

IMPROVEMENT OF OHIO UNIVERSITY COMPUTER NETWORKS WITH  
INTERNET PROTOCOL VERSION 6

A Thesis Presented To

The Faculty of the

Fritz J. and Dolores H. Russ  
College of Engineering and Technology

Ohio University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Vitali Chipitsyn

June, 2000

THIS THESIS ENTITLED  
“Improvement of Ohio University Computer Networks with Internet Protocol  
Version 6”  
by Vitali Chipitsyn  
has been approved

for the School of Electrical Engineering and Computer Science  
and the Russ College of Engineering and Technology

---

Shawn D. Ostermann  
Associate Professor of Computer Science

---

Warren K. Wray, Dean  
Fritz J. and Dolores H. Russ  
College of Engineering and Technology

DISCARD THIS PAGE

## TABLE OF CONTENTS

	Page
LIST OF TABLES . . . . .	v
LIST OF FIGURES . . . . .	vi
1. INTRODUCTION . . . . .	1
1.1 Overview of IPv6 . . . . .	1
1.1.1 Large Address Space . . . . .	3
1.1.2 Route Aggregation and IPv6 Aggregatable Address . . . . .	3
1.1.3 Host Autoconfiguration . . . . .	4
1.1.4 Domain Name System . . . . .	5
1.2 Weaknesses of Ohio University Computer Networks . . . . .	6
1.2.1 Shortage of Available IP Address Space . . . . .	6
1.2.2 Deficiencies in Routing Topology . . . . .	7
1.2.3 Issues with Support of Dynamic Host Configuration . . . . .	7
1.3 Our Proposal . . . . .	8
2. TRANSITION TO IPV6 . . . . .	9
2.1 IPv6 Address Block and 6BONE . . . . .	10
2.2 Networks and Services . . . . .	12
3. OUR PROPOSAL AND ITS IMPLEMENTATION . . . . .	16
3.1 Proposed Addressing Hierarchy . . . . .	16
3.1.1 Overview of Network Topology . . . . .	17
3.1.2 Issues With The Existing Topology . . . . .	19
3.1.3 The Proposed Addressing Hierarchy . . . . .	22
3.2 IPv6 DNS Service . . . . .	35
3.2.1 Integration of IPv6 DNS into the Production Environment . . . . .	36
3.2.2 IPv6 DNS Test-Servers . . . . .	41

	Page
3.2.3 Configuration of the IPv6 Domain Name System . . . . .	43
3.3 End-User Support for IPv6 . . . . .	54
3.3.1 Software Available to The End-users . . . . .	55
3.3.2 Unix Operating System . . . . .	56
3.3.3 Microsoft Windows Operating System . . . . .	57
3.3.4 Macintosh Operating System . . . . .	57
4. CONCLUSIONS . . . . .	58
4.1 Improvements with Internet Protocol Version 6 . . . . .	58
4.2 Future Work . . . . .	59
BIBLIOGRAPHY . . . . .	61
 <b>APPENDIX</b>	
A. CONNECTION TO 6BONE . . . . .	63
B. IPV6 DNS MASTER FILES . . . . .	64
B.1 Master File for Zone ip6.int. . . . .	64
B.2 Master File for Zone ip6.ohiou.edu . . . . .	65
B.3 Master File for Zone cns.ohiou.edu . . . . .	67
B.4 Master File for Zone cs.ohiou.edu . . . . .	68
C. BIND-9.0.0B2 . . . . .	69
D. ADJUSTMENT OF DNS SOA RESOURCE RECORDS . . . . .	71
ABSTRACT . . . . .	74

## LIST OF TABLES

Table	Page
3.1 Compilation of Top-Level Aggregation Prefixes . . . . .	27
A.1 Measurements of Closeness of 6BONE pTLAs. . . . .	63

## LIST OF FIGURES

Figure	Page
1.1 The Structure of Aggregatable Global Unicast IPv6 Address . . . . .	4
2.1 Diagram of Test IPv6 Environment . . . . .	13
2.2 IPv6 Configuration of the University Core IPv6 Router . . . . .	14
2.3 IPv6 Configuration of the IRG IPv6 Router . . . . .	15
3.1 Layout of Ohio University Computer Networks . . . . .	17
3.2 Conceptual Diagram of the Proposed Addressing Hierarchy . . . . .	24
3.3 Allocation of the Top Portion of the Address Space . . . . .	26
3.4 The Addressing Hierarchy for the Main Campus . . . . .	28
3.5 The Addressing Hierarchy for a Regional Campus . . . . .	31
3.6 An Example Addressing Hierarchy for Student Networks . . . . .	33
3.7 Examples of Obsolete IPv6 DNS Resource Records . . . . .	37
3.8 Hierarchy of DNS Test-Servers . . . . .	44
3.9 Textual Representation of the A6 Resource Record . . . . .	48
3.10 Textual Representation of the DNAME Resource Record . . . . .	49
3.11 Assignment of DNS Names to Network Prefixes . . . . .	51
D.1 Original Contents of the SOA Resource Record . . . . .	72
D.2 Adjusted Contents of the SOA Resource Record . . . . .	73

## 1. INTRODUCTION

The Internet Protocol (IP) is a primary protocol of the TCP/IP protocol suite that is used to provide communication between machines on the Internet. IPv4 has been the basis for the Internet for two decades. Internet Protocol Version 6 (IPv6) has been designed as a replacement for IPv4 and is expected to gradually replace IPv4, with the two protocols coexisting for a number of years during the transition period. The first blocks of IPv6 addresses have already been allocated.

IPv6 improves many qualities of IPv4 and adds many new features. This chapter provides an overview of those features of IPv6 that appear to be the most appealing for improving computer networks. Also, in this chapter, we describe the potential weak points that we recognize in the computer networks of Ohio University, and suggest ways in which use of IPv6 may introduce constructive means of improvement.

### 1.1 Overview of IPv6

IPv6 is the next generation of Internet Protocol, and has been designed as a replacement for IPv4. The need for a more improved Internet Protocol introduced itself as shortcomings of IPv4 became more apparent. IPv6 solves a number of problems that have been found in IPv4, and extends functionality of the Internet Protocol as a whole. IPv6 is designed to improve scalability, security, ease of configuration, and network management of IPv4 [HD98]. At the present time, the Internet relies on a much wider set of functionality than people considered at the time when IPv4 was originally built. Before IPv6, any new functionality of the protocol would be carefully

built on top of the existing functionality. That would permit continuous extension of IP functionality in response to appearing demands. IPv6 integrates in one design the original IPv4 functionality and almost all added functionality. Also, capability of adding new functionality to the protocol has become a part of IPv6. This, coupled with combining all present functionality of IPv4 in one design, makes IPv6 more attractive for development in the existing and emerging computer networks.

Primarily, IPv6 has been designed to enable higher performance, scalable internetworks that would operate as needed for decades to come. IPv6 offers a number of enhanced protocol features, among which are larger address space [HD98], presence of host autoconfiguration protocols [TN98], and aggregatable structure of the IP address [HOD98] that permits an efficient, scalable routing hierarchy. In addition, IPv6 packet structure has been changed to simplify and regularize the packet structure, and to optimize packet processing time in the intermediate routers by moving all but absolutely necessary parameters to the optional fields [HD98]. Optional packet header information is transmitted in independent “extension headers” that come after the packet header and need not be examined by the intermediate routers. Moreover, network management capabilities of IPv6 allow automatic router discovery and automatic renumbering of routers and network nodes without human intervention. A more detailed overview of the improvements that IPv6 brings to networking in the Internet can be found in Section 1.3.

For testing of IPv6 as a new Internet Protocol, a worldwide IPv6 testing and preproduction deployment network, 6BONE, was established [Fin00]. The 6BONE may be thought of as a set of IPv6-enabled sites scattered in the sea of IPv4 networks. Such an architecture demonstrates that IPv6 can coexist with, as well as take advantage of, the existing IPv4 networks. By early 1998, the 6BONE reached approximately 400 sites and networks in 40 countries [Fin00]. The 6BONE has been built by an active population of protocols inventors, designers, and programmers to solve the questions that were arising during development of IPv6.

### 1.1.1 Large Address Space

Expansion of IP address space is a primary advantage of IPv6. Due to the fact that the size of the Internet constantly increases, IPv4 address space is being depleted quickly. There have been attempts [FLYV93] to extend availability of IPv4 addresses to some degree, but inevitability of complete exhaustion of IPv4 address space is now very obvious. IPv6 makes a successful attempt to effectively remove address space limitation by extending the length of an IP address from 32 bits to 128 bits. This extension provides a very large address space<sup>1</sup> that is approximately 1,500 IP addresses for each square meter of the surface of the planet Earth<sup>2</sup> [Hui94]. It is predicted that many hand held devices will soon require Internet connectivity. Even taking into consideration the potentially large number of such devices, the IPv6 address space should remain sufficient for a sensibly long period of time.

### 1.1.2 Route Aggregation and IPv6 Aggregatable Address

IPv6, with its larger address space, defines a multi-level, hierarchical, global routing architecture. IPv6 address space can be allocated in a way that facilitates route aggregation and controls expansion of routing tables in the core Internet routers. Use of IPv6 aggregatable global unicast address is the key that makes it possible. Aggregatable IPv6 addresses are organized into a three level hierarchy: *public topology*, *site topology*, and *interface identifier*. Public topology is the collection of providers who provide public Internet transit services. Site topology is local to a specific site or organization which does not provide public transit service to nodes outside of the site. Interface identifiers identify network hosts on links. The structure of aggregatable global unicast IPv6 address is depicted in Figure 1.1.

---

<sup>1</sup>The IPv6 address space accommodates  $3.4 \times 10^{38}$  different IPv6 addresses.

<sup>2</sup>According to [Hui94], the most pessimistic estimate would provide 1,564 addresses, while the most optimistic estimate would allow for  $3.9 \times 10^{18}$  addresses.

3	13	8	24	16	64 bits
FP	TLA ID	RES ID	NLA ID	SLA ID	Interface ID

Figure 1.1 The Structure of Aggregatable Global Unicast IPv6 Address.

This figure shows how the aggregatable global unicast IPv6 address is split into hierarchical levels. In the figure, FP – Format Prefix (001), TLA ID – Top-Level Aggregation Identifier, RES – Reserved for future use, NLA ID – Next-Level Aggregation Identifier, SLA ID – Site-Level Aggregation Identifier, INTERFACE ID – Interface Identifier. The public topology is comprised of FP, TLA ID, RES, and NLA ID. The site topology is represented by SLA ID.

We define route aggregation as collecting a group of network addresses with the same prefix and representing the routes to those networks as one route to a shorter network prefix. For example, all networks within a site – the networks that have a common SLA ID prefix – are advertised by site border routers to the NLA’s router as a single prefix. By the present time, the routing tables in the core Internet routers have become very large. This happened because, with IPv4, every single network must be listed in the routing tables of core Internet routers. The increase of address space with the introduction of IPv6 suggested that unless route aggregation is in place, the routing tables will become overloaded making routing impossible. Hence, IPv6 requires route aggregation as a standard routing policy. Such a requirement makes routing simpler and more efficient in most situations.

### 1.1.3 Host Autoconfiguration

A major component of network administration today is the assignment of networking parameters to network nodes, which permits the nodes to be used on the network. For IP, such parameters include at least the IP address of the node, the default router, and the DNS server. These parameters are either manually set by system administrators or automatically obtained from a server that runs Dynamic Host

Configuration Protocol (DHCP) [Dro97]. For networks with many nodes<sup>3</sup>, it may be relatively complicated to configure all nodes in a timely fashion or reliably maintain a DHCP server. Moreover, any change in the network configuration requires that the changes in the network parameters be propagated to the involved network nodes.

IPv6 provides several mechanisms for a network node to obtain its IP parameters; these mechanisms are referred to as host autoconfiguration. The node may use either stateless [TN98] or stateful host autoconfiguration to configure its network parameters. If the node uses stateless host autoconfiguration, it first generates an IPv6 address unique on the local link by combining the node's network interface identifier with the default prefix for link-local addresses that is defined in [HD98]. Then, if necessary, the node configures its own globally unique IPv6 address in cooperation with a local IPv6 router. If the node uses stateful host configuration, it has to locate a server that provides network configuration parameters consequently asking that server to provide the node with IPv6 configuration parameters. IPv6 implements a new version of DHCP, DHCPv6, for the networks that employ Dynamic Host Configuration Protocol to reduce network administration efforts.

#### 1.1.4 Domain Name System

We consider the Domain Name System because it is a crucial part of any networked system, as users generally refer to network nodes via node names, not IP addresses. Naturally, DNS has been extended to support IPv6. Initially, extensions to IPv6 DNS were developed only to store IPv6 addresses in DNS servers. The corresponding IPv6 resource record type is denoted with AAAA<sup>4</sup>. This extension is widely used and has worked reliably for a long period of time. Later, another type of resource record, A6, was introduced to provide support for the extended functionality of IPv6[CHT00]. We discuss A6 in Section 3.2.3.4.

---

<sup>3</sup>On the order of thousands, tens of thousand, and more.

<sup>4</sup>Pronounced "quad a".

A powerful feature of IPv6 DNS is the linking of IPv6 DNS to IPv6 host auto-configuration, coupled with secure DNS [Eas98]. Whenever a network node obtains a new IPv6 address, the node securely and automatically updates its DNS records with the new address. Such capability brings more convenience to the process of making changes to network configuration.

IPv6 DNS supports the renumberable and aggregatable IPv6 addressing hierarchy. This feature is in the latest specification of IPv6 DNS which is now “work in progress” waiting to be converted to a Request For Comments (RFC). New resource record types are introduced to the IPv6 DNS to implement this feature. However, the changes are still designed to be compatible with the existing application programming interfaces (API). The existing support for IPv4 addresses has been retained.

## 1.2 Weaknesses of Ohio University Computer Networks

As in almost any organization, the computer networks at Ohio University have certain weaknesses. While not being critical at the present time, the weaknesses may become problems in the future, unless they are eliminated. The following subsections analyse and describe these potential weaknesses.

### 1.2.1 Shortage of Available IP Address Space

It is conceived that IPv4 address space is being depleted, eventually relinquishing very few IPv4 addresses available. Consequently, is very unlikely that Ohio University will be able to obtain more IPv4 addresses in the future. The block of IPv4 addresses allocated to the university by IANA is limited to one class B subnet, which is 65,536 different addresses, and a few class C subnets that contain 255 addresses each. Certain portions of this address space cannot be used for assignment to network nodes, as they are being used to build the addressing hierarchy in the university [Hui94]. According to the existing usage of the address pool, only a few 254-node subnets remain in

reserve for future use. This does not leave much room for expansion, should it become necessary. Ultimately, when the university needs to build new networks or expand existing ones, no IP addresses will be available.

### 1.2.2 Deficiencies in Routing Topology

Administration and management of the routing topology may be improved. The university has many small networks, and the way those networks are topologically connected makes routing tables unnecessarily large. IPv4, by its nature, was not designed to take advantage of aggregation of IP addresses, although this was partially solved later with Classless Inter-Domain Routing (CIDR) [FLYV93]. Ohio University networks route traffic using the Routing Information Protocol (RIP) [Mal98]. Granularity of subnet assignment is typically 254-node subnets (24-bit network portion and 8-bit host portion), which gives a maximum of 255 different subnets. While this is not a large amount of routing information for a router to process, it is still relatively complicated to dependably manage the addressing hierarchy, especially if the hierarchy is not well-structured. If Ohio University computer networks are to be expanded, the severity of this problem will increase, as the only solution to expansion is production of higher granularity networks with longer network prefixes. To prevent this from happening, certain optimizations to, or redesign of, the current addressing scheme may be necessary.

### 1.2.3 Issues with Support of Dynamic Host Configuration

Reliability of dynamic host configuration techniques is a very critical issue for sites with a large network population. Ohio University is such a site. Clearly, it would be impossible for administrators to configure all networked hardware manually. To avoid manual configuration, Ohio University uses a DHCP server to provide dynamic host configuration and reduce network maintenance costs. A large proportion of the computers at Ohio University obtain their network configuration parameters from the

DHCP server. Because of this fact, failure of the DHCP server leaves a large portion of users without Internet connections. An accidental misconfiguration of the DHCP server could result in the server running out of available IP addresses rendering it unable to satisfy the requests of clients.

### 1.3 Our Proposal

We propose that IPv6 be used to fix the problems described in the section above. First, IPv6 easily solves the problem of insufficient IP address space. Second, use of route aggregation in IPv6 presents a good alternative to the current network topology at the university. A new addressing hierarchy for the network topology may be built using existing knowledge of needs and expectations for the entire network, upon the successful parts of topology, and the advancement against the known downfalls. Thirdly, introduction of a new networking protocol of such importance offers a good chance to introduce the new routing topology, as well as replace RIP with a more scalable routing protocol. Finally, host autoconfiguration capabilities of IPv6 – both stateless and stateful – can be employed to eliminate the risks of losing network access. Stateless host autoconfiguration will guarantee that any IPv6 node will always be able to obtain necessary network configuration parameters, while stateful host autoconfiguration (e.g., DHCPv6) may be used in special cases when network administrators desire to assign the nodes specific IP addresses.

