IPv6 on z/OS

Mike Fitzpatrick – mfitz@us.ibm.com

IBM Raleigh, NC

Thursday, August 9th, 9:30am

Session: 11340

Trademarks, notices, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries or both:

- Advanced Peer-to-Peer Networking®
- AIX®
- alphaWorks®
- AnyNet®
- AS/400®
- BladeCenter®
- Candle®
- CICS®
- DataPower®
- DB2 Connect
- DB2®
- DRDA®
- e-business on demand®
- e-business (logo)
- e business(logo)®
- ESCON®
- FICON®

- GDDM®
- GDPS®
- Geographically Dispersed Parallel Sysplex
- HiperSockets
- HPR Channel Connectivity
- HyperSwap
- i5/OS (logo)
- i5/OS®
- IBM eServer
- IBM (logo)®
- IBM®
- IBM zEnterprise™ System
- IMS
- InfiniBand ®
- IP PrintWay
- IPDS
- iSeries
- LANDP®

- Language Environment®
- MQSeries®
- MVS
- NetView®
- OMEGAMON®
- Open Power
- OpenPower
- Operating System/2®
- Operating System/400®
- OS/2®
- OS/390®
- OS/400®
- Parallel Sysplex®
- POWER®
- POWER7®
- PowerVM
- PR/SM
- pSeries®
- RACF®

- Rational Suite®
- Rational®
- Redbooks
- Redbooks (logo)
- Sysplex Timer®
- System i5
- System p5
- System x®
- System z®
- System z9®
- System z10
- Tivoli (logo)®
- Tivoli®
- VTAM®
- WebSphere®
- xSeries®
- z9®
- z10 BC
- z10 EC

- zEnterprise
- zSeries®
- z/Architecture
- z/OS®
- z/VM®
- z/VSE

* All other products may be trademarks or registered trademarks of their respective companies.

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries or both:
- Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
- Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license there from.
- Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
- InfiniBand is a trademark and service mark of the InfiniBand Trade Association.
- Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
- UNIX is a registered trademark of The Open Group in the United States and other countries.
- Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
- ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.
- IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency, which is now part of the Office of Government Commerce.

Notes:
- Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.
- IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.
- All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.
- This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.
- All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.
- Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.
- Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

Refer to www.ibm.com/legal/us for further legal information.

Other IPv6 Sessions at this SHARE

11156: IPv6 Basics	Wednesday, August 8th: 8:00AM-9:00AM
11155: IPv6 Addressing	Wednesday, August 8th: 9:30AM-10:30AM
11157: IPv6 Design	Wednesday, August 8th: 1:30PM-2:30PM
11158: IPv6 Implementation	Wednesday, August 8th: 3:00PM-4:00PM
11347: IPv6 User Experience: Planning meets Deployment	Thursday, August 9th: 11:00AM-12:00PM
11351: IPv6 Configuration on z: Hands-on Lab	Thursday, August 9th: 1:30PM-2:30PM
11364: IPv6 Deep Dive	Thursday, August 9th: 3:00PM-4:00PM
11363: How to Avoid Common IPv6 Mistakes in z/OS	Thursday, August 9th: 4:30PM-5:45PM

Agenda

- ❑ **What is IPv6?**

- ❑ **Coexistence and Migration**

- ❑ **Enabling IPv6 on z/OS**

- ❑ **What functions are IPv6-enabled?**

- ❑ **Planning how to get there**

Disclaimer: All statements regarding IBM future direction or intent, including current product plans, are subject to change or withdrawal without notice and represent goals and objectives only. All information is provided for informational purposes only, on an "as is" basis, without warranty of any kind.

What is IPv6?

So - what is IPv6?

