

Porting Applications to IPv6

Yukwen Hsu

Microsoft Corporation – Windows Division

TWNIC 2001 IPv6 Conference

Microsoft

Agenda

- ✍ Why IPv6
- ✍ Technology Highlights
- ✍ Development Issues
- ✍ Getting Started
- ✍ Q & A

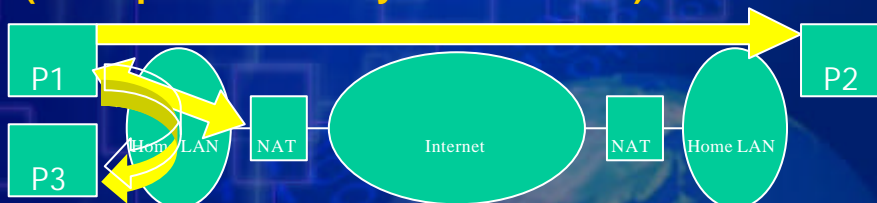
Microsoft

Why IPv6 is Important ?

- ✂ IPv4 address space become exhausted
- ✂ The need for simpler configuration
- ✂ IP-layer security (IPSec)
- ✂ Quality of Service (QOS)
- ✂ End-to-end address transparency
- ✂ Mobility
- ✂ Selected by 3GPP

Microsoft

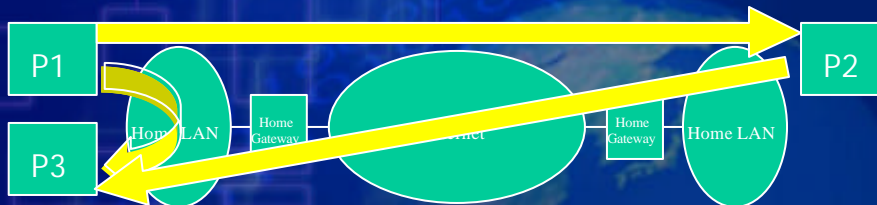
Scenario 1: Multiparty game (Example: DirectPlay based AoE-II)



- ✂ With NAT, complex and brittle software:
 - ✂ 2 Addresses, inside and outside
 - ✂ P1 provides “inside address” to P3, “outside address” to P2
 - ✂ Need to recognize inside vs. outside
 - ✂ P1 does not know outside address of P3 to inform P2

Microsoft

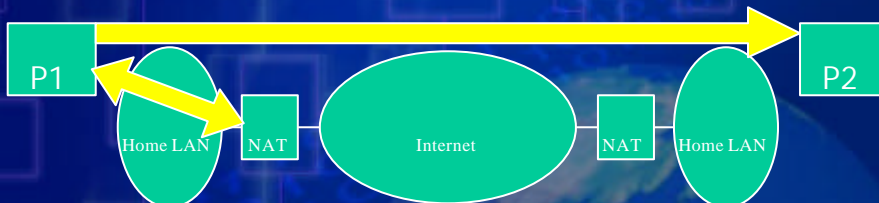
Scenario 1: Multiparty game (Example: DirectPlay based AoE-II)



- ✦ **With IPv6:**
 - ✦ Just use IPv6 addresses

Microsoft

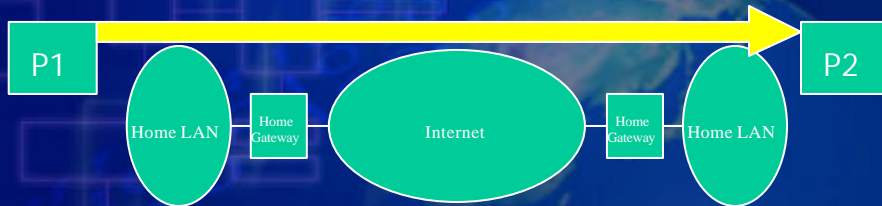
Scenario 2: Peer-to-peer (Example: RTC or file retrieval)



- ✦ **With NAT:**
 - ✦ Need to learn the address "outside the NAT"
 - ✦ Provide that address to peer
 - ✦ Need either NAT-aware application, or application-aware NAT
 - ✦ May need a third party registration server to facilitate finding peers

Microsoft

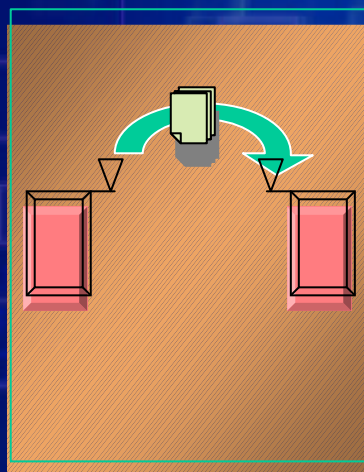
Scenario 2: Peer-to-peer (Example: RTC or file retrieval)