This thesis describes the procedures that are imperative for applying IPv6 to the infrastructure of computer networks at Ohio University and provides relevant discussions. We describe what has been done or should be done to eliminate the weaknesses specified above to make improvements to the computer networks.

## 2. TRANSITION TO IPv6

The first step on the way to introducing improvements for the computer networks at Ohio University is to build and refine the IPv6 environment that would serve as the basis for future development. It is important to be familiar with the subtle details of running an IPv6 environment. It is also important to have a generalized scenario for transition to IPv6, so that this scenario could be used to simplify further expansion of IPv6 in the university. This chapter describes the IPv6 environment that we established, and how it was built.

Transition from IPv4 to IPv6 is split into several distinct subtasks. The first subtask is to obtain a block of IPv6 addresses to be used for the network. The second subtask is to prepare a router that would serve as a core IPv6 router for the network. Third, IPv6 networks are established and network nodes are configured with IPv6 support. Next, the IPv6 equivalents for the primary IPv4 services have to be placed in operation on the IPv6 networks to provide basic IPv6 support to IPv6-enabled hosts. Conclusively, it is necessary to establish a connection between the Ohio University IPv6 network and other IPv6 networks on the Internet.

The scenario described in the previous paragraph has been applied to the computer network of Ohio University. Section 3.3 of Chapter 3 describes configuration of IPv6-enabled network nodes. Section 3.2 of Chapter 3 describes configuration of IPv6 services, leaving the remaining sections of this chapter to provide more detailed discussions of all other transitional subtasks.

## 2.1 IPv6 Address Block and 6BONE

The current practice [HOD98] is that the IPv6 address block is obtained at the time of establishing IPv6 connectivity with an IPv6 aggregator of any level [HOD98]. IPv6 is relatively new in the production world, partially because allocation of production IPv6 address blocks began quite recently in the Summer of 1999. At this moment in time, there is no production-quality Top-Level Aggregator (TLA) [HOD98] near Ohio University, the issue being that only a few of the large Internet Service Providers (ISPs) have begun to provide IPv6 services. Because of this limitation, Ohio University was connected to one of the 6BONE aggregators.

To provide IPv6 connectivity to the Internet, the university must maintain a permanent connection with 6BONE. Such a connection is established via a static IPv6-over-IPv4 tunnel with a pseudo Top-Level Aggregator (pTLA). Rarely there exists a physical wire that links the router of the connecting site to a router at the 6BONE pTLA. In all other cases, Ohio University being one of them, the packets traveling between the connecting site and 6BONE have to traverse IPv4-only networks that are incapable of handling IPv6 traffic. Tunneling makes such traversal possible.

The tunnel may be established with one or more 6BONE pTLAs. The current list of available pTLAs is available at <http://www.6bone.net>. According to the 6BONE practice<sup>1</sup>, a site willing to join 6BONE should select a pTLA that is closest to the site, and then request a tunnel endpoint from the IPv6 contact for the selected pTLA. Based on performed research, as described in Appendix A, it was decided that the pTLA for Ohio University should be Merit, Inc. The IPv6 contact at Merit was contacted, and he allocated Ohio University a block of IPv6 addresses.

After obtaining a block of IPv6 addresses, Ohio University became the Site-Level Aggregator (SLA) [HOD98] for IPv6 prefix `3FFE:1CEF::/48` (the format of IPv6 address is defined in [HD98]). The lowest 80 bits of the 128 bits that comprise our

---

<sup>1</sup>See Appendix A for more information on 6BONE.

IPv6 address are available for the university networks. Out of those 80 bits, 64 bits are reserved for the host portion of IPv6 address [HOD98], and 16 bits are available for building an addressing hierarchy for the university.

The sizes of IPv4 and IPv6 address blocks available to the university may be compared as follows. The IPv4 address block presently available to the university permits us to split the university network into 254 subnets with the maximum of 254 network nodes on each subnet. On the other hand, the IPv6 address block offers 65,536 subnets, while the number of network nodes on each subnet is virtually unlimited<sup>2</sup>. Clearly, IPv6 address block offers greater room and flexibility for designing an efficient addressing hierarchy than the currently available IPv4 address block.

We use a dedicated router as a core IPv6 router for Ohio University. This provides separation of the testing IPv6 environment from the production IPv4 environment, and thus helps ensure that experiments with IPv6 do not have a negative impact on the availability of critical production systems at Ohio University. Nevertheless, after IPv6 reaches production quality, the functionality of this router should be moved to one of production routers.

The Ohio University core IPv6 router is a Cisco 2621 router, with two Fast Ethernet network interfaces, four Ethernet network interfaces, and two Wide Area Network (WAN) network interfaces. The choice of network interfaces on the router was made based on the results of router requirements analysis conducted during several discussions with Ohio University network administrators. Presently, only two Ethernet network interfaces are in use. After the initial stage of transition from IPv4 to IPv6 is completed, experiments in more specific areas of IPv6, such as multihoming, dial-up, or operation on point-to-point links, will utilize the remaining network interfaces.

---

<sup>2</sup>The number of hosts on one network is limited by  $2^{64} = (4 \text{ billion})^2$ .

## 2.2 Networks and Services

Two test IPv6 networks have been established on the main campus of the university. We shall denote those networks as network A and network B. Network A is the research network in the laboratory of Ohio University Internetworking Research Group (IRG). Network A has been running for one year – since Ohio University first connected to 6BONE. Network B is a testing network at Ohio University CNS. This network was established in January of 2000 and became the network providing IPv6 services to the university. The tunnel to 6BONE was switched from network A to the main router of network B. A diagram in the Figure 2.1 illustrates the primary components of the IPv6 environment, interconnection of the test networks, and IPv6 tunnels. Every link and tunnel coming out of the core IPv6 router and the IRG IPv6 router are marked with names in accordance with the actual configuration of the router. The IPv6-related configuration options of the IPv6 routers are shown in Figure 2.2 and Figure 2.3.

The test networks can be described as follows. Each test network is connected to the Internet via a dual-stack IPv4/IPv6 Cisco router. Network nodes are Sun machines running either Solaris 7 with IPv6 patches or IPv6-enabled Solaris 8. At this point in time, it is neither desired nor necessary to configure IPv6 networks as IPv6-only networks, because they would not be able to communicate with other machines without IPv6 support. An IPv6 DNS server runs on each test network. Presence of a DNS server on each network is not required, yet convenient, since in our environment we use multiple DNS servers to experiment with a multi-level infrastructure of IPv6 DNS server. There is no implementation of DHCPv6, as the latest specification for DHCPv6 will remain incomplete until October 2000. This does not presently allow us to perform experiments substituting DHCP with DHCPv6. The IPv6 router on network A is connected to the core IPv6 router of the university via a static link. This link provides IPv6 connectivity for network A.

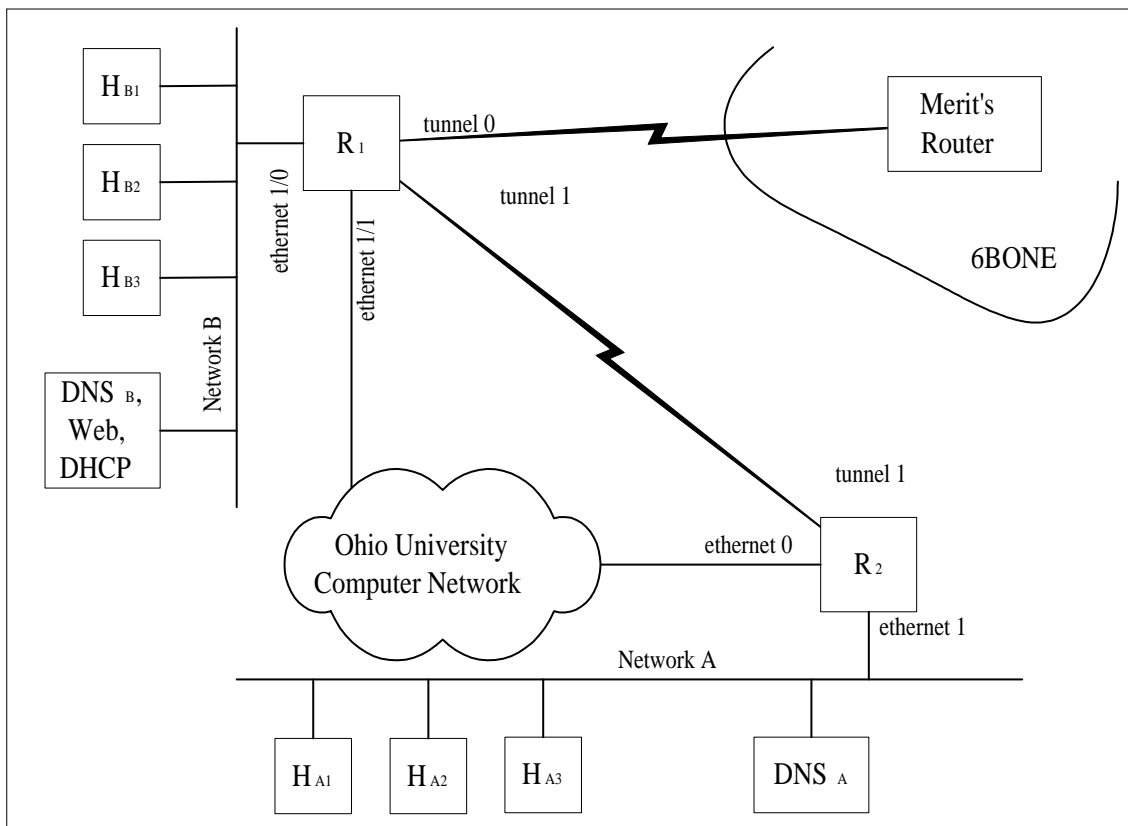


Figure 2.1 Diagram of Test IPv6 Environment.

This figure illustrates interconnection of the test networks and the 6BONE network. Straight lines denote physical links, while “lightning” lines denote IPv6 tunnels between non-adjacent routers.  $R_1$  is the core IPv6 router for the university, and  $R_2$  is the IPv6 router on the IRG network.

```

ipv6 unicast-routing
interface Tunnel0
  description IPv6 tunnel to 6bone.merit.edu (Merit, Inc.)
  ipv6 enable
  ipv6 address 3FFE:1CFF:0:F8::2/64
  tunnel source 132.235.79.253
  tunnel destination 198.108.0.3
  tunnel mode ipv6ip
!
interface Tunnel1
  description IPv6 tunnel to IRG Lab, OU
  ipv6 enable
  ipv6 address 3FFE:1CEF:0:1E00::2/64
  tunnel source 132.235.79.253
  tunnel destination 132.235.3.190
  tunnel mode ipv6ip
!
interface Ethernet1/0
  description CNS GPTN Ethernet
  ipv6 enable
  ipv6 address 3FFE:1CEF:0:0F:1:42FF:FE7F:2E88/64
!
interface Ethernet1/1
  description HDLC Router Connection
  ip address 132.235.79.253 255.255.255.240
!
ipv6 auto-tunnel
! Route to the IRG Network
ipv6 route 3FFE:1CEF:0:2F::/64 Tunnel1
! Route to the 6BONE backbone
ipv6 route 3FFE::/16 Tunnel0

```

Figure 2.2 IPv6 Configuration of the University Core IPv6 Router.

Two Tunnel interfaces are defined in this configuration. The interface `Tunnel0` provides the permanent connection with 6BONE. The interface `Tunnel1` serves the static tunnel that connects the IRG IPv6 network to the rest of the IPv6 networks in the university. We use network `3FFE:1CEF:0:1E00::/64` to establish tunnels.

```

ipv6 unicast-routing
!
interface Tunnel1
  description IPv6 tunnel to CNS IPv6 router
  ipv6 enable
  ipv6 address 3FFE:1CEF:0:1E00::3/64
  tunnel source 132.235.3.190
  tunnel destination 132.235.79.253
  tunnel mode ipv6ip
!
interface Ethernet0
  ip address 132.235.3.190 255.255.255.0
  ipv6 enable
  ipv6 address 3FFE:1CEF:0:2F:CFF:FE76:3DC6::0/64
!
ipv6 auto-tunnel
ipv6 route 3FFE::0/16 Tunnel1

```

Figure 2.3 IPv6 Configuration of the IRG IPv6 Router.

Network interface `Ethernet0` is configured for the network `3FFE:1CEF:0:1E00::/64`. Interface `Tunnel10` is an end-point of the IPv6 tunnel to the Core IPv6 router. This tunnel provides connectivity of the IRG network with the IPv6 world.

A Sun Ultra10 machine was acquired to run IPv6 services for the university. Planning ahead, these services would include central IPv6 DNS server for domain `ohiou.edu`, an IPv6 web server, DHCPv6 server, and other services as the need rises. For the purpose of today, we concentrate our attention on IPv6 DNS, as this is one of the most widely used and imperative services on the Internet. All of the services are presently concentrated on a single machine. Later, after the use of IPv6 services increases, certain services may be moved to separate machines to distribute load and possibly provide backup of services.

### 3. OUR PROPOSAL AND ITS IMPLEMENTATION

This chapter summarizes the work that we have done to improve the computer networks at Ohio University. As mentioned in our proposal, we rely on the improved and new features of IPv6 to eliminate certain weaknesses in the existing university networks.

The work that we performed consisted of two major parts: design and refinement of a new addressing hierarchy for the university computer networks, and deployment of this hierarchy on the test IPv6 network to resolve issues associated with deployment of IPv6. Section 3.1 describes the proposed addressing hierarchy, sections 3.2 and 3.3 describe the two primary components of the deployment process: configuration of Domain Name System and configuration of end-user support for IPv6. The changes that need to be made to the router configurations to adjust the topology of the current testing environment to the proposed topology are trivial. Therefore, they are not discussed in this paper.

#### 3.1 Proposed Addressing Hierarchy

The addressing hierarchy that is currently in use at Ohio University has certain weaknesses. In this section we propose a new addressing hierarchy that is designed to eliminate these weaknesses and introduce certain improvements. Our goal is to use the large block of IP addresses available from adoption of IPv6 to structure the addressing hierarchy in a scalable and efficient manner. First, we discuss the physical network infrastructure at Ohio University. Second, we describe the currently existing designs and point out certain weaknesses that we see in the addressing hierarchy.

Next, we explain our proposal for a new design of the addressing hierarchy. Finally, we discuss advantages and disadvantages of our proposal.

### 3.1.1 Overview of Network Topology

A diagram of Ohio University networks is shown in Figure 3.1.

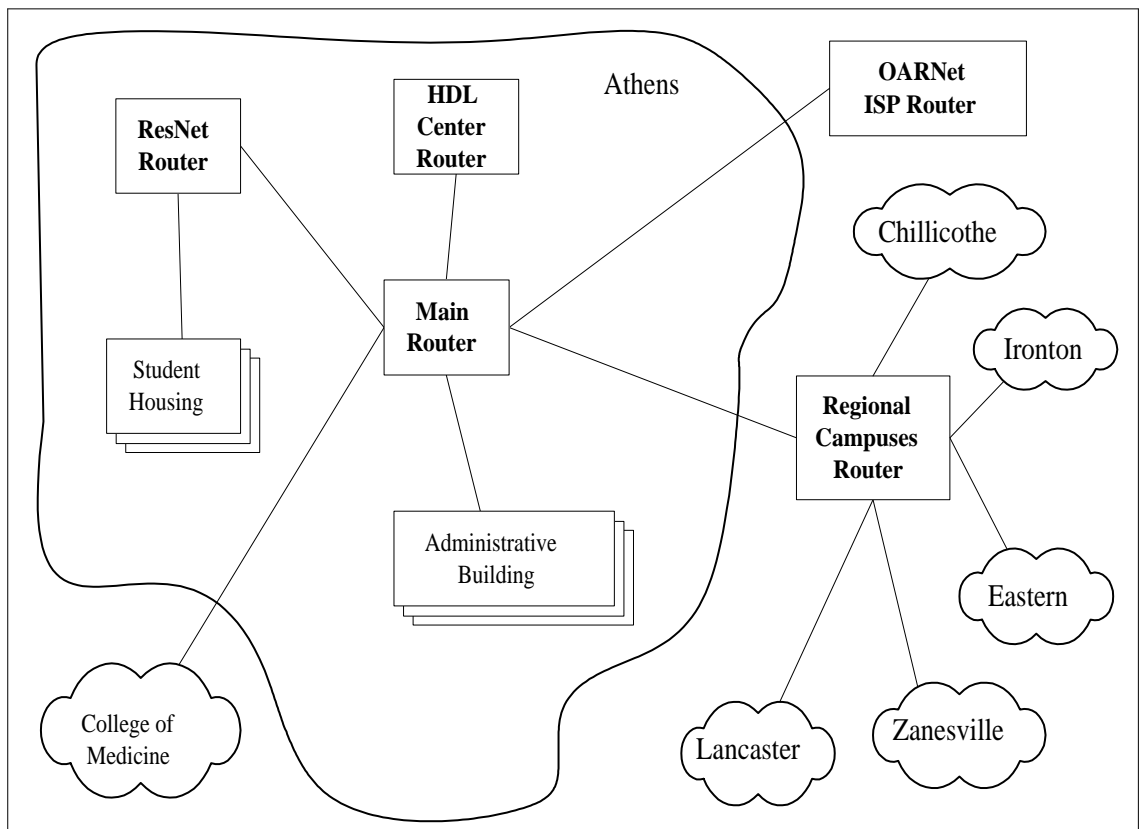


Figure 3.1 Layout of Ohio University Computer Networks. This figure shows interconnections of the main routers, vital links, and networks.