- **IPv6 is an evolution of the current version of IP, which is known as IPv4**
  - Work on new IETF standard started in early 90's
  - Not backward compatible, but migration techniques defined
- **Today's IPv4 has 32 bit addresses**
  - Theoretical limit is around 4 billion addresses
  - Due to IPv4 address assignment structure and policies, the practical limit is less than 1 billion useable global addresses

> IPv4 Address:
> 9.67.122.66

- **IPv6 provides almost unlimited number of addresses**
  - IPv6 addresses are 128 bits
  - No practical limit on global addressability

> IPv6 Address:
> 2001:0DB8:4545:2::09FF:FEF7:62DC

  - Enough address space to meet all imaginable needs for a while
  - More addresses *cannot* be retrofitted into IPv4
- **Other improvements important, but to some extent secondary:**
  - Facilities for automatic configuration
  - Improved support for site renumbering
  - End to end IP security
  - Mobility with route optimization (important for wireless)
  - Miscellaneous improvements aimed at improving router performance

Important IPv6 technical features

- **IPv6 header and extensions header**
  - Streamlined IPv6 header
    - Fixed length to speed up forwarding processing in routers
  - Optional extension headers for fragmentation, security, etc.
- **Routers are no longer allowed (able) to fragment forwarded datagrams**
  - Path MTU discovery is always used
- **Expanded size of IP address space**
  - Address space increased to 128 bits
    - Provides 340,282,366,920,938,463,463,374,607,431,768,211,456 addresses
    - Enough for many(!) addresses per person on the planet
  - A 64-bit subnet prefix identifies the link
  - Followed by a 64-bit Interface Identifier (IID)
- **IID may be derived from IEEE identifier (MAC address)**
  - Only leftmost 64 bits available for routing and "network addressing"
  - The rightmost 64-bits identify the host on the target link

Network Prefix (n bits)	Subnet ID (64-n bits)	Interface Identifier (IID) (64 bits)
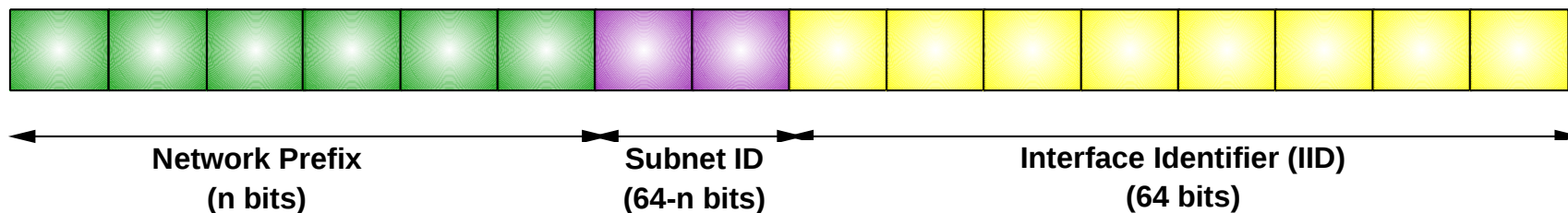
IPv6 address textual representation

- Addresses are represented as 8 segments of 4 hex digits (16 bits), separated by colons

  2001:0DB8:0:0:240:2BFF:FE3D:71AD

- Two colons in a row can be used to denote one or more sets of zeroes, usually used between the prefix and the interface ID
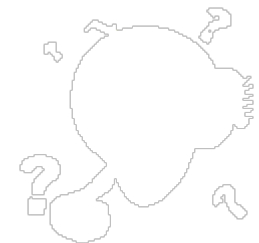
  2001:0DB8::240:2BFF:FE3D:71AD

- The prefix length can be indicated after a slash at the end

  2001:0DB8::240:2BFF:FE3D:71AD/64

- A prefix alone is represented as if the interface ID bits are all zero

  2001:0DB8::/64

- Obviously, this syntax may be a bit difficult for humans…..
  - Use of DNS/hostnames is no longer an optional convenience

Common IPv6 addresses and prefixes

- **::/128**
  INADDR6_ANY (the unspecified address)

- **::1/128**
  IPv6 loopback address

- **FF00::/8**
  Multicast addresses

8 bits	4 bits	4 bits	112 bits
11111111	Flags	scope	Group ID

- **FE80::/10**
  Link-local addresses

10 bits	54 bits	64 bits
1111111010	0	Interface ID

- **FC00::/7**
  Unique local addresses

7 bits	121 bits
1111110	Local address

- **::FFFF/96**
  IPv4-Mapped IPv6 Address

80 bits	16 bits	32 bits
0	FFFF	IPv4 address

- **2000::/3**
  Current globally unique IPv6 address space (may change in the future

3 bits	45 bits	16 bits	64 bits
Network prefix		Subnet	Interface ID
001	Network id		