- ✦ With IPv6:
- ✦ Just use IPv6 address

Microsoft

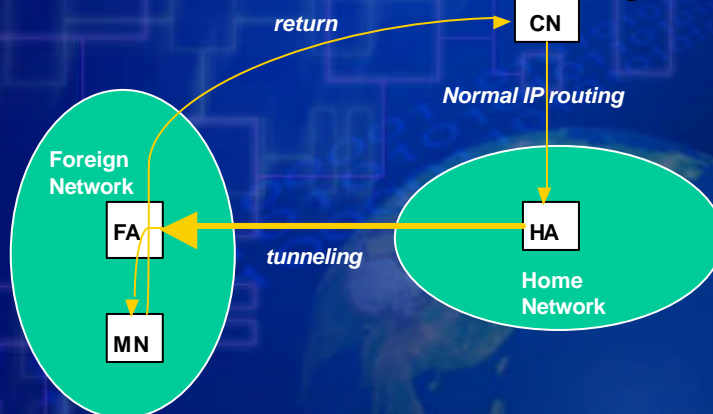
Scenario 3: Ad-hoc networking (Example: Pocket PC)



- ✦ IPv4: media lock + 63 sec.
 - ✦ Try DHCP
 - ✦ Wait for timeout
 - ✦ Select AutoNet address
 - ✦ Conflict detect
- ✦ IPv6: media lock + 1 sec.
 - ✦ Configure using MAC
 - ✦ Conflict detect

Microsoft

Scenario 4: IP Mobility

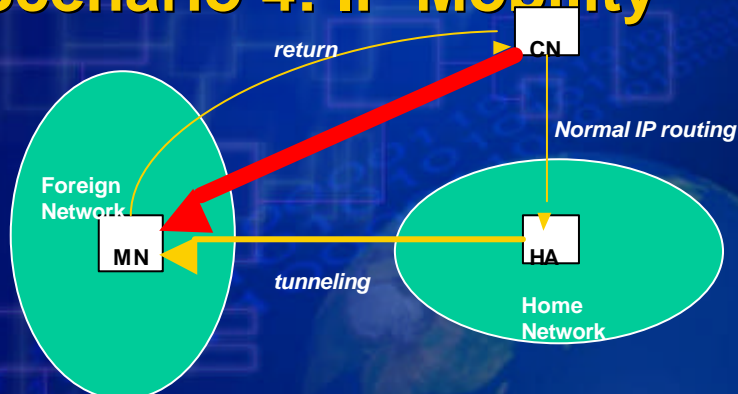


Base Mobile IPv4:

- ✦ Packets are relayed through the HA
- ✦ Single point of failure (HA)
- ✦ Triangle routing, ingress filtering issues

Microsoft

Scenario 4: IP Mobility



Mobile IPv6:

- ✦ Packets take direct path
- ✦ No single point of failure (HA)
- ✦ Solve triangle routing, ingress filtering issues

Microsoft

Technology Highlights

- ✦ Addressing model
- ✦ Differences between IPv4 & IPv6

Microsoft

Addressing Model

- ✦ Addresses are assigned to interfaces
 - ✦ No change from IPv4 Model
- ✦ Interface can have multiple addresses
- ✦ Addresses have scope
 - ✦ Link Local
 - ✦ Site Local
 - ✦ Global



Microsoft

Differences Between IPv4 and IPv6

Feature	IPv4	IPv6
Address length	32 bits	128 bits
IPSec support	Optional	Required
QoS support	Some	Better
Fragmentation	Hosts and routers	Hosts only
Checksum in header	Yes	No
Options in header	Yes	No
Link-layer address resolution	ARP	Multicast Neighbor Discovery Messages
Uses broadcasts	Yes	No
Configuration	Manual, DHCP	Automatic, DHCP
DNS name queries	Uses A records	Uses AAAA or A6 records
DNS reverse queries	Uses IN-ADDR.ARPA	Uses IP6.INT or IP6.ARPA
Minimum MTU	576 bytes	1280 bytes

Microsoft

Development Issues

- ✍ Tools
- ✍ API changes
- ✍ Porting issues
- ✍ Reading list

Microsoft

The Checkv4 Utility

- ✍ Parses your code for IPv4 specific usages

- ✍ Finds problem areas and suggests changes

```
test.c(35) : gethostbyname : use  
getaddrinfo instead
```

```
test.c(48) : SOCKADDR_IN : use  
SOCKADDR_STORAGE instead, or use  
SOCKADDR_IN6 in addition for IPv6  
support
```

- ✍ Located:

- ✍ In IPv6 Technology Preview for Windows 2000

Microsoft

Core Sockets Functions

- ✍ Core APIs don't change

- ✍ Use IPv6 Family and Address Structures
- ✍ socket() Uses PF_INET6

- ✍ Functions that pass addresses

- ✍ bind()
- ✍ connect()
- ✍ sendmsg()
- ✍ sendto()

- ✍ Functions that return addresses

- ✍ accept()
- ✍ recvfrom()
- ✍ recvmsg()
- ✍ getpeername()
- ✍ getsockname()