As can be seen from Figure 3.1, Ohio University is divided into six campuses. The main campus is located in Athens, and the five regional campuses are located in

Chillicothe, Eastern, Ironton, Lancaster, and Zanesville. Central routing equipment is located on the main campus. Each regional campus communicates with the main campus via a link that goes from the central router on the regional campus into the Regional Campuses router on the main campus. All networks on the main campus and all networks on the regional campuses are controlled by the network managers at Ohio University. All of these networks are administered by a single authority. There are several groups of networks that, even though they connect to the Ohio University network, are managed by other authorities. This does not seriously affect network management. However, the presence of parts of the network that are outside of the control of the central authority should be considered when designing a new addressing hierarchy for the network topology.

Principal structure and organization of the regional campuses resemble those on the main campus, but are of a smaller size. Student and employee populations on the regional campuses taken together are smaller than the ones on the main campus. Certain university facilities are available only on the main campus, and not on the regional campuses. The regional campuses perform a limited subset of functions. Therefore, the computer networks on regional campuses are smaller and their topologies are simpler.

Each campus of Ohio University has a number of buildings for educational, administrative, recreational, and other facilities. Most of these buildings need to be equipped with computer networks. Educational buildings offer students remote terminals and computer laboratories. Administrative buildings host one or more administrative units. In this case, the cohabitating administrative units may, or may not, require use of physically independent networks. It is often necessary that computer networks be isolated from one another. Data being sent within one network may not need to appear on the neighbouring computer networks, either because the data are of use

only to the users on this network, or because the data are sensitive and may be accessed only by certain types of users on this network. From these observations, it is apparent that Ohio University needs many distinct computer networks.

### 3.1.2 Issues With The Existing Topology

We performed an analysis of the existing topology of Ohio University networks. The primary weaknesses that we discovered fall into the following categories: complexity of management, inefficiency in assignment of IP addresses, low granularity of subnets assignment, and low performance routing protocols. These issues are discussed in the remaining subsections.

#### 3.1.2.1 Complexity of Management

Recall from Chapter 1 that the Ohio University computer network is split into many small networks. Assignments of IP address ranges to these networks do not follow a predefined and structured pattern. The first consequence, which we mention earlier, is that the routing tables in the routers are unnecessarily large. Another consequence is that it is harder to determine topological location of a node or a network, or the topological distance between two network nodes based on their IP addresses. Also, it is harder for a network manager to keep track of the correct configuration of subnets. In result, if there was a misconfiguration, it would be harder to restore correct configuration, because the IP address of an arbitrary network does not provide any information about the relative location of that network within the network topology.

The use of route aggregation permits systematic delegation of IP addresses and maintenance of the address pool based on the point of physical attachment of the network within the site. Knowing the IP address of a network, it is relatively easy to infer where in the entire hierarchy this network belongs. Also, route aggregation decreases the size of routing tables. For example, instead of advertizing all 32 subnets

that correspond to a 5-bit wide network range, the router advertizes only one subnet that includes all 32 subnets.

### 3.1.2.2 Inefficiency in Assignment of IP Addresses

During several discussions with Ohio University network administrators, it was discovered that there is a common practice in the university to assign IP addresses that belong to 254-node subnets to network interfaces on router-to-router, point-to-point links. These 254-node subnets belong to the globally unique range of addresses, and such assignment does not make an efficient use of globally unique IP addresses. First, it is not necessary to assign globally unique IP addresses to network interfaces that are only used by the two routers for local communication. Second, a point-to-point link needs only two IP addresses, one address on each side of the link. In other words, only two network interfaces can be physically attached to a point-to-point link, as it has only two physical connection points, one on each end of the link. However, 254 addresses are consumed. The remaining 252 IP addresses are never used on the link, and cannot be used on any other network because addresses of networks must not be duplicated [Pos81]. As a result, 252 globally unique IP addresses are wasted on each router-to-router, point-to-point link.

IPv6 provides specific link-local addresses for such needs. Every network interface is automatically assigned a link-local IPv6 address that is not globally unique. The node uses this address to communicate only with the machines connected to the same wire.

### 3.1.2.3 Low Granularity of Subnets

We identify at least one part of the university network that has low granularity of subnet assignment. This is a part of the Athens campus network called ResNet.

ResNet serves the college greens, which are groups of buildings, mostly student housing buildings, that are located in one geographical vicinity. There are three greens: South Green, West Green, and East Green.

ResNet contains the largest proportion of network users on the main campus. Every room of every student housing building has an Ethernet 10BASE-T network connection. All rooms in a single building are connected to the building hub, which in turn is connected to the ResNet router. Therefore, the ResNet router is linked to every student housing building. All networks that run from the buildings are configured as a single, shared-medium network with many network users. Such a configuration provides an easy way to manage many users at once.

The large number of users on ResNet network negatively impacts the performance of the present architecture of ResNet, because Ethernet uses CSMA/CD protocol [Com95] for transmission of data over the physical link. An important property of a CSMA/CD protocol is that before a host may transmit on the network, it has to ensure that no other host is transmitting. The time that the host spends waiting for the medium to become available is referred to as “contention time.” The contention time increases with the increase of the amount of traffic on the medium, the number of computers, and span of the medium. Long networks with a large number of active users have longer contention time than both shorter networks with the same number of active users or the same networks with a smaller number of users.

In the present ResNet architecture, each packet sent by a user in one building is seen by all machines in all other buildings. It is known that students do not run university-wide services on their room computers. Based on this observation, we speculate that the traffic leaving ResNet towards the Internet outweighs the traffic between ResNet computers. Clearly, it is not efficient for the ResNet router to forward all packets going from a student computer in one building to an Internet site to all machines in all other buildings on ResNet. This unnecessarily increases traffic in all buildings and increases the span of ResNet. The result is the increased contention

time and consumption of bandwidth. A solution is to assign a separate network to each building. If the ResNet network could be split into subnets, the performance on each of the subnets could be improved.

Certain functionality of Microsoft and Novell networks can be affected by splitting ResNet into subnets. Microsoft and Novell Networks use IP broadcast messages to communicate information to all machines on a network. Splitting the entire ResNet network into subnets limits reachability of broadcast messages to only the nodes that are connected to each subnet. If no special measures are taken, broadcast messages would not be able to reach all student computers on ResNet. As a solution, it is possible to replace functionality of IP broadcast with IP multicast. The use of IPv6 is appropriate here. IPv6 allows network nodes to be assigned multiple IP addresses. Therefore, several IPv6 multicast addresses may be assigned to a network interface to enable its participation in more than one broadcast-based service with clear differentiation of each particular services. Assignment of the multicast addresses may be automated on ResNet network, provided that IPv6 multicast is properly configured on the ResNetrouter. We discuss the use of IPv6 with Microsoft software in Section 3.3.3.

#### 3.1.2.4 Low Performance Routing Protocols

Routing information in the university networks is distributed via RIP [Ma198]. While use of this protocol is sufficient, better route calculations and more efficient routing is offered by more complex protocols such as OSPF [Moy98] or IGP [Gro92]. IPv6 already has support for OSPFv6.

### 3.1.3 The Proposed Addressing Hierarchy

We propose application of route aggregation techniques to the address space provided by IPv6, as the basis for designing an addressing hierarchy to make routing of computer networks at Ohio University more efficient and scalable. There are issues

associated with the use of route aggregation that need to be carefully considered in order to prevent problems in the future. Therefore, the addressing hierarchy that we propose must satisfy the following requirements:

1. Address allocation must use route aggregation.
2. Address ranges must be allocated to computer networks in ways that would permit, at any level of the addressing hierarchy, ease of reasonably large expansion of computer networks in conformance with requirements for route aggregation.
3. Address space should not be overly allocated. Reservation of address space for future expansion of the currently existing networks must be reasonable.
4. It should be possible to vary depths of any level of the addressing hierarchy as necessary, within the boundaries applied by the size of the available address space.

Recall from Section 2.1 that Ohio University is a SLA for the IPv6 address prefix `3FFE:1CEF::/48`, which means that the first 48 bits of our 128-bit address are fixed. Out of 80 remaining bits available to the university networks, 64 bits are reserved for the host portion of an IPv6 address, which leaves 16 bits that may be used to build the addressing hierarchy within the university. If no addressing hierarchy is used within the site, the 16 bits would represent at most 65,535 different subnets. This would be more than enough for the university. However, we do not advise this approach because it only eliminates shortage of IP addresses, and leaves untouched other weaknesses that we identified.

The addressing hierarchy that we propose consists of five levels and is based on a mix of geographical and administrative separations. Geographical separation, employed at the top level of the hierarchy, provides better isolation of different campuses of the university, thus guaranteeing administrative network independency for every campus. At the levels below the campus level, geographical separation may not be

appropriate for every case. Hence, to compensate for insufficiency of the geographical separation, administrative separation may be employed where necessary. In terms of addressing hierarchy, its first level is Ohio University as a whole. At the next level, the address space is divided among campuses. Within each campus, distinct areas are identified, based on their either geographical or on administrative attributes. Any such area consists of one or more sites. A site may contain a number of networks. The conceptual diagram in Figure 3.2 illustrates this addressing hierarchy.

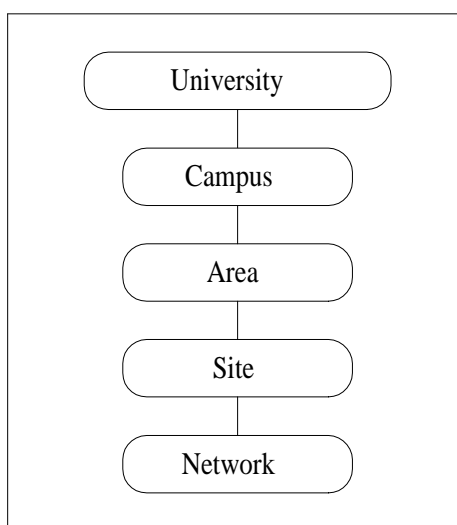


Figure 3.2 Conceptual Diagram of the Proposed Addressing Hierarchy.

#### 3.1.3.1 Top Level of the Addressing Hierarchy

The 16-bit wide address space available to Ohio University gives the maximum 65,535 different networks. Ohio University does not presently need such large address block. Therefore, we allocate to the university one half of the available address space. The other half is reserved for future allocations. To implement this, we reserve all

IPv6 addresses that have bit 15 set<sup>1</sup>. After the reservation, the available address space is represented by IPv6 prefix  $3FFE:1CEF::/49$ .

The prefix  $3FFE:1CEF::/49$  is divided among the campuses. Table 3.1 contains the summary of all allocated prefixes, assignment of those prefixes to the campuses, and reservations of address space. The main campus has a greater number of networks and network users than all the regional campuses put together. Based on this information, the address space allocated for the main campus should be larger than the address space allocated for all regional campuses. A portion of the address space should also be reserved for future use. The next bits are utilized to distinguish campuses. The first two bits produce four combinations of prefixes:  $00_2$ ,  $01_2$ ,  $10_2$ , and  $11_2$ . The main campus receives prefix  $00_2$ , the regional campuses share the address space under prefix  $01_2$ , and prefixes  $10_2$  and  $11_2$  are reserved. All five regional campuses share their address space under the prefix  $01_2$  by using the next three bits; that provide eight prefixes from  $000_2$  to  $111_2$ . Concatenated with the regional campus prefix, these result in:  $01000_2$  for Chillicothe,  $01001_2$  for Ironton,  $01010_2$  for Eastern,  $01011_2$  for Lancaster, and  $01100_2$  for Zanesville. If a regional campus needs expansion of the allocated address space, addresses from the remaining three prefixes and from the reserved space at the above levels should be used.

The Figure 3.3 displays a binary tree that graphically demonstrates how the top portion of the address space is allocated and explains the contents of Table 3.1. The number on an edge represents the binary digit in the prefix, and the distance of the edge from the root of the tree represents the offset of that binary digit from the most significant bit of the 16-bit address prefix. The top-level edges correspond to the most significant bit of the 16-bit address prefix.

---

<sup>1</sup>The bits are numbered relatively to their position within the 16-bit address space available to the university, the numbers being from 0 to 15.

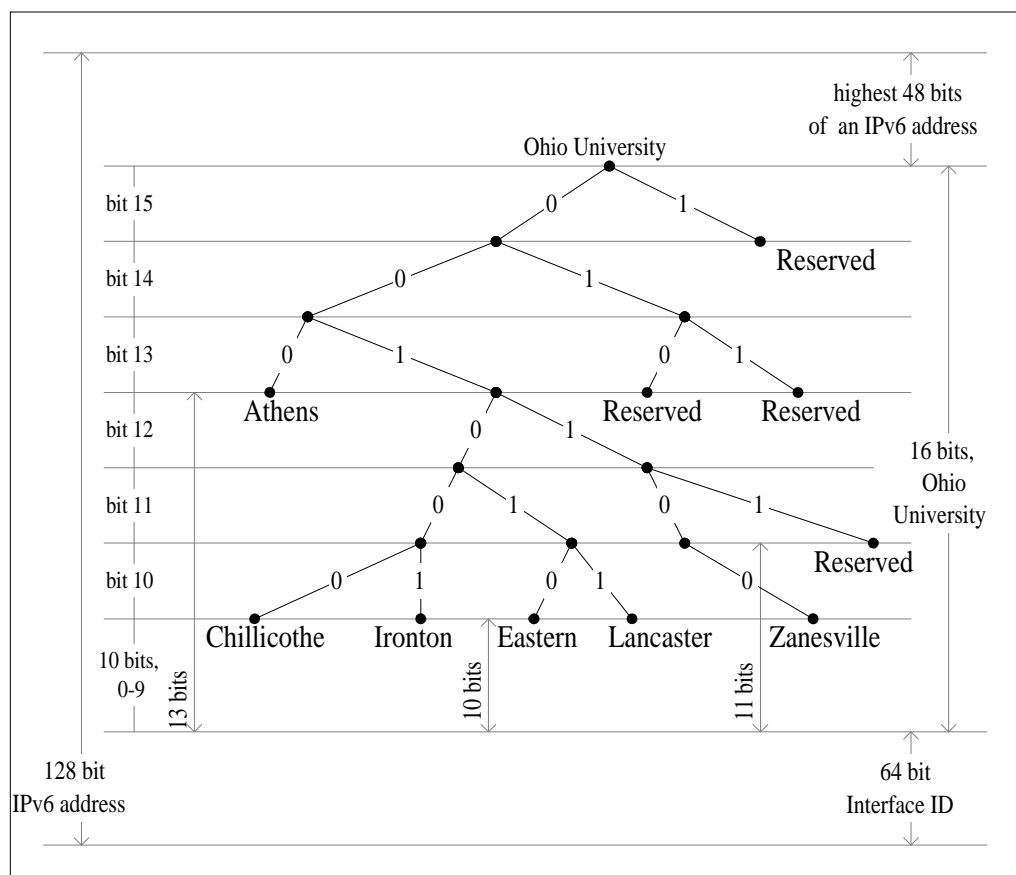


Figure 3.3 Allocation of the Top Portion of the Address Space.

This figure shows how the top several bits of the site topology portion of the IPv6 address are allocated to divide the available address space among campuses.

