from the wireless network, watch for new nodes as they join, and then get the nodes Ethernet address from the DHCP traffic sniffed from the network. With the Ethernet address in hand the attacker uses published algorithm to figure out the node's link local address[7]. From this point on the safety of the node depends on how well it defends against IPv6 attacks. The belief is that a user who does not know they have IPv6 in their system, which is most users, will not work to defend against attacks over IPv6. Writing a program to sit in a wireless network, collect Ethernet addresses and attack them is relatively straightforward, but has not been undertaken here.

## 2.4 ... and Remediation

When the report in Section 2.3 was first received by the FreeBSD security team the author was both a member of the security team as well as an active supporter and maintainer of IPv6 in FreeBSD. The report brought out a typical dilemma for network researchers, operating systems developers and security people. On the one hand researchers and developers want to see new and interesting protocols deployed, and so we recommend that they be added as default components to systems, so that turning them on, and trying them out, is as effortless and painless to the user as changing a single configuration variable. On the other hand, as a person responsible for the safe operation of a system, the tendency is to turn everything off that a user is not actively using. Systems and components that cannot execute usually cannot be the source of a safety problem. These two points are at the extremes of a spectrum and the mark of a good development group is the ability to find a point that is acceptable to the majority of users somewhere between convenience and security.

One of the goals of FreeBSD developers, and one often stated in discussions of this sort on our mailing lists is the Principle of Least Astonishment, POLA for short, which states that the system should act in a way that has the least surprising results for the largest set of users[9]. It was decided by the security team, after discussions with the Kame developers and internally that any system that did not explicitly and completely enable IPv6 through the use of the `ipv6_enable` configuration variable would not have link local addresses automatically assigned to its network interfaces when they were brought up. An extra configuration variable `auto_linklocal` was added so that it would be possible to deploy systems that acted as they had before.

## 3. THE SOCIAL PROBLEM

The technical problems laid out in Section 2.3 were, in themselves, not difficult to solve. The addition of an extra variable or two, and maybe some 10 or 20 lines of source

```
ed0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
    ether 00:72:31:5a:a9:7f
    inet 10.211.55.8 netmask 0xfffff00 broadcast 10.211.55.255
    inet6 fe80::272:31ff:fe5a:a97f%ed0 prefixlen 64 scopeid 0x1
    media: Ethernet autoselect (10baseT/UTP)
```

**Figure 1: Network Interface with Link-Local Address**

- People are afraid of the unknown.
- Being surprised by the unknown increases people's level of fear.
- POLA must be applied to the design and deployment of any system, especially network protocols, to keep the level of fear below the threshold where it will cause a user to turn a feature off.

**Figure 2: Three Observations**

code fixed the issue. The bigger lessons to be drawn from this issue are social, as opposed to technical, but they must influence how network protocol researchers act now and in the future. The three observations that come out of this incident are summarized in Figure 2. I will assume that most people can agree on the first two points and will expand on the third.

As was discussed in Section 1 deploying a new infrastructure protocol, such as IPv6, requires many actors for the deployment to be successful. Network hardware manufacturers and computer operating system developers must build, test and support the protocol in their products. ISPs must then provide a set of usable services with the new protocol. At the moment there are very few economic reasons driving the deployment of IPv6, and there are no services that users want which depend solely on the new protocol. Deployment must proceed for other reasons, related to management of the network and the provision of future, as yet undeveloped, services. In a market based system, such as the modern Internet, trying to deploy a protocol without an economic need is a tenuous proposition at best. Any reason that can be given not to deploy the new protocol will be used by those responsible for running the business that is the Internet as a way to not provide IPv6.

Computer and network security issues are in the news headlines daily, and these headlines drive the fears of users and decision makers who can make or break the acceptance of a new system. If a new protocol, such as a peer to peer protocol, provides a significant perceived gain for users, without the need for the users or service providers to make any significant changes to their system, then it will be immediately adopted. The fact that just about any traffic can now be tunnelled over port 80 (HTTP) should be sufficient to demonstrate this principle. IPv6 is not that type of protocol.

If IPv6 is to be successfully deployed it must not only demonstrate its abilities to technologists, through the provision of a much larger address space, smaller routing tables, auto-configuration, and the like, but it must also be safe and secure. It can be argued that the principle of least astonishment should be applied to all systems, but in the case of

a piece of software that is responsible for connecting computers together, safety is of the utmost importance. A loss of confidence in the safety of the next generation Internet protocol could set back its adoption by years, if not permanently.

## 4. CONCLUSIONS

It would be a mistake to neglect the IPv6 related security issue that was reported between the time that this paper was submitted to the conference and when it was accepted for publication. Although the details are too large to be addressed in this paper, they are addressed in the original work [2]. The security issues around route header zero (RH0) presented a larger danger to the IPv6 Internet than those presented here and have taken longer to address because they are protocol specification issues. The issues raised are currently being worked out both in code and in the specification process of the IETF and provide an interesting illustration of the principles described in this paper[1].

In this paper we have looked at a single IPv6 related safety incident. The incident itself is not nearly as significant as the lessons to be drawn from it. IPv6 was the last infrastructure level protocol to be designed while the Internet could be said to be a research network, where the majority of users were engineers and scientists and where there was little economic activity carried out on the network. In a way the fact that IPv6's design and deployment straddled the Internet's growth from a research to a commercial network has put IPv6 in the worst possible position. Designed for a less hostile world the protocol and its implementers must now face an array of dangers which were not imagined in the early 1990s.

As IPv6 continues to be deployed it becomes more important to look at what makes sense from the standpoint of a very hostile networking environment. Features and configurations will need to be adjusted, as they were in this case, so that they fit the POLA model described in this paper.

Another important contribution from the Internet Community would be a set of documents, like those on transition mechanisms, that describe how to set up nodes and networks so that they are as safe as possible.

For future protocol designers and implementers the rules of the game are now much clearer than they were before IPv6. Stated simply, we can no longer have "Security Considerations" sections of RFCs which say "Security issues are not discussed in this memo." The safety of any system we wish to deploy must always be our first concern.

## 5. REFERENCES

- [1] N.-N. G. Abley J., Savola P. Deprecation of type 0 routing headers in ipv6 (draft 01). Technical report, IETF, 2007.

- [2] A. Biondi, P. Ebalard. Ipv6 routing header security. Technical report, CanSec West, 2007.
- [3] H. R. Deering S. Rfc 2460: Internet protocol, version 6. Technical report, IETF, 1998.
- [4] D. S. Hinden R. Rfc 2373: Ip version 6 addressing architecture. Technical report, IETF, 1998.
- [5] G. Huston. Opinion: The mythology of ip version 6. *The Internet Protocol Journal*, 6(2), 2003.
- [6] S. K. Li Q., Jinmei T. *IPv6 Core Protocols Implementation*. Morgan Kaufmann, 2006.
- [7] C. M. Rfc 2464: Transmission of ipv6 packets over ethernet networks. Technical report, IETF, 1998.
- [8] N. N. G. McKusick M. *The Design and Implementation of the FreeBSD Operating System*. Addison Wesley Longman, 2004.
- [9] T. F. D. Project, editor. *FreeBSD Handbook*, chapter Glossary. The FreeBSD Project, 2007.
- [10] D. R. Rfc 2131: Dynamic host configuration protocol. Technical report, IETF, 1997.
- [11] K. S. Rfc 2401: Security architecture for the internet protocol. Technical report, IETF, 1998.