Microsoft

Porting Issues

- ✂ Running on ANY system
- ✂ Address size
- ✂ New IPv6 APIs for IPv4/IPv6
- ✂ Ordering of API calls
- ✂ User interface issues
- ✂ Higher layer protocol changes

Microsoft

Specific things to look for

- ✂ Storing IP address in Dword or 4 bytes of an array.
- ✂ Use of explicit dotted decimal format in UI.
- ✂ Obsolete / New:
 - ✂ AF_INET – replaced by AF_INET6
 - ✂ SOCKADDR_IN – replaced by SOCKADDR_STORAGE
 - ✂ IPPROTO_IP – replaced by IPPROTO_IPV6
 - ✂ IP_MULTICAST_LOOP – replaced by SIO_MULTIPOINT_LOOPBACK
 - ✂ gethostbyname – replaced by getaddrinfo
 - ✂ gethostbyaddr – replaced by getnameinfo

Microsoft

Using getaddrinfo()

```
Client Side..
if (getaddrinfo(server_name, port, NULL, &ai) != 0) { /* Error Handling */ }

conn_socket = socket(ai->ai_family, ai->ai_socktype, 0);
    if (conn_socket < 0) { /* Error Handling */ }

if (connect(conn_socket, ai->ai_addr, ai->ai_addrlen) == SOCKET_ERROR)
{ /* Error Handling */ }

freeaddrinfo(ai);
```

```
Server Side..
hints.ai_family = AF_INET6;
hints.ai_socktype = SOCK_STREAM;
hints.ai_flags = AI_NUMERICHOST | AI_PASSIVE;

retval = getaddrinfo(interface, port, &hints, &ai);
if (retval != 0) { /* Error Handling */ }

listen_socket = socket(ai->ai_family, ai->ai_socktype, 0);
if (listen_socket == INVALID_SOCKET) { /* Error Handling */ }

if (bind(listen_socket, ai->ai_addr, ai->ai_addrlen) == SOCKET_ERROR)
{ /* Error Handling */ }

freeaddrinfo(ai);
```

Microsoft

Client apps

- ✍ Resolve names *before* opening socket

```
getaddrinfo(...)
```

```
s = socket(ai->ai_family,
    ai->ai_socktype, ai-
    >ai_protocol);
```

```
connect(...)
```

NOT

```
s = socket(AF_INET, SOCK_XXX, 0);
```

```
gethostbyname(...)
```

```
connect(...)
```

Microsoft

Server Apps

- ✦ Use two listening sockets, one for IPv4, one for IPv6
- ✦ Service should start as long as *either* socket can be opened
 - ✦ Could also use WSAEnumProtocols()

Microsoft

Porting Steps - Summary

- ✦ Determine stack availability
- ✦ Use IPv4/IPv6 protocol/address family
- ✦ Fix address structures
 - ✦ in6_addr
 - ✦ sockaddr_in6
 - ✦ sockaddr_storage to allocate storage
- ✦ Fix wildcard address use
 - ✦ in6addr_any, IN6ADDR_ANY_INIT
 - ✦ in6addr_loopback, IN6ADDR_LOOPBACK_INIT
- ✦ Use IPv6 socket options
 - ✦ IPPROTO_IPV6, options as needed
- ✦ Use getaddrinfo()
 - ✦ For Address Resolution

Microsoft

Demo

- ✦ IPv6 enabled client/server app both run on IPv4/IPv6 nodes
 - ✦ Listen to both IPv4/IPv6
 - ✦ Listen to IPv6 only
 - ✦ Listen to IPv4 only

Microsoft

Reading List

- ✦ RFC 2373 Address Architecture
- ✦ RFC 2460 IPv6 Specification
- ✦ RFC 2461 Neighbor Discovery
- ✦ RFC 2462 Stateless Autoconfiguration
- ✦ RFC 2463 ICMPv6
- ✦ RFC 2507 Header compression
- ✦ RFC 2732 Literal IPv6 Addresses in URL's
- ✦ RFC 1981 Path MTU for IPv6
- ✦ RFC 2526 Subnet Anycast
- ✦ RFC 2675 IP Jumbograms
- ✦ RFC 2472 IPv6 over PPP
- ✦ RFC 2473 Generic IPv6 packet tunneling
- ✦ RFC 1886 DNS Extensions
- ✦ draft-ietf-ipngwg-dns-lookups-07.txt

Microsoft

Getting Started

- ✎ Download the technology preview from MSDN
- ✎ Run the code scrubber tool checkv4
- ✎ Move from gethostbyname to getaddrinfo

Microsoft

Additional Info

Statement of Objectives

<http://www.microsoft.com/windows2000/library/howitworks/communications/networkbasics/IPv6.asp>

Technology Introduction

<http://www.microsoft.com/windows2000/library/howitworks/communications/nameadrmgmt/introipv6.asp>

Windows 2000 Technology Preview Download

<http://msdn.microsoft.com/downloads/sdks/platform/tpipv6.asp>

Microsoft