Provider	Address Space Bits					Allocation	IPv6 Prefix
	15	14	12	10	9-0		
		13	11				
3FFE:1CEF:	1					<i>Reserved</i>	3FFE:1CEF:8000::/51
	0	00				Athens	3FFE:1CEF::/51
	0	01				Regionals	3FFE:1CEF:2000::/51
	0	01	00	0		Chillicothe	3FFE:1CEF:2000::/54
	0	01	00	1		Ironton	3FFE:1CEF:2400::/54
	0	01	01	0		Eastern	3FFE:1CEF:2800::/54
	0	01	01	1		Lancaster	3FFE:1CEF:2C00::/54
	0	01	10	0		Zanesville	3FFE:1CEF:3000::/54
	0	01	10	1		<i>Reserved</i>	3FFE:1CEF:3400::/54
	0	01	11			<i>Reserved</i>	3FFE:1CEF:3800::/53
	0	10				<i>Reserved</i>	3FFE:1CEF:4000::/51
	0	11				<i>Reserved</i>	3FFE:1CEF:6000::/51

Table 3.1 Compilation of Top-Level Aggregation Prefixes

### 3.1.3.2 Addressing Hierarchy for the Main Campus

To design a good addressing hierarchy within a site, it is necessary to have sufficient knowledge of the network topology at that site. The previous sections of this chapter described the topology of Ohio University networks. We propose the addressing hierarchy for the main campus based on this information, as shown in Figure 3.4.

According to Table 3.1, the address space for the main campus includes bits 0 through 12, which is 13 bits<sup>2</sup>. Referring to the conceptual diagram of the addressing hierarchy shown in Figure 3.2, we identify three hierarchical levels within a campus: area, site, and network. We assign each level a number of consequent bits from the available 13 bits, and the number of bits that we assign determines the width of that hierarchical level.

<sup>2</sup>It is  $16 - 3 = 13$ , where 16 is the number of bits available to Ohio University, and 3 is the number of bits used to divide the entire address space between all campuses and the reserve.

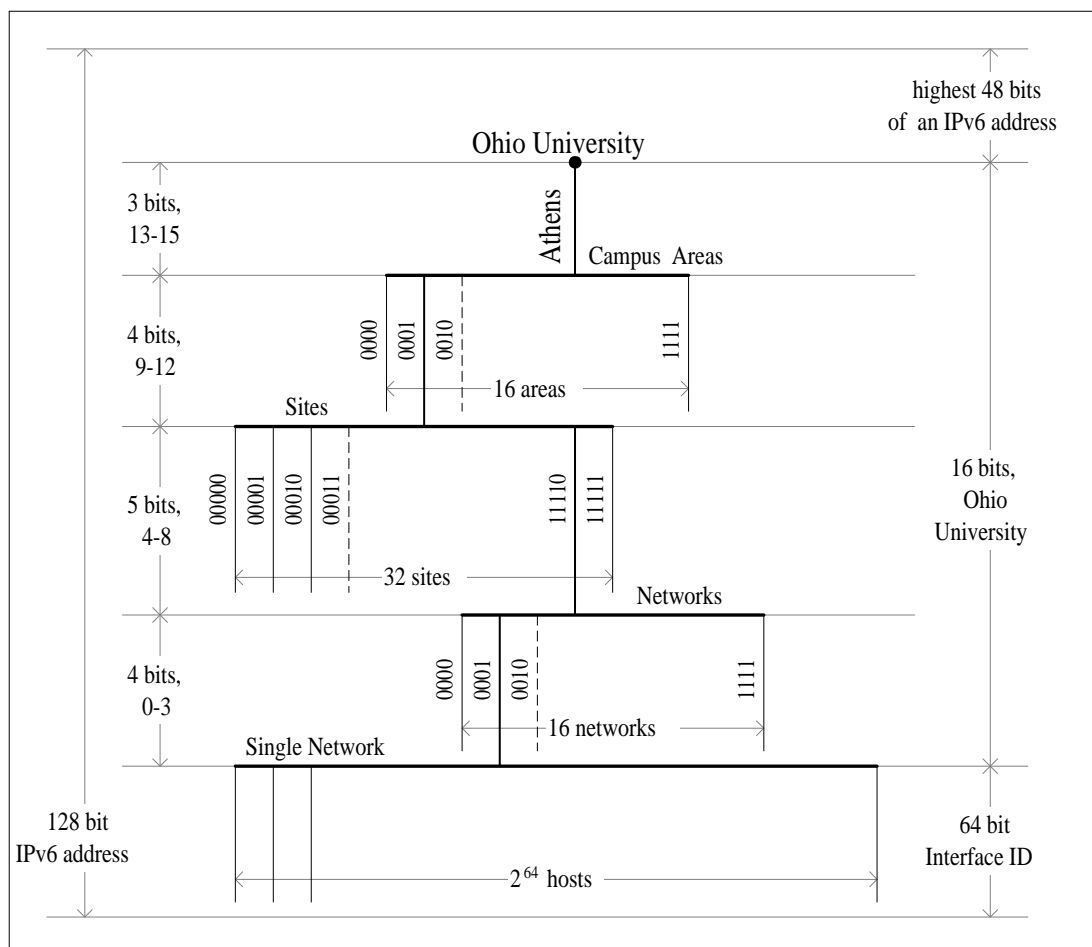


Figure 3.4 The Addressing Hierarchy for the Main Campus.

The first level within the campus addressing hierarchy describes campus “areas”. A campus may be divided into distinct areas, such as administrative buildings, housing areas, groups of educational buildings. This division may be either geographical or functional, or mixed. We allocate 4 bits for this level, which provides 16 different campus areas. This number should be sufficient, because we assume low granularity division of the campus into areas.

The next level of the addressing hierarchy is for “sites”, which in most cases represent buildings. An area is comprised of a number of sites. The sites are viewed as a specific level because of the physical layout of networks. Each building is generally equipped with routing equipment that is responsible for communicating with the central network and perhaps for managing the traffic inside the building. We allocate 5 bits for this level, which is at most 32 buildings per area. If a certain area is larger than that, another prefix for campus areas may be used, increasing the maximum size to 64 buildings, and so forth. The cost such an increase depends on which area prefixes are used. If the additional area prefix is next<sup>3</sup> to the prefix in use, then the routing entry for the area will have to be adjusted. If the additional area prefix is separated from the prefix in use by other prefixes, then an additional routing table entry will be added.

The third level of the addressing hierarchy manages “networks”. These are the fundamental components of a site. A sufficiently large building may have many networks, either because there may be several distinct administrative units located in the building that require separate networks, or because there is a need to physically divide a network into several smaller networks. The number of network nodes that can be connected to one physical network is very large<sup>4</sup> [HD98]. This permits division of the site into networks to be based purely on logical needs, and need not be affected

---

<sup>3</sup>Two prefixes are “next” to one another, when their mathematical difference equals 1. For example,  $00110110_2 - 00110101_2 = 1$ .

<sup>4</sup>The number of hosts on one network is limited by  $2^{64} = (4 \text{ billion})^2$ . It is larger than the size of the entire existing IPv4 address space.

by the projected number of hosts that will connect to the network. We allocate 4 bits for networks within one building. This translates into at least 16 networks per building. Additional networks may be acquired by assigning more building prefixes to the building.

The choice of bit distribution among the three hierarchical levels is determined by the trade-off between having too many areas or too many networks. We do not want to split the address space into many areas, because it will degrade routing efficiency at the top level, but we also do not want to have too little capability for differentiating distinct areas of a campus because we have too few area prefixes. To determine in what proportion to allocate bits for sites and networks, we considered the typical number of networks that exist in the buildings on campus. Reasonable choices were 8, 16, and 32 networks. We chose 16 as a trade-off. If it was a smaller number, there would be too many routing entries in the routers. If it was a larger number, the buildings with only few networks would be wasting too many network prefixes. It is more appropriate to allocate an additional building prefix to a building, should the building need more than 16 networks. Finally, by allocating 4 bits to the area and the network levels, 5 bits remain for use at the sites level. The overall hierarchy seems to be appropriate for most cases. Exceptional cases may be handled separately. An example of such case is provided in Section 3.1.3.4.

### 3.1.3.3 Addressing Hierarchy for a Regional Campus

Structure and organization of the regional campuses resemble those on the main campus, but are of a smaller size. Therefore, the addressing hierarchy for the main campus is appropriate and acceptable for the regional campuses. However, the structure for the main campus cannot be directly adopted by the regional campuses, because the main campus has prefix length three bits longer than that for the regional campuses. This difference in the lengths is appropriate because the regional campuses are smaller than the main campus, and their network topology is simpler. To

compensate for the shortage of bits, we propose to take one bit out of each level of the addressing hierarchy for the main campus. This would result in the following bit distribution: 3 bits for areas (the maximum of 8 areas), 4 bits for sites (the maximum of 16 sites), and 3 bits for networks (the maximum of 8 networks). This adjusted addressing hierarchy is illustrated in Figure 3.5.

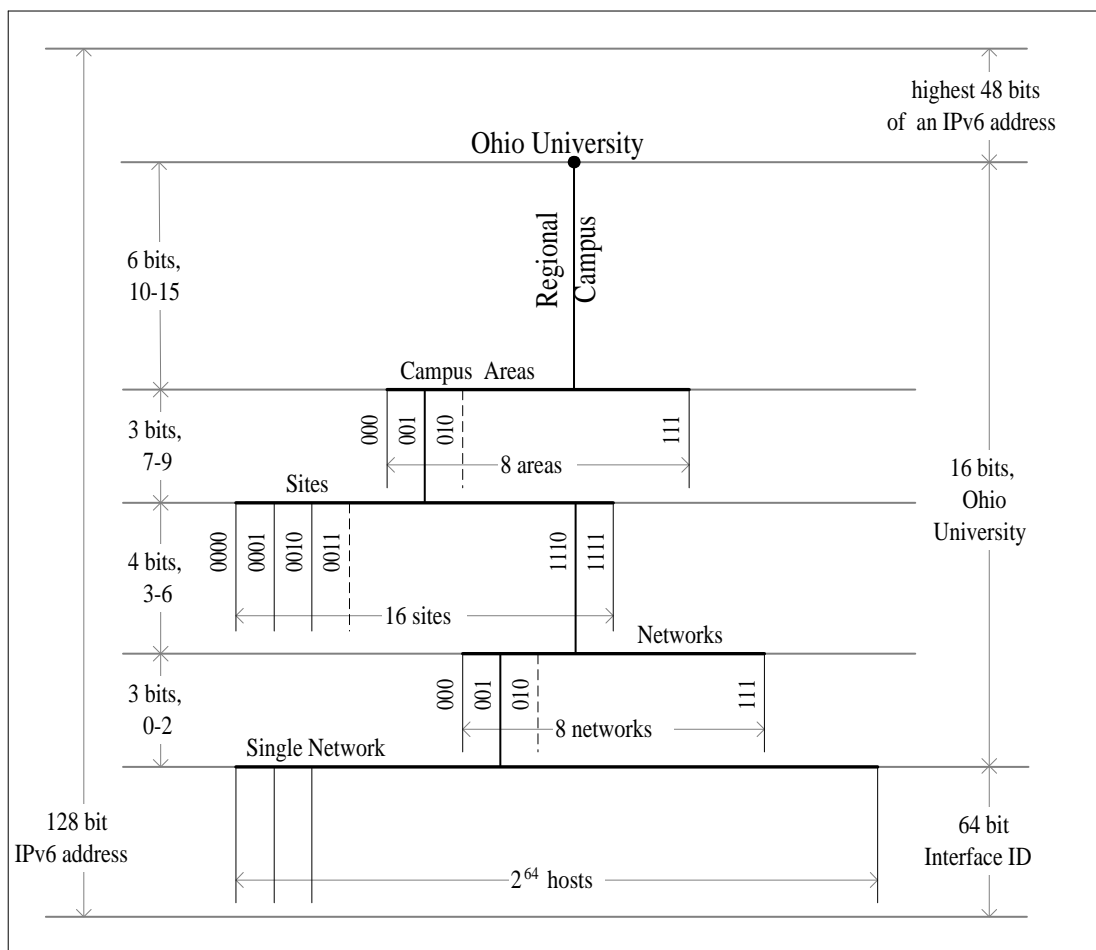


Figure 3.5 The Addressing Hierarchy for a Regional Campus.

### 3.1.3.4 Example Exceptional Case

In this section we demonstrate how the proposed addressing hierarchy deals with a case when there is a change that needs to be made, and it does not fit in the proposed addressing hierarchy. Let us consider what happens if every room in the student housing facility was to have a its own computer network. The present configuration both with IPv4 and IPv6 assumes that only one computer can be connected in a student's room. Some students would find it useful to be able to connect several networked items together and have Internet access.

First we calculate how many networks would be required. We could assume that all enrolled students may possibly live on the main campus. Clearly, some students live off-campus, so the actual number is lower. Calculations based on actual numbers obtained from Ohio University webpages show that the upper bound on the number of networks that need to be allocated is approximately 8,000. This number can be represented by the lower 13 bits of the address space. A portion of the reserved address space should be used to allocate the required block of IPv6 network addresses to these student networks.

Presently, student housing buildings do not have routers installed, and building hubs are connected directly to the ResNet router. It is impossible to employ route aggregation below the ResNet router in such an environment. As a result, the ResNet router would have to route a maximum of 8,000 networks. It is not an efficient approach. Instead, routers should be installed in every building. In that case, the 13-bits address space could be further subdivided to aggregate networks within each building and advertize to the ResNet router only the aggregated prefixes. An example aggregation is shown in Figure 3.6.

Following are the numbers provided by Ohio University housing department<sup>5</sup>. The three greens South Green, East Green, and West Green have 19, 14, and 7 buildings respectively. Student populations on these greens are 2,450, 2,450, and

---

<sup>5</sup>These numbers are available from the URL <http://www.ohiou.edu/housing>.

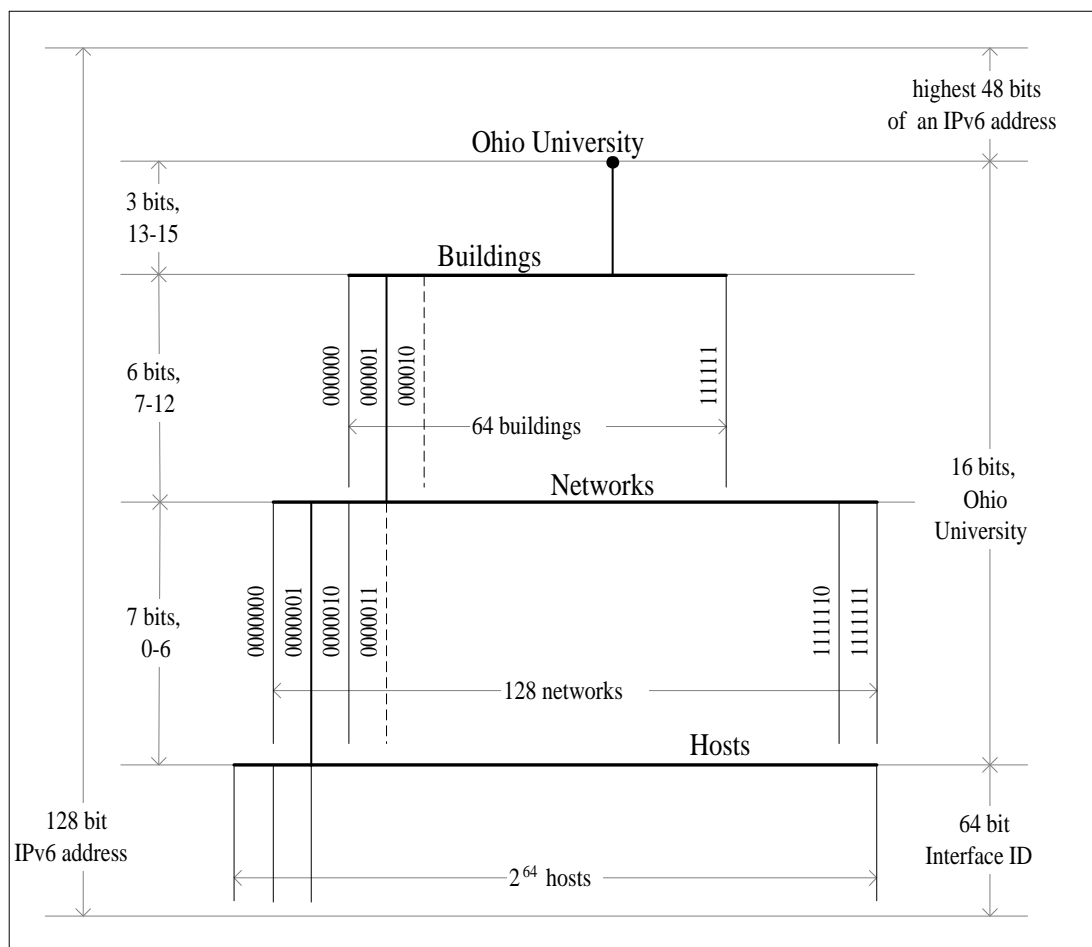


Figure 3.6 The Example Addressing Hierarchy for the ResNet network.

1,779 students. There are 40 buildings total, which can be represented by a minimum of 6 bits. This means that 7 bits remain to be used within a building. However, 7 bits provide a maximum of 128 networks, while some buildings accommodate almost 400 students. As mentioned earlier, in such cases several prefixes may be merged to provide the needed number of network numbers. The routing tables would be of the following sizes. Since there is only one router within a single building, further route aggregation within a building is not possible. Therefore, the building router would have to route 100 to 366 networks, depending on the size of the building. Only one IPv6 prefix per every 128 networks would be advertized by the building router to the ResNet router. The ResNet router would deal with the routing of at most 64 networks network prefixes. Summarizing, these numbers represent a much lighter load on every participating router for the cost of acquiring new routers.

#### 3.1.3.5 Summary

The addressing hierarchy that we propose eliminates certain weaknesses that we found in the presently used addressing hierarchy by introducing a number of improvements. We use a large block of IP addresses available from adoption of IPv6 to structure the addressing hierarchy in a scalable and efficient manner.

The proposed addressing hierarchy allows route aggregation. Each campus of the university is allocated a separate address block that the campus may arbitrary use within its boundaries. Routing information among campuses is represented by a single IPv6 prefix per campus. The regional campuses router thus has 6 routes in its routing table, which makes routing on that router very efficient. The central Ohio University router has more entries in its routing table, but these are limited by a magnitude of tens and not hundreds, as they are now.

Requirement for route aggregation imposes certain limitation on address assignment. A network may no longer be assigned an arbitrary network address, because its address should be aggregatable with the addresses of the higher levels of the employed

network topology. This is an architectural disadvantage of the proposed addressing hierarchy, but careful planning of the addressing hierarchy should minimize this undesirable effect.

Division of the address space into five hierarchy levels should not be considered fixed. It is understood that one addressing hierarchy cannot fit all needs. This is one of the reasons why we leave a large portion of the available IPv6 address space in reserve. As for the already defined addressing hierarchy, it is always possible to select a subtree of the campus hierarchical tree, and employ a completely different addressing hierarchy within that subtree. As long as care is taken to insure that new hierarchy conforms to the requirements of route aggregation, this is an alternative to the addressing hierarchy that we propose for the university as a whole.

## 3.2 IPv6 DNS Service

This section is dedicated to the provision of IPv6 DNS services at Ohio University. We describe details on configuration of the IPv6 DNS service in regard to the Ohio University network environment, the reasons that guided the design of the presented IPv6 DNS architecture, and the guidelines for setup and maintenance of the IPv6 DNS servers that have been placed in operation on the university networks. This section assumes that the reader is familiar with the basic principles of DNS for IPv4 [Moc87a] [Moc87b].

The DNS service is an essential part of any network architecture. Although the principal application of DNS is to provide name-to-address and address-to-name mapping for an IP address, there are other indirect applications. First, as mentioned earlier, it is a general practice to refer to the computers on the Internet via their names, not their IP addresses. IP addresses may be harder to remember than their corresponding names, especially since IPv6 addresses which are longer than IPv4 addresses. Second, domain names help separate addressing of a host at the network

level [Com95] from the services provided by the host. From the users' standpoint, a host represents services, and therefore should be assigned a name describing services available on the host. Third, provided that IP addresses are hidden underneath their domain names, as viewed by the users, it is possible to make changes to IP addresses without affecting the view of users. For instance, the IP address of the host `www.ohiou.edu` may change, but users will still be able to access the same service via its name, as long as the corresponding resource record is updated in the DNS server.

All the DNS related issues that we describe in the previous paragraph remain valid with introduction of IPv6. The primary difference is that the IPv6 address is four times longer than IPv4 address, thus making IPv6 addresses even harder to type and remember.

### 3.2.1 Integration of IPv6 DNS into the Production Environment

Prior to commencing deployment of IPv6 DNS, it is important to be sure that the new service can be seamlessly integrated into the existing IPv4 DNS environment [GN96]. This section describes the issues that were encountered during the integration process and how they were resolved.

#### 3.2.1.1 DNS Extensions to Support IPv6

A number of extension resource records have been introduced to the Domain Name System to enable DNS support for IPv6. IPv6 extensions to DNS are described in [TH95] and [CHT00]. At the present time, there exist two generations of IPv6 extensions to DNS, one of which is becoming obsolete. This generation of extensions was designed at the early stages of IPv6 development in an attempt to integrate DNS with IPv6. The resource record AAAA was used to store IPv6 addresses, similar to resource record A in IPv4. The only difference between A and AAAA resource records is the type of IP address each stores: A stores an IPv4 address [Pos81]; AAAA stores an IPv6 address [HD98]. An example assignment of the domain name `host1.domain.example`

to the IPv6 address `3FFE:1CEF:0000:0000:1234:5678:9ABC:DEF0` is provided in Figure 3.7. For reverse lookups, IPv6 uses a slight modification of the mechanism employed for IPv4. This mechanism is fully described in [TH95], and we briefly summarize it here. Reverse DNS entries are stored under the top-level domain `IP6.INT` [TH95]. Every byte of the IP address represents a next level domain, descending from `IP6.INT` in such a way that the most significant byte of the IP address comes immediately after `IP6.INT`, and the last significant byte comes last. The ordering is determined by the fact that DNS labels are ordered with the most significant label being the right-most within the domain name.

The name-to-address mapping:

```
host1.domain.example. \
    IN AAAA 3FFE:1CEF:0000:0000:1234:5678:9ABC:DEF0
```

The address-to-name mapping:

```
0.F.E.D.C.B.A.9.8.7.6.5.4.3.2.1.\
0.0.0.0.0.0.0.0.F.E.C.1.E.F.F.3.ip6.int. \
    IN PTR host1.domain.example.
```

Figure 3.7 Examples of Obsolete IPv6 DNS Resource Records.

In our work, we purposely avoided using the first generation of IPv6 extensions to DNS because it is being replaced by the second generation of IPv6 extensions. The second generation of IPv6 extensions to DNS, which we discuss in Section 3.2.3.4, provides a level of IPv6 support that includes a much wider set of IPv6 functionality, especially since the functionality that emerged after the first generation of IPv6 extensions to DNS, and is completely different from the first generation. We explored what was necessary to put the newly developed functionality of IPv6 DNS to use. The current practice with these second generation resource records is minimal, as it

has not been used in any working environment. The deployment of this functionality has not been performed, and no documentation on this topic is presently available. The work described in this section explores the most recent improvements to IPv6 DNS and demonstrates applicability of their new functionality to a real world system, as well as the ability of IPv6 DNS to operate in parallel with the production IPv4 DNS.

### 3.2.1.2 The Choice of Software for Provision of IPv6 DNS

We use Berkley Internet Name Daemon (BIND) [Con00] version 9.0.0b2 to run the DNS service at Ohio University. As of April 2000, it is the only available implementation of the new specification of IPv6 DNS. BIND-9.0.0b2 is a second beta revision of the code. Some parts of the DNS functionality are not yet implemented, but the new IPv6 extensions are already in place. During the course of using this software, we detected a number of minor coding problems and bugs. Solutions to these problems were empirically found and documented. It is expected that a production-quality release of BIND will become available in the Summer of 2000.

### 3.2.1.3 The Need for DNS Test-Servers

We need to setup IPv6 DNS test-servers. It is not acceptable to use the available DNS software, BIND-9.0.0b2, in the production environment at Ohio University. Any new software has to go through a test cycle prior to moving into a production environment. The testing should be complex enough to check compliance with the requirements of a production environment. Second, BIND-9.0.0b2 cannot be qualified as a complete and stable implementation of DNS specification. None of the production servers can be used for IPv6 support, because the software that runs the production DNS services does not have support for the functionality that we desire to utilize. From the observation above, it is apparent that one or more DNS server instances must be configured in a non-production environment on non-production

machines to test operation of IPv6 DNS. We will refer to such DNS servers as “DNS test-servers.”

#### 3.2.1.4 Integration of DNS Test-Servers into the Operating Environment

A DNS test-server should be designed to perform functions of the default DNS server for IPv6-enabled hosts. Therefore, from the host’s standpoint, the DNS test-server must be a complete substitute of its production analog. It must be able to provide all functionality available from the main production server. In addition to this, the DNS test-server must also provide IPv6 functionality. In this section, we explain the issues with configuring a DNS test-server to substitute the main production DNS server at Ohio University.

It is essential to prepare one top-level IPv6 DNS server that could provide DNS information, both IPv4 and IPv6, for the domain `ohiou.edu` and its subdomains. This server would be the default DNS server for most of the IPv6-enabled hosts. Establishing such a DNS test-server is a non-trivial task because of the nature of operation of the DNS protocol. An obvious solution would be for the DNS test-server to provide only IPv6 resource records for the entire domain `ohiou.edu`, and send all IPv4-related queries to the main production server. However, such a scenario is not feasible because of the following reasons. One of the primary requirements for our DNS test-server is that it provides *authoritative* answers for the domain `ohiou.edu` and all descending subdomains, the way the production DNS server does. A client receiving an answer to its query from a DNS server may be assured that the answer is correct – neither outdated nor invalid – only if the server returns an authoritative answer. A server provides authoritative answers only if it is the primary server for the domain or one of the secondary servers. It is assumed by the DNS protocol that an authoritative server is in possession of the complete information for the domain. Therefore, if the authoritative server contained only IPv6 resource records, from the protocol’s standpoint it would be correct to assume that no IPv4 resource records

are defined for the domain. Clearly, the only way to configure the DNS test-server so that it would provide both IPv4 and IPv6 RRs is to keep both IPv4 and IPv6 resource records under the authority of that DNS test-server.

### 3.2.1.5 Automatic Combining of IPv4 and IPv6 Resource Records

In order to implement integration of the DNS test-server with our operating IPv6 environment, we developed software that allows us to combine IPv4 and IPv6 resource records for a specific zone<sup>6</sup>, which come from multiple DNS servers, under the authoritative control of a single DNS server. Supplied with the name of the domain to operate on, the software accomplishes its task without human intervention, through a series of steps.

As the first step, the software performs an IPv4 zone transfer [Moc87b] for the specified Internet domain (e.g. `cs.ohiou.edu`) from the production DNS server that is authoritative for that domain. A standard tool named-xfer<sup>7</sup> is used to perform zone transfers. The tool stores the extracted IPv4 resource records in the master file format [Moc87a]. A zone transfer is a term used to denote a process of acquiring from a DNS server all possessed information regarding a specified domain.

As the second step, the software includes the IPv6 resource records for the specified domain in the master file with the IPv4 resource records obtained from the production DNS server in the first step. We store all IPv6 RRs for a domain in a single file. The contents of this file conforms to the master file format [Moc87a], but does not include the SOA record [Moc87a] that describes the parameters of the zone itself. The SOA record should not be included in this file, because this file is included in a master file obtained from the production DNS server and that file already contains the SOA resource record for that zone. We use an “INCLUDE” directive supported by BIND

---

<sup>6</sup>For the purposes of explanation, we shall assume the term “zone” to have the same meaning as the term “domain.”

<sup>7</sup>named-xfer is a part of the standard distribution of BIND.

to include the IPv6 master file into the IPv4 zone master file. Ultimately, the IPv4 and the IPv6 resource records are combined under a single zone.

As the final step, certain parameters of the zone are adjusted to correspond to the configuration of the particular DNS test-server. Originally, these parameters were retrieved from the production DNS server, so they were set to the values corresponding to, and appropriate for, that DNS server. Prior to using the zone file on the DNS test-server, changes of certain parameters are required. These parameters are: the fields `MNAME` and `RNAME` of the SOA resource record[Moc87a], as well as NS RR and, possibly, MX<sup>8</sup> RR for the zone label. We also adjust the value of the serial number for the zone. The software parses the master file and makes necessary substitutions of the values of these parameters, as well as of the serial number. The process of adjustment of the serial number, along with an example illustration of the substitution process can be found in Appendix D.

### 3.2.2 IPv6 DNS Test-Servers

Ohio University may need several IPv6 DNS servers. At least one DNS server – a top-level IPv6 DNS server – is required to provide IPv6 DNS service for the users [Bra89]. However, one DNS server is not sufficient to properly reflect the existing infrastructure of DNS servers in the university, as not all domains under the Ohio University top-level domain `ohiou.edu` are managed by the central authority of the university. When a site that runs its own DNS begins using IPv6, it will need to setup its own local IPv6 DNS server. That local DNS server will be placed hierarchically underneath the Ohio University top-level IPv6 DNS server. To be prepared for this, we designed our test IPv6 DNS infrastructure in a way that resembles this predicted scenario. We would like to be prepared for providing a relatively complex infrastructure and the use of only one server does not reveal certain details that can only be

---

<sup>8</sup>Change of MX is optional, depending on whether it is desired to intercept all email traffic for the domain. This may be useful because people may be sending email using IPv6.

attributed to multi-level hierarchies. This section describes the DNS test-servers we that have set up, their role in the overall DNS infrastructure, and their correlation among themselves.

### 3.2.2.1 The Multilevel Hierarchy of DNS Test-Servers

Our infrastructure consists of two IPv6 DNS servers: one Ohio University top-level DNS test-server (labeled `nsv6.cns.ohiou.edu`), and one DNS test-server for the Computer Science (CS) Department (labeled `nsv6.cs.ohiou.edu`). The top-level server is functionally equivalent to the Ohio University main production IPv4 DNS server (labeled `watson.cns.ohiou.edu`). Thus any IPv6-enabled host willing to resolve host names to IPv6 addresses should refer to `nsv6.cns.ohiou.edu` as its primary DNS server. It should be mentioned that although `nsv6.cns.ohiou.edu` is the top-level IPv6 DNS server, it does not provide authoritative answers for hosts in every domain under `ohiou.edu`. The server provides authoritative answers for hosts in only those domains that contain IPv6-enabled hosts. In those cases, however, the server is authoritative for both IPv4 and IPv6 resource records<sup>9</sup>. The DNS server in the CS department is authoritative only for its own domain, that is `cs.ohiou.edu`. For this server, the IPv4 resource record information is extracted from the IPv4 DNS server for the CS department, and combined with the IPv6 resource records for IPv6-enabled hosts in the departmental networks. In addition, the top-level DNS server is configured to refer all queries regarding domain `cs.ohiou.edu` to the CS department server, thus making IPv6 resource records of the CS department available to the machines outside the department.

### 3.2.2.2 Provision of Secondary DNS Test-Servers

In our design, the two DNS test-servers mutually serve as secondary servers for one another's domains. One reason for such a configuration is the need for provision

---

<sup>9</sup>Section 3.2.1.5 described how this is accomplished.

of secondary servers [Moc87b]. Another reason is our desire to better utilize resources of the machine and exercise performance of the DNS test-servers in a more complex environment. The hierarchical and functional relationships established between the two servers are shown in the Figure 3.8.

As can be seen from the diagram on Figure 3.8, we allocate only one secondary server per primary server in our model. It is a Standards requirement [Moc87a] that a primary DNS server in a production environment must have at least two secondary DNS servers for redundancy [Moc87a]. We do not conform to this requirement on purpose, based on the fact that we are not running a production IPv6 DNS and need not provide high availability of IPv6 resource records. In general, primary DNS servers should be backed up by secondary DNS servers for reliability and load distribution reasons. Since the population of IPv6 users at the university is low at the moment, the need for load distribution is negligible. Because we have established only two test IPv6 networks, introducing more than two servers offers protection only against host failures, but not against network failures, as the additional servers would have to be placed on the same networks with the existing servers. This does not introduce significant benefit. Also, there is no conceptual distinction between different secondary servers from the standpoint of the primary server. Therefore, it is sufficient, for the purposes of configuration and testing of the IPv6 DNS servers, to maintain only one secondary DNS server per primary DNS server.

### 3.2.3 Configuration of the IPv6 Domain Name System

Our main goals while configuring the DNS are to provide mapping between names and IPv6 addresses, to utilize more powerful modern IPv6 extensions to DNS, and to improve the ease of network renumbering. After the DNS test-servers were placed in operation, the first two goals have been accomplished. The third goal required more attention. All following sections discuss how we facilitate renumbering of networks.

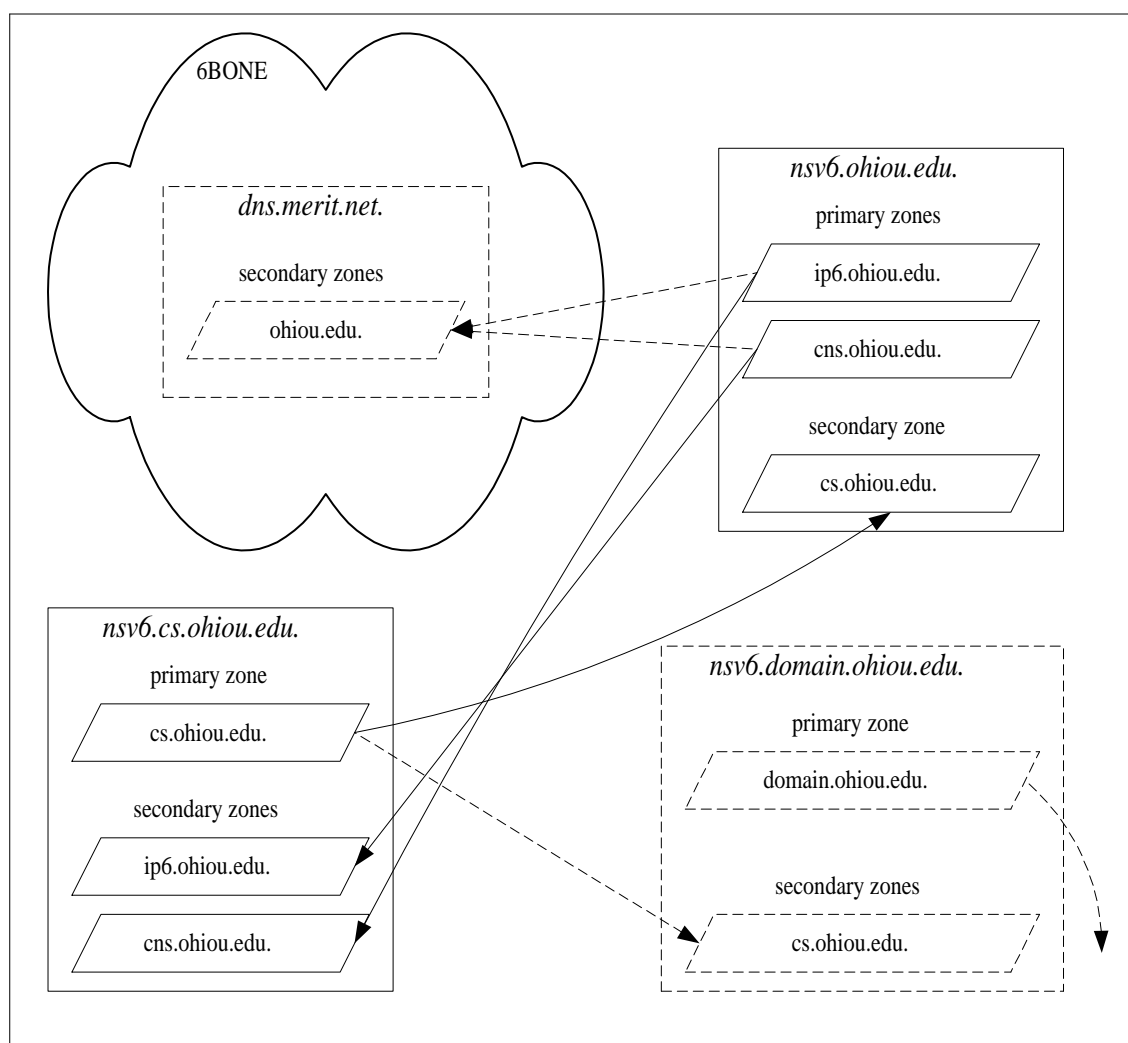


Figure 3.8 Hierarchy of DNS Test-Servers.

This figure shows hierarchical and functional relationship between the two DNS Test-Servers, as well as the future expansion plan (in dashed lines). Rectangles denote DNS servers, parallelograms denote the zones served by the DNS server, the arrows denote presence and direction of zone transfers (from the primary DNS server to the secondary). The parts of the figure depicted in dashed lines illustrate how the overall design may be expanded to accommodate more DNS servers and to integrate Ohio University IPv6 DNS service with the global IPv6 DNS.

### 3.2.3.1 Configuration of the IPv6 DNS Software

Configuration of IPv6 DNS servers was performed in accordance with the guidelines provided in the documentation for BIND-9.0.0b2. No specific issues of any concern were discovered. As a reference, BIND-9.0.0b2 configuration files for both DNS test-servers are supplied in Appendix C, along with the discussion of their contents.

### 3.2.3.2 Storing the Addressing Hierarchy within IPv6 DNS

In Section 3.1.3, we proposed an addressing hierarchy for the entire IPv6 address space available to Ohio University. As we have mentioned, although this hierarchy consists of five major levels, it is possible to merge certain levels into one level or split one level into two or more sublevels. Also, positioning of the levels within the 16-bit address space available to the university may vary in different parts of the hierarchy.

Let us represent the addressing hierarchy as a tree structure, with the root being the IPv6 prefix allocated to Ohio University by 6BONE. Any node on this tree can be identified by its own IPv6 prefix, which would be the concatenation of all bits on the 6BONE prefix and all bits on the path from the root of the tree to the node. The IPv6 prefix of any child node is the concatenation of the bits in the prefix of the node's parent node and the bits on the path from the parent node to the child node. This is a convenient representation of a node's prefix, because it hides the details of the addressing hierarchy, and eliminates any need for the child node to know about the hierarchy above its parent node. This isolates the child node from the transformations in the hierarchical levels that are above the parent node's level.

We propose that every node of the tree structure described in the previous paragraph is assigned a DNS name according to the position of the node on the tree relative to the root. All nodes already have an IPv6 address represented by their IPv6 network prefix, which is obtained by bit concatenation. The assignment of

names to those IPv6 prefixes gives us the ability to refer to an element of the addressing hierarchy without knowing its actual relative position, which is a convenient feature that facilitates renumbering of networks.

We propose that a new domain, `IP6.OHIOU.EDU`, be created to hold the DNS names that describe the addressing hierarchy. This single domain would naturally become a complete description of the topological structure of computer networks at Ohio University, and the addressing hierarchy would be hidden underneath it. There exists more IPv6-specific DNS information that will also be stored under this domain. However, host names should not be placed within `ip6.ohiou.edu`, the reasons being explained in Section 3.2.3.3.

### 3.2.3.3 Network Topology vs. DNS Topology

The addressing hierarchy or the network topology must be clearly separated from the DNS naming hierarchy. The two hierarchies should be completely independent of one another, as the former represents organization by physical networks, while the latter represents organization by logical units.

Let us consider a department that has domain name `department.ohiou.edu`. It is a logical unit. It may use several physical networks; for example, it could have two networks in `building A` and 3 networks in `building B`. `Building A` may be located in the northern area of the campus, while `building B` may be located in the southern area. We assume that the two buildings belong to completely separate sections of the addressing hierarchy tree. However, all hosts on those networks still logically belong to a single department. Therefore, a logical approach is to name the hosts in the form of `hostN.department.ohiou.edu`, independent of whether they are connected to networks in `building A` or in `building B`. Section 3.2.3.5 provides a complete description of how such architecture can be accomplished.

### 3.2.3.4 A6 and DNAME Resource Records

The A6 and DNAME resource records comprise the second generation of IPv6 extensions to DNS. They have been designed to provide DNS with support of the most modern specification of IPv6.

For name-to-address resolution, IPv6 DNS uses a new resource record, A6 [CHT00]. The textual format of this resource record presented in Figure 3.9. An IPv6 address for a domain name may be stored in one or several A6 resource records. A single A6 resource record may include a complete IPv6 address or only a portion of it along with the information leading to the remaining portion. This information comprises the prefix length (BITS) and the DNS name (REFERRAL) that defines the remaining prefix of the IPv6 address via the A6 resource record for that domain name. When the prefix length is zero, the entire IPv6 address is contained within the A6 resource record. Generally, there may be multiples levels of indirection, which need to be traversed to obtain a complete IPv6 address. The relevant performance implications are discussed in Section 3.2.3.6.

For the reverse lookups, IPv6 DNS uses a new resource record, DNAME [Cra99b], that relies on a new type of DNS label called “bit-string-label” [Cra99a]. The textual format of the DNAME resource record is shown in Figure 3.10. The contents of a DNAME resource record permits resource records to be stored at arbitrary bit-boundaries [CHT00]. The domain name IP6.INT is used to store resource records for reverse lookups. The bit-string-label compactly represents an arbitrary string of bits which is treated as a hierarchical sequence of one-bit domain labels. The DNAME resource record provides aliasing information for a specific bit-string-label that represents a portion of an IPv6 address. When the most significant portion of the IPv6 address that is being looked up matches the bit-string-label of a DNAME, the returned result provides the name of the domain that should be queried next. Next, the portion of the IPv6 address that corresponds to the matched bit-string-label is removed from the IPv6 address producing a new IPv6 address, and a DNS request

```

NAME                \
  TTL   TYPE  BITS  ADDRESS                REFERRAL
host1.domain.example. \
  21600 IN A6  0    3FFE:1CEF:0000:0000:1234:5678:9ABC:DEF0
host2.domain.example. \
  21600 IN A6  64   ::0FED:CBA9:8765:4321 network1.domain.example.
network1.domain.example. \
  21600 IN A6  48   0:0:0:1E05::          ip6.domain.example.
ip6.domain.example.   \
  21600 IN A6  0    3FFE:1CEF::

```

Figure 3.9 Textual Representation of the A6 Resource Record.

The IPv6 address of the domain name `host1.domain.example` is stored in a single A6 resource record. The IPv6 address of the domain name `host2.domain.example` is stored in three resource records. Note that the three resource records do not have to be in the same master file.

for that new IPv6 address is sent to the DNS server of the domain that should be queried next, according to the last response.

### 3.2.3.5 Description of the Master Files

In this section, we describe the contents of the master files that are used by our DNS test-servers. The master files that we composed contain operationally valid resource records. The master files provide both proper DNS configuration for the existing operational IPv6 networks<sup>10</sup> and demonstrate how to populate master files in accordance with the proposed addressing hierarchy. Our ability to use these master files in operation demonstrates IPv6 operability of BIND-9.0.0b2 and the correctness of our design and implementation.

For demonstration reasons, we use a only small portion of the entire university network. This portion includes two educational buildings (Stocker Center and Morton

<sup>10</sup>See network A and network B in Figure 2.1.

BIT-STRING-LABEL-NAME	TYPE	REFERRAL
\[x3FFE1CEF00000000123456789ABCDEF0/128].domain.example.	IN PTR	host1.domain.example.
\[x3FFE1CEF0000/48].ip6.int.	IN DNAME	ip6.domain.example.
\[x1E05/16].ip6.domain.example.	IN DNAME	domain.example.
\[x0FEDCBA987654321/64].domain.example.	IN PTR	host2.domain.example.

Figure 3.10 Textual Representation of the DNAME Resource Record.

The resource record for the domain name `host1.domain.example` is stored as a single PTR resource record. However, a more common implementation is the one shown for the domain name `host1.domain.example`. This example shows that after the first 48 bits of the IPv6 address are located via a search under the top-level domain `ip6.int`, the search continues within the domain `domain.example`, gradually obtaining the missing sections of the IPv6 address. Eventually, the search encounters the PTR resource record that contains the domain name and returns it in the answer.

Hall), one administrative building (HDL Center), and three student housing buildings (Atkinson House, Smith House, and True House). All of them are located on the main campus. For simplicity, we represent regional campuses as single points, without describing any details of their lower hierarchical levels.

We followed specific rules when assigning names to the network prefixes within the addressing hierarchy. Two rules help determine what name should be assigned to a network prefix. First, the name should be as short as possible, yet informative. The restriction on the length of a name is enforced by the fact that DNS uses UDP [Pos80] with the maximum packet size of 512 bytes<sup>11</sup>. This places limits on the amount of information that a DNS server may include in its answer to a DNS query. Second, an integer number should immediately follow the name of the prefix. For example, the proposed design makes it possible that a building uses more than one network prefix within the area in the cases when the building has more networks than one building

---

<sup>11</sup>This restriction is enforced to guarantee that DNS packets are never fragmented in the intermediate routers.

prefix provides. For such cases, a numerical suffix permits to distinguish different prefixes.

We identify two campus areas on the Athens campus. These two campus areas are Athens Buildings and College Greens<sup>12</sup>. We assign domain name `ath1-bld1` to the campus area Athens Buildings, and collect educational and administrative buildings under this area. We name the buildings within the area as `hd11`, `stocker1`, and `morton1`. Each building name is assigned its own 4-bit prefix. Next, we identify some of the networks within each building and name them, as illustrated in Figure 3.11 and described in Appendix B. Every network is assigned one of the network prefixes available under the building to which the network belongs. Because networks are assigned their prefixes within the prefix of a specific building, the names of these networks are considered to be subdomains of the domain of that building. The same concept applies to the buildings within areas, and so forth, along the hierarchy. For instance, the IRG network that is physically located in Stocker Center and is assigned one of network prefixes within Stocker Center building, has domain name `irg1.stocker1.ath1-bld1.ip6.ohiou.edu`. Another network, the network of the CS department located in Morton Hall has domain name `cs1.morton1.ath1-bld1.ip6.ohiou.edu`.

The next step is to assign domain names to the IPv6 addresses of the hosts located on the networks within buildings. We will use the IRG test-network<sup>13</sup> as an example. The domain name for the hosts on the IRG network is `cs.ohiou.edu`, similar to all machines in the CS department. Therefore, we store the A6 resource records for the IRG hosts in the master file for the domain `cs.ohiou.edu`. For each host, the A6 resource record contains only the host portion of host's entire IPv6 address, while the name of the IRG network, `irg1.stocker1.ath1-bld1.ip6.ohiou.edu`, is used as the referral. This way, when a DNS server locates the A6 RR for a host name, it then

---

<sup>12</sup>All student housing buildings belong to the College Greens. They are interconnected in the ResNet network.

<sup>13</sup>We described the test networks in Section 2.2.

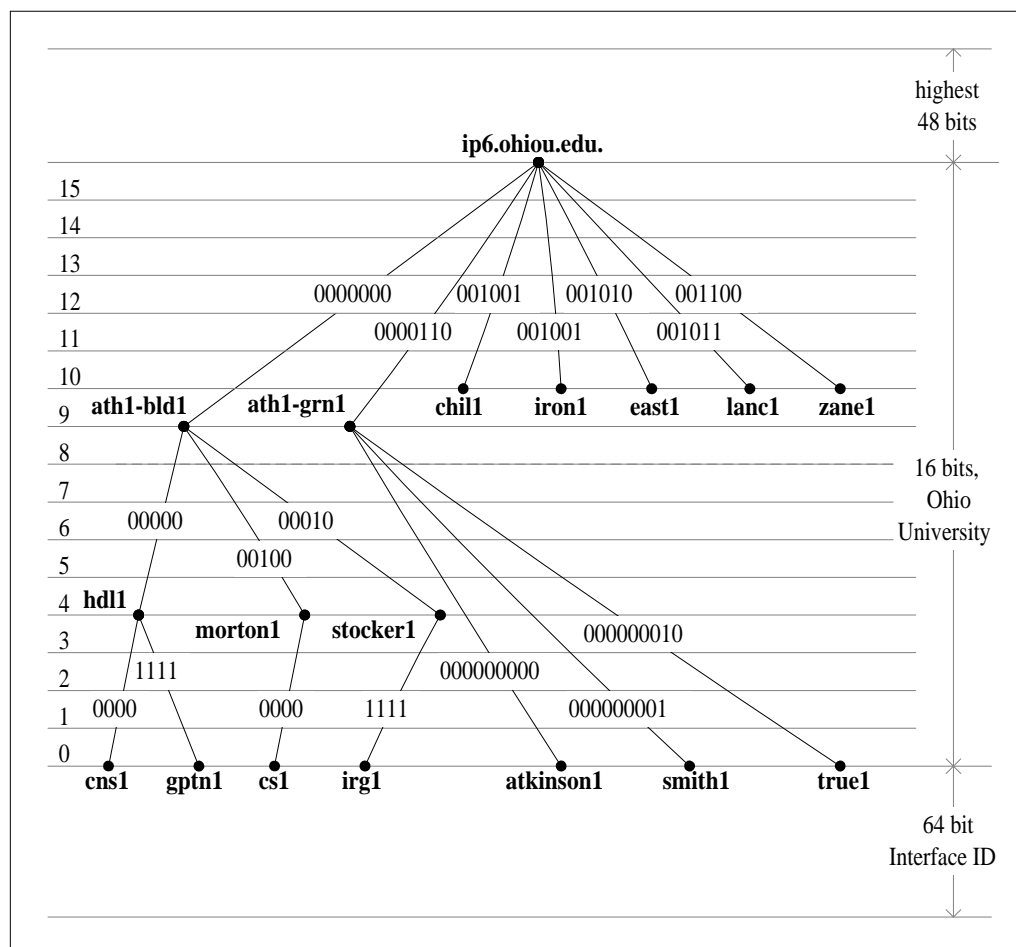


Figure 3.11 Assignment of DNS Names to Network Prefixes.

looks up the IPv6 address of the referral to consequently merge the two addresses and obtain the complete IPv6 address for the host [CHT00].

The use of A6 resource records described in the previous paragraph poses a potential problem. Consider the following extension to the example from the previous paragraph. The same master file for the domain `cs.ohiou.edu` stores A6 resource records for the hosts on the other network that belongs to CS department. Again, only the host portion of host's IPv6 address is stored in the A6 RR, but this time the A6 referral is set to `cs1.morton1.ath1-bld1.ip6.ohiou.edu`. The lookup procedure is the same as for the IRG network. Because a single master file stores A6 resource records for two different networks, and the A6 RRs contain only the host portion of each host's IPv6 address, it is possible that the same interface identifier be used on more than one network. Client DNS query will locate and return two IPv6 addresses in this case, one of which will be incorrect. However, this problem with the same interface identifiers is unlikely to occur because introduction of host autoconfiguration in IPv6 has eliminated any need for manual configuration of IPv6 addresses. The hardware identifiers of network interfaces which are used by stateless autoconfiguration [TN98] are globally unique, and therefore cannot exist on more than one network. For the supposedly rare cases of use of manual host configuration, care should be taken not to assign the same interface identifier on more than one network within a single DNS domain.

The proposed use of A6 resource records considerably facilitates changes in the network topology or, in other words, renumbering of networks. In the event of network renumbering, the master files for the domain, to which the network belongs, does not need to be changed. Only the relevant topological resource records within the domain `ip6.ohiou.edu` need to be changed to reflect the change of address of the network. If change of the physical location of the network must be performed, this will require changing the master file of the network's domain, because the referral name in the A6 resource records changes. However, a properly designed master file will require only

one change per network, not per host. This can be accomplished via use of the master file option “ORIGIN”. Refer to the master files for the domains `cs.ohiou.edu` and `cs.ohiou.edu` in Appendix B for the example.

Now we will consider a different area of the Athens campus, and illustrate more issues. We group all student housing buildings under area name `ath1-grn1`. Those buildings are a part of the ResNet network and are connected to the rest of the campus via the ResNet router. The buildings under area `ath1-grn1` are assigned the following names: `atkinson1`, `smith1`, `true1`. By concatenating this name with the names of the above prefixes, the name `atkinson1.ath1-grp1.ip6.ohiou.edu` is assigned to Atkinson House. In the existing network topology, only one network is in use in each student housing building. Therefore, allocation of 16 networks per building, according to the proposed addressing hierarchy, is not efficient. Instead, we merge the building and the network levels (refer to Figure 3.11), which gives us 9-bit block of network prefixes, for a total of 512 networks. This approach allows us to allocate more than one network to student housing buildings on demand, and not waste a significant number of network prefixes per building. This also facilitates routing, in the sense that we proposed in Section 3.1.2.1 to split the existing ResNet network into smaller networks that connect directly to the ResNet router. In addition, this case of the College Greens demonstrates ease of adjustment of the proposed addressing hierarchy to special needs.

### 3.2.3.6 Performance of DNS Lookups

This section provides a note on performance of DNS lookups when non-zero BITS A6 resource records are utilized. When IPv6 prefixes are “delegated upward” with A6 records, the number of DNS resource records required to establish a single IPv6 address increases by a non-trivial factor. Those records will typically, but not necessarily, come from different DNS zones (which can independently suffer failures).

When obtaining multiple IPv6 addresses, this increase in RR count will be proportionally less – and the total size of a DNS reply might even decrease – if the addresses are topologically structured. But the records could still easily exceed the space available in a UDP response which returns a large set of resource records [EB97]. The possibilities for overall degradation of performance and reliability of DNS lookups are numerous, and increase with the number of prefix delegations involved. However, these failures may be easily predicted by analyzing the specifics of a particular IPv6 address allocation scheme and IPv6 addressing hierarchy.

In the course of developing the IPv6 DNS topology described in Section 3.2.3.5, we ran into the problem of insufficiency of the size of a UDP datagram. The original design had the Athens campus given a name, similar to how DNS resource records in `ip6.ohiou.edu` for the regional campuses are configured now. Campus areas were descending from the name for the Athens campus. The IRG network had name `irg1.stocker1.bld1.ath1.ip6.ohiou.edu`. However, this architecture generated too many indirection records being returned in the answer. As a result, there was no room in the UDP datagram to return the complete answer for an address-to-name resolution query, and the query failed.

### 3.3 End-User Support for IPv6

Support of IPv6 by end-user applications is crucial to successful deployment of IPv6 in the computer networks at Ohio University. Unless the software that uses computer networks as a means of communication can utilize IPv6, the end-users will not be able to move towards IPv6. Clearly, all existing software cannot be quickly and easily converted to support IPv6. Therefore, only the software of the most importance to the end-users, the software absolutely necessary for businesses, and the most widely used software should be converted first. To run IPv6-enabled applications, the underlying operating system must understand IPv6. This section

provides information about the availability of IPv6 support in the existing applications and operating systems. As IPv6 becomes a standard part of operating systems and a more robust implementation, more production IPv6 services will start emerging. Provided availability of IPv6 to the end-users via software applications, this will contribute to the overall expansion of IPv6 on the Internet.

### 3.3.1 Software Available to The End-users

At the present time, a number of software applications have been adapted for IPv6 compatibility. A new network sockets API [GBST99] [ST98] was developed to provide IPv6 functionality to operating systems and applications. Conversion of an application to IPv6 consists primarily of replacing the IPv4 specific network-related system and library calls with their corresponding IPv6 equivalents. The most widely used networking applications are web servers and browsers, email clients and transport agents, and remote access software.

#### Remote Access

Secure Shell version 1 (SSH1) is available on any Unix operating system platform that supports IPv6. We have compiled it and successfully run it to test its operability. A popular File Transfer Protocol (FTP) [PR85] [AMO98] client, NcFTP, is available on Unix and Windows platforms.

#### Web Access

Apache web server version 1.3.11 has IPv6 support via applying a patch. There is another web server thttpd that originally supports IPv6. The text-based browser w3c originally supports IPv6. The other three available Web browsers that can be converted to IPv6 are Mozilla, Lynx 2.8.2, and wMosaic. They need to be patched.

#### Email Access

Sendmail version 8.9.1 can be patched to support IPv6. Sendmail version 8.10.0

comes with the IPv6 capability by default. Qmail can be enabled to support IPv6. At this point in time, there are no windows-based email clients that support IPv6. However, it is believed that Microsoft Windows Outlook will be capable of IPv6 operation once the operating system understands IPv6.

### 3.3.2 Unix Operating System

Unix is the operating system that is most commonly used for research. Most flavors of Unix are also under the Open Source License. This is why IPv6 support is already enabled in most of the existing Unix operating systems and has undergone considerable testing and refinement. In particular, Linux kernels version 2.2 and above ship with an IPv6 implementation built in, Solaris 8 and BSDI BSD/OS v4.0 ship with an IPv6 implementation built in. FreeBSD has been supporting IPv6 for the last few releases, and will have it integrated with the kernel in all versions after version 3.4.

We configured a number Solaris 8 machines with IPv6 support. The task is relatively trivial when the operating system supports IPv6 by default. The network interfaces on the machine autoconfigure their IPv6 addresses according to the specification of stateless host autoconfiguration [TN98]. There are three types of addresses in use on our test machines: the loopback address, the link-local address, and the globally unique unicast address. The latter two are obtained by combining the MAC manufacturer ID of the network interface with the network prefix. The network prefix is fixed for the link-local address, and is learned from a local IPv6 router for the global unicast address.

We tested operability of host renumbering by changing configuration of the IPv6 router. IPv6 addresses on the machines change shortly after the network prefix advertised by the router is changed.

### 3.3.3 Microsoft Windows Operating System

Although most of the research work is conducted on Unix operating systems, the vast majority of end-users use Microsoft Windows or Macintosh operating systems on their desktop computers. Therefore, these operating systems should provide support for IPv6 to aid contribute to the expansion of IPv6 among the end-user population. Microsoft Windows does not currently ship with a built-in IPv6 implementation. However, Microsoft has an add-on IPv6 implementation for Windows 2000 available for developer use. Microsoft Research (MSR) has an add-on IPv6 implementation for Windows 2000 and NT 4 available. Trumpet also has an IPv6 Winsock implementation available for Windows, covering all versions from Windows 3.1 through Windows 98.

### 3.3.4 Macintosh Operating System

Macintosh operating system has support for IPv6 only in Mac OS X. IPv6 is shipped as a part of the kernel. All versions of the operating system prior to Mac OS X neither support IPv6 in the kernel nor can they be patched to enable IPv6.

## 4. CONCLUSIONS

We have established a basic IPv6 environment at Ohio University to improve certain weaknesses in the existing computer networks in the university. However, the IPv6 environment that we setup does not qualify as a production environment, largely because the implementations of IPv6 that are available at this time for operating systems and routers are not yet sufficiently robust for production use. Furthermore, changes to all networking applications have to be made by their vendors prior to converting the developed IPv6 environment into a production environment. However, as the quality of the implementation of IPv6 increases, certain parts of the IPv6 environment, as well as the IPv6 services, will eventually reach a production quality. At the same time, the IPv6 environment may be expanded and new IPv6 components may be introduced.

### 4.1 Improvements with Internet Protocol Version 6

The new addressing hierarchy that we designed eliminated certain weaknesses of the computer networks at Ohio University and provided better scalability of the address space and efficient use of route aggregation. The addressing hierarchy strongly relies on use of route aggregation that is a requirement for IPv6. The weaknesses that were improved by the addressing hierarchy are complexity of network management, inefficiency in assignment of IP addresses, and low granularity of subnets assignment. The complexity of network management was decreased by introducing a structured assignment of addresses. The inefficiency of assignment of IP addresses ceased being an issue with introduction of IPv6. Our proposed solution for the low granularity

of networks provided theoretically better network architecture and improved network traffic flow. The new portions of the university will support only IPv6. In an ideal case, both IPv4 and IPv6 addresses should be allocated to the new portions of the network. In reality, at a certain point in time, no IPv4 addresses will be available. Therefore, the new portions of the university network will not understand IPv4.

IPv6 DNS service was not easy to configure and integrate with the existing IPv4 environment, because it is in its early stage of implementation. IPv6 DNS was configured only partially, to support the most recent functionality. No operational tests could be performed for the lack of necessary testing tools. However, correctness of the configuration presented in the paper was successfully tested. Use of IPv6 DNS in an operational environment will become possible only with appearance of proper client IPv6 DNS tools.

Support of IPv6 by operating systems is not very complete and robust. There is little uniformity in the way different operating systems provide IPv6. The support is vendor-driven, which potentially does not guarantee cooperation between the implementors. IPv6 needs more testing before it can become a standard part of every operating system.

Adaptation of software applications for IPv6 is in an early stage. Only a few applications have been converted. Again, the reason being that there has not been a standardized IPv6 among different operating systems. Although IPv6 APIs are proposed standards, some other parts of the specification are still in development.

## 4.2 Future Work

This section describes the work that can be performed in the future to further improve functionality of the computer networks at Ohio University by introducing more IPv6 features.

Stateful host autoconfiguration permits assignment of specific IPv6 addresses to network nodes, bypassing or as an alternative to the stateless host autoconfiguration. Stateful configuration takes into consideration the fact that many non-IP parameters exist, which need to be assigned as parts of the host configuration process. Therefore, it may be useful to implement DHCPv6 that has support for IPv6 addresses, after an implementation becomes available.

Dynamic DNS and secure DNS may be deployed to contribute to the ease of host and network renumbering and security of DNS operations. Dynamic DNS capability allows a network node to send queries to the DNS server requesting the server to update the resource records corresponding to that node. This capability decreases the amount of work that DNS administrators need to perform, and helps keep DNS resource records updated in on a regular basis. Secure DNS implements secure and authenticated queries between DNS clients and servers. It is necessary to insure that network nodes may request updates of only those DNS resource records to which the nodes are authorized. Therefore, employment of Secure DNS, in conjunction with Dynamic DNS Updates, should be considered for employment in the computer networks at Ohio University.

We suggest that necessary research and experiments be performed in the area of Mobile IP. IPv6 introduces new grounds for a better and more efficient implementation of Mobile IP. Mobile IP is now not an upgrade to the protocol, but an integrated part of it.

## BIBLIOGRAPHY

## BIBLIOGRAPHY

- [AMO98] M. Allman, C. Metz, and S. Ostermann. FTP Extensions for IPv6 and NAT, September 1998. RFC 2428.
- [Bra89] R. Braden. Requirements for Internet Hosts – Application and Support, October 1989. RFC 1123.
- [CHT00] M. Crawford, C. Huitema, and S. Thomson. DNS Extensions to Support IPv6 Address Aggregation and Renumbering, March 2000. (work in progress).
- [Com95] Douglas E. Comer. *Internetworking with TCP/IP, Volume I, Principles, Protocols, and Architecture*. Prentice Hall, 3rd edition, 1995.
- [Con00] Internet Software Consortium. ISC Web Site, 2000. URL <http://www.isc.org>.
- [Cra99a] M. Crawford. Binary Labels in the Domain Name System, August 1999. RFC 2673.
- [Cra99b] M. Crawford. Non-Terminal DNS Name Redirection, August 1999. RFC 2672.
- [Dro97] R. Droms. Dynamic Host Configuration Protocol, March 1997. RFC 2131.
- [Eas98] D. Eastlake. Domain Name System Security Extensions, March 1998. RFC 2535.
- [EB97] R. Elz and R. Bush. Clarifications to the DNS Specification, July 1997. RFC 2181.
- [Fin00] W. Fink. 6BONE Web Site, 2000. URL <http://www.6bone.net>.
- [FLYV93] V. Fuller, T. Li, J. Yu, and K. Varadhan. Classless Inter-Domain Routing (CIDR), September 1993. RFC 1519.
- [GBST99] R. Gilligan, J. Bound, W. Stevens, and S. Thomson. Advanced Sockets API for IPv6, March 1999. RFC 2553.

- [GN96] R. Gilligan and E. Nordmark. Transition Mechanisms for IPv6 Hosts and Routers, April 1996. RFC 1933.
- [Gro92] P. Gross. Choosing a “Common IGP” for the IP Internet, October 1992. RFC 1371.
- [HD98] R. Hinden and S. Deering. IP Version 6 Addressing Architecture, July 1998. RFC 2373.
- [HOD98] R. Hinden, M. O’Dell, and S. Deering. An IPv6 Aggregatable Global Unicast Address Format, July 1998. RFC 2374.
- [Hui94] C. Huitema. The H Ratio for Address Assignment Efficiency, November 1994. RFC 1715.
- [Ma198] G. Malkin. RIP Version 2, November 1998. RFC 2453.
- [Moc87a] P. Mockapertis. Domain Names – Implementation and Specification, November 1987. RFC 1035.
- [Moc87b] P. Mockapertis. Domain Names - Concepts And Facilities, November 1987. RFC 1034.
- [Moy98] J. Moy. OSPF Version 2, April 1998. RFC 2328.
- [Pos80] J. Postel. User Datagram Protocol, August 1980. RFC 768.
- [Pos81] J. Postel. Internet Protocol, September 1981. RFC 791.
- [PR85] J. Postel and J. Reynolds. File Transfer Protocol (FTP), October 1985. RFC 959.
- [ST98] W. Stevens and M. Thomas. Advanced Sockets API for IPv6, February 1998. RFC 2292.
- [TH95] S. Thomson and C. Huitema. DNS Extensions to support IP version 6, December 1995. RFC 1886.
- [TN98] S. Thomson and T. Narten. IPv6 Stateless Address Autoconfiguration, December 1998. RFC 2462.

## APPENDIX

## A. CONNECTION TO 6BONE

In this appendix, we describe why Merit, Inc. was selected to be the 6BONE pTLA for Ohio University. There are practical routing considerations of the underlying IPv4 path that will be in use. In particular, as IPv6 packets are carried over an IPv4 tunnel, the route they take is determined by the network topology between the two ends of the tunnel. Therefore, a new site should seek to attach to a 6BONE point that is reasonably adjacent to the site’s IPv4 paths into the Internet. It is best to establish a tunnel with a site that is topologically close. A typical determination of closeness is expressed in terms of number of hops and the ping time between two network nodes.

We performed an analysis of closeness of most of the 6BONE sites. The results are supplied in Table A.1. Based on obtained measurements, it is clear that MERIT/US-MI<sup>1</sup> is the best candidate for establishing a connection with 6BONE. MERIT/US-MI can be reached via the least number of hops and within the least amount of time.

pTLA Name	pTLA’s Endpoint for Tunnels	Distance, hops	Ping, ms
INNER/US-VA	avarice.inner.net	12	66.5
ESNET/US	esnet-v6r1.es.net	15	70.1
ISI-LAP/US-CA	bah.isi.edu	15	94.3
VERIO/US	sea-ipv6.verio.net	19	138.2
<b>MERIT/US-MI</b>	<b>6bone.merit.edu</b>	<b>10</b>	<b>28.9</b>
SPRINT/US	sl-bb1-6bone.sprintlink.net	16	49.2
DIGITAL-CA/US	ipv6-gw1.pa-x.dec.com	14	103.8

Table A.1 Measurements of Closeness of 6BONE pTLAs.

---

<sup>1</sup>Symbolic names of pTLAs are available at <http://www.6bone.net>.

## B. IPv6 DNS MASTER FILES

This appendix provides master files for the new zones and for the zone that contain IPv6-enabled hosts. The zone `ip6.ohiou.edu` is a new zone that we created to hold information about Ohio University network topology. The zone `ip6.int` is a temporary zone that we had to create for testing operability of A6 and DNAME records, because the production top-level domain `ip6.int` does not support these resource records. The zones `cns.ohiou.edu` and `cs.ohiou.edu` are exported from production DNS servers, their IPv4 resource records are merged with the IPv6 resource records for IPv6-enabled hosts that are configured within those zones. For these two zones, we provide only the master files that contain the IPv6 resource records, because IPv4 resource records are outside of our control.

### B.1 Master File for Zone `ip6.int`.

```
$TTL 21600
@           IN      SOA    nsv6.cns.ohiou.edu. \
                    ipv6adm.nsv6.cns.ohiou.edu. (
                    2000042700      ; Serial
                    86400           ; Refresh (1 day)
                    3600            ; Retry   (1 hour)
                    3600000         ; Expire  (1000 hours)
                    21600 )         ; Minimum (6 hours)
                    IN      NS    nsv6.cns.ohiou.edu.
                    IN      MX    10 centaur.cns.ohiou.edu.
;
$ORIGIN ip6.int.
\[x3FFE1CEF0000/48] DNAME      ip6.ohiou.edu.
```

## B.2 Master File for Zone ip6.ohiou.edu

```

$TTL      21600
@          IN      SOA      nsv6.cns.ohiou.edu. \
                    ipv6adm.nsv6.cns.ohiou.edu. (
                    2000042700      ; Serial
                    86400           ; Refresh (1 day)
                    3600            ; Retry   (1 hour)
                    3600000         ; Expire  (1000 hours)
                    21600          )      ; Minimum (6 hours)

        IN      NS      nsv6.cns.ohiou.edu.
        IN      NS      nsv6.cs.ohiou.edu.
        IN      MX      10 centaur.cns.ohiou.edu.
        IN      A6 0    3ffe:1cef::0

$ORIGIN ip6.ohiou.edu.

;           Main Campuses Topology
;-----
; Athens
;-----
; Buildings
ath1-bld1      IN      A6 48    0:0:0:0000::0      @
\[x00/7]      DNAME   ath1-bld1      ; \[b0000 000-]
; Greens
ath1-grn1     IN      A6 48    0:0:0:1C00::0      @
\[x1C/7]     DNAME   ath1-grn1     ; \[b0001 110-]

; Chillicothe
;-----
chil1         IN      A6 48    0:0:0:2000::0      @
\[x20/6]     DNAME   chil1         ; \[b0010 00--]

; Ironton
;-----
iron1        IN      A6 48    0:0:0:2400::0      @
\[x24/6]     DNAME   iron1        ; \[b0010 01--]

; Eastern
;-----
east1        IN      A6 48    0:0:0:2800::0      @
\[x28/5]     DNAME   east1        ; \[b0010 10--]

```

```

; Lancaster
;-----
lanc1          IN          A6 48      0:0:0:2C00::0      @
\[x2C/5]      DNAME      lanc1          ; \[b0010 11--]

; Zanesville
;-----
zane1          IN          A6 48      0:0:0:3000::0      @
\[x30/5]      DNAME      zane1          ; \[b0011 00--]

;                Athens Campus
;=====
; Athens Buildings (ath1-bld1)
;=====
$ORIGIN ath1-bld1.ip6.ohiou.edu.
; HDL Center
\[x00/5]      DNAME      hd11          ; \[b0000 0---]
hd11          IN          A6 55      0:0:0:0000::0      @
; Stocker Center
\[x10/5]      DNAME      stocker1      ; \[b0001 0---]
stocker1      IN          A6 55      0:0:0:0020::0      @
; Morton Hall
\[x20/5]      DNAME      morton1       ; \[b0010 0---]
morton1       IN          A6 55      0:0:0:0040::0      @

; HDL Center
; -----
$ORIGIN hd11.ath1-bld1.ip6.ohiou.edu.
\[x0/4]       DNAME      cns.ohiou.edu. ; \[b0000]
cns           IN          A6 60      0:0:0:0000::0      @
\[xF/4]       DNAME      cns.ohiou.edu. ; \[b1111]
gptn          IN          A6 60      0:0:0:000F::0      @
; Stocker Center
; -----
$ORIGIN stocker1.ath1-bld1.ip6.ohiou.edu.
\[xF/4]       DNAME      cs.ohiou.edu.  ; \[b1111]
irg           IN          A6 60      0:0:0:000F::0      @
; Morton Hall
; -----
$ORIGIN morton1.ath1-bld1.ip6.ohiou.edu.
\[x0/4]       DNAME      cs.ohiou.edu.  ; \[b0000]

```

```

cs                IN          A6 60    0:0:0:0000::0        @

; Athens Greens (ath1-grn1)
;=====
$ORIGIN ath1-grn1.ip6.ohiou.edu.
; Atkinson
\[x000/9]         DNAME      dorm.ohiou.edu.    ; \[b0000 0000 0---]
; Smith
\[x008/9]         DNAME      dorm.ohiou.edu.    ; \[b0000 0000 1---]
; True
\[x010/9]         DNAME      dorm.ohiou.edu.    ; \[b0000 0001 0---]

```

### B.3 Master File for Zone cns.ohiou.edu

```

; IPv6 Serial Number: 2000042700

; Direct Mapping
;=====

; CNS, HDL Center
$ORIGIN cns.hdl1.ath1-bld1.ip6.ohiou.edu.
unicorn.cns.ohiou.edu. IN      A6 64    0::0A00:20FF:FE9E:97F2 @

; GPTN, HDL Center
$ORIGIN gptn.hdl1.ath1-bld1.ip6.ohiou.edu.
centaur.cns.ohiou.edu. IN     A6 64    0::0A00:20FF:FEC1:6815 @
test1.cns.ohiou.edu.   IN     A6 64    0::1234:5678:9ABC:DEF0 @

; Reverse Mapping
;=====
$ORIGIN cns.ohiou.edu.

; GPTN, HDL Center
\[x0A0020FFFEC16815/64] PTR    centaur
\[x123456789ABCDEF0/64] PTR   test1

; CNS, HDL Center
\[x0A0020FFFE9E97F2/64] PTR   unicorn

```

## B.4 Master File for Zone cs.ohiou.edu

```

; IPv6 Serial Number: 2000042701

; Direct Mapping
;=====

; IRG Lab, Stocker
$ORIGIN irg.stocker1.ath1-bld1.ip6.ohiou.edu.
6bone.cs.ohiou.edu.      IN      A6 64   0::0000:0cff:fe76:3dc6  @
jarok.cs.ohiou.edu.      IN      A6 64   0::0800:20ff:fe22:466a  @
vger.cs.ohiou.edu.       IN      A6 64   0::0800:20ff:feb0:ce33  @
picard.cs.ohiou.edu.     IN      A6 64   0::0800:20ff:fe85:af9e  @
thrawn.cs.ohiou.edu.     IN      A6 64   0::0800:20ff:fe7a:880b  @
thoth.cs.ohiou.edu.      IN      A6 64   0::0800:20ff:fe7b:596c  @
pride.cs.ohiou.edu.      IN      A6 64   0::0800:20ff:fe7b:5983  @
auden.cs.ohiou.edu.      IN      A6 64   0::0800:20ff:fe7a:dcdc  @

; CS Dept, Morton
$ORIGIN cs.morton1.ath1-bld1.ip6.ohiou.edu.
boss.cs.ohiou.edu.       IN      A6 64   0::1234:5678:9ABC:0000  @
oucsace.cs.ohiou.edu.    IN      A6 64   0::1234:5678:9ABC:1111  @
prime.cs.ohiou.edu.      IN      A6 64   0::1234:5678:9ABC:2222  @

; Reverse Mapping
;=====
$ORIGIN cs.ohiou.edu.

; IRG Lab, Stocker
\[x00000CFFFE763DC6/64] PTR      6bone
\[x080020FFFE22466A/64] PTR      jarok
\[x080020FFFEBOCE33/64] PTR      vger
\[x080020FFFE85AF9E/64] PTR      picard
\[x080020FFFE7A880B/64] PTR      thrawn
\[x080020FFFE7B596C/64] PTR      thoth
\[x080020FFFE7B5983/64] PTR      pride
\[x080020FFFE7ADCDC/64] PTR      auden

; CS Dept, Morton
\[x123456789ABC0000/64] PTR      boss
\[x123456789ABC1111/64] PTR      oucsace
\[x123456789ABC2222/64] PTR      prime

```

### C. BIND-9.0.0b2

This appendix provides a discussion of the contents of the configuration files that we used to place BIND-9.0.0b2 in operation. The configuration files used by our two DNS test-servers are very similar, to include configuration file only for the central DNS test-server. The only difference is between the sets of the primary and the secondary zones configured for the DNS test-servers. We specify two different logging channels to separate operational messages and messages about the periodic zone transfers. We permit the server to listen to DNS queries both on its IPv4 and IPv6 interfaces (IP addresses). Declaration of zone “.” informs the server which file should be read to obtain the a list of root nameservers that should be contacted to retrieve the most recent list of the root nameservers. Information for master zones `cns.ohiou.edu` and `ip6.ohiou.edu` should be read from master files `master/cns.ohiou.edu` and `master/cs.ohiou.edu` respectively. Zone transfers for zone `cs.ohiou.edu` should be performed from the DNS server at IPv4 address 132.235.3.132. Below is the BIND-9.0.0b2-style configuration file:

```
logging {
    channel named_log {
        file "logs/named.log";
        print-time yes;
        print-category yes;
        print-severity yes;
        severity debug 1; };
    channel xfer_log {
        file "logs/xfer.log";
        print-time yes;
        print-category yes;
        print-severity yes;
        severity debug 1; };
```

```
category load { named_log; default_debug; };
category xfer-in { xfer_log; };
category xfer-out { xfer_log; };
category default { named_log; default_debug; };
};
options {
    directory "/var/named";
    pid-file "named.pid";
    listen-on {
        132.235.79.211;          // listen only one these two
        3ffe:1cef:0:1:a00:20ff:fec1:6815; }; // interfaces
    transfer-format many-answers;
    statistics-file "named.stats";
    auth-nxdomain yes;
};
zone "." {
    type hint;
    file "named.root";
};
zone "ip6.ohiou.edu" {
    type master;
    file "master/ip6.ohiou.edu";
};
zone "cns.ohiou.edu" {
    type master;
    file "master/cns.ohiou.edu";
};
zone "ip6.int" {
    type master;
    file "master/ip6.int";
};
zone "cs.ohiou.edu" {
    type slave;
    file "slave/cs.ohiou.edu";
    masters { 132.235.3.132; }; // vger.cs.ohiou.edu
};
```

## D. ADJUSTMENT OF DNS SOA RESOURCE RECORDS

DNS uses a serial number<sup>1</sup> to keep track of changes in the master file [Moc87a]. An increase of the value of the serial number flags that the zone information has been changed, consequently triggering zone transfers to the secondary DNS servers for the zone. A DNS server may control more than one zone, and each zone under control of a DNS test-server must separately maintain its serial number. Our design<sup>2</sup> is such that there always are two serial numbers present for such IPv6-enabled zone. One serial number comes with the IPv4 zone transfer performed in the first step – the value of this serial number is maintained by the administrator of the production DNS server. The other serial number reflects the changes to the IPv6 resource records – it is stored in the IPv6 master file. The values of the two serial numbers change independently from one another. These two serial numbers must be merged into a single serial number that would then be used by the DNS test-server. Our software retrieves the serial numbers for the zone, detects changes in their values, and uses built-in logic to determine whether the zone records have changed and the DNS test-server must be reloaded. As an example, even when no changes have been made to the IPv6 resource records, the IPv4 resource records on the production DNS server for that domain might have changed. When this is the case, the master file must be rebuilt and the DNS test-server must reload it.

We illustrate the results of the adjustments of DNS SOA resource records in Figure D.1 and Figure D.2. Figure D.1 demonstrates what data the SOA resource

---

<sup>1</sup>The field SERIAL of the SOA record [Moc87a].

<sup>2</sup>Refer to Section 3.2.1.5.

record contains before the adjustment, and Figure D.2 contains the adjusted values that are discussed in Section 3.2.1.5.

```

; BIND version in.named LOCAL-990914.09/13/99M28
;                               Tue Sep 14 17:32:28 PDT 1999
; zone 'cns.ohiou.edu'  last serial 0
; from 132.235.64.1  at Wed May  3 01:00:01 2000
$ORIGIN ohiou.edu.
cns      IN      SOA      watson.cns.ohiou.edu. \
                               watkins.watson.cns.ohiou.edu. (
2000041300 86400 43200 604800 86400 )
      IN      NS      watson.cns.ohiou.edu.
      IN      NS      holmes.cns.ohiou.edu.
      IN      NS      boss.cs.Ohiou.Edu.
      IN      NS      oucsace.cs.Ohiou.Edu.
      IN      NS      dns2.cso.uiuc.edu.
      IN      NS      dns1.cso.uiuc.edu.
      IN      HINFO   "Communication Network Services" "Scott Quad"
      IN      HINFO   ".cns.ohiou.edu" "132.235.64.x, ...210.x"
      IN      A       132.235.197.200
      IN      MX      100 watkins2.cns.OhioU.Edu.
$ORIGIN cns.OhioU.edu.
chubb-mdf-s1  IN      A       132.235.10.253
dhcp-232-015  IN      A       132.235.232.15
[ ... ]
rtech-311-h1  IN      A       132.235.56.248
rtech-311-h2  IN      A       132.235.56.249

```

Figure D.1 Original Contents of the SOA Resource Record.

```

$TTL 21600
; BIND version in.named LOCAL-990914.09/13/99M28
;                               Tue Sep 14 17:32:28 PDT 1999
; zone 'cns.ohiou.edu'  last serial 0
; from 132.235.64.1  at Thu Apr 27 01:51:08 2000
$ORIGIN ohiou.edu.
cns      IN      SOA      nsv6.cns.ohiou.edu. \
                               ipv6adm.nsv6.cns.ohiou.edu. (
2000042602 86400 43200 604800 86400 )
      IN      HINFO     "Communication Network Services" "Scott Quad"
      IN      HINFO     ".cns.ohiou.edu" "132.235.64.x, ...210.x"
      IN      A         132.235.197.200
      IN      NS        nsv6.cns.ohiou.edu.
      IN      NS        nsv6.cs.ohiou.edu.
      IN      MX        100 watkins2.cns.OhioU.Edu.
$ORIGIN cns.OhioU.edu.
chubb-mdf-s1  IN      A         132.235.10.253
dhcp-232-015  IN      A         132.235.232.15
[ ... ]
rtech-311-h1  IN      A         132.235.56.248
rtech-311-h2  IN      A         132.235.56.249
;
$INCLUDE /var/named/master/cns.ohiou.edu-ipv6

```

Figure D.2 Adjusted Contents of the SOA Resource Record.

CHIPITSYN, VITALI. M.S. June, 2000  
Electrical Engineering

Improvement of Ohio University Computer Networks with Internet Protocol Version 6  
(73 pp.)

Director of Thesis: Shawn D. Ostermann

This thesis demonstrates how Internet Protocol Version 6 (IPv6) may be used to improve certain operational parameters of a large Internet site. The computer network at Ohio University is used as the practical application of IPv6. The thesis outlines the weaknesses that exist in the university network and practical solutions to eliminate those weaknesses. Extension of IP address space is easily accomplished with IPv6. An addressing hierarchy different from the one that exists in the university is introduced to improve management of the computer network and make efficient use of route aggregation capability of IPv6. The Domain Name System (DNS) service is considered to be a very important network service; hence, IPv6 DNS service was integrated with the production environment of the university. The thesis provides detailed demonstration of configuration of a multi-level infrastructure of DNS service. Use of IPv6 DNS allows better utilization of host autoconfiguration capability provided by IPv6, and contributes to the ease of networks renumbering. In this paper, we also perform analysis of IPv6 functionality available to the end-users on a variety of platforms. Finally, this thesis considers how other operational parameters of the computer network may be improved by application of a wider functionality offered by IPv6 to the computer networks at Ohio University.

Approved: \_\_\_\_\_