IPv6 scoped unicast addressing

- Concept of scoped unicast addresses is part of architecture
- Link-local addresses for use on a single link
    - Primarily used for bootstrapping and infrastructure protocols such as Neighbor Discovery
    - Address = well-known link-local prefix plus node-generated IID
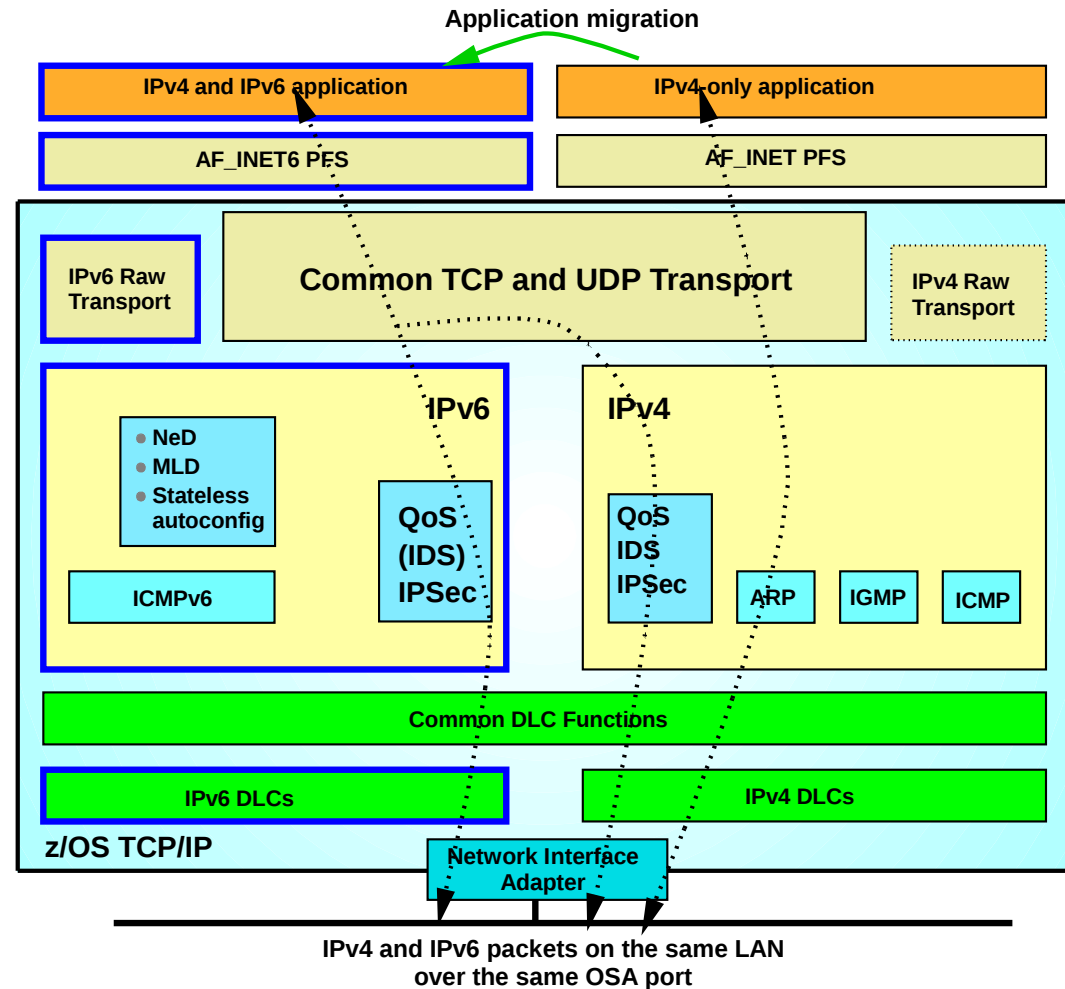- Unique Local IPv6 Unicast addresses for use within a site
    - Like net 10 (not routable in the Internet backbone)
    - Site-local addresses
        - Part of early IPv6 standards - has been deprecated by the IETF
- Global address prefixes are provided by ISPs

This is **legal** because the fe80::12 address is only required to be unique per-link

fe80::1234:5678:9999:aaaa

fe80::1111

fe80::12

fe80::12

link 1

link 2

fe80::1234:5678:9abc:def0

fe80::1234

fe80::1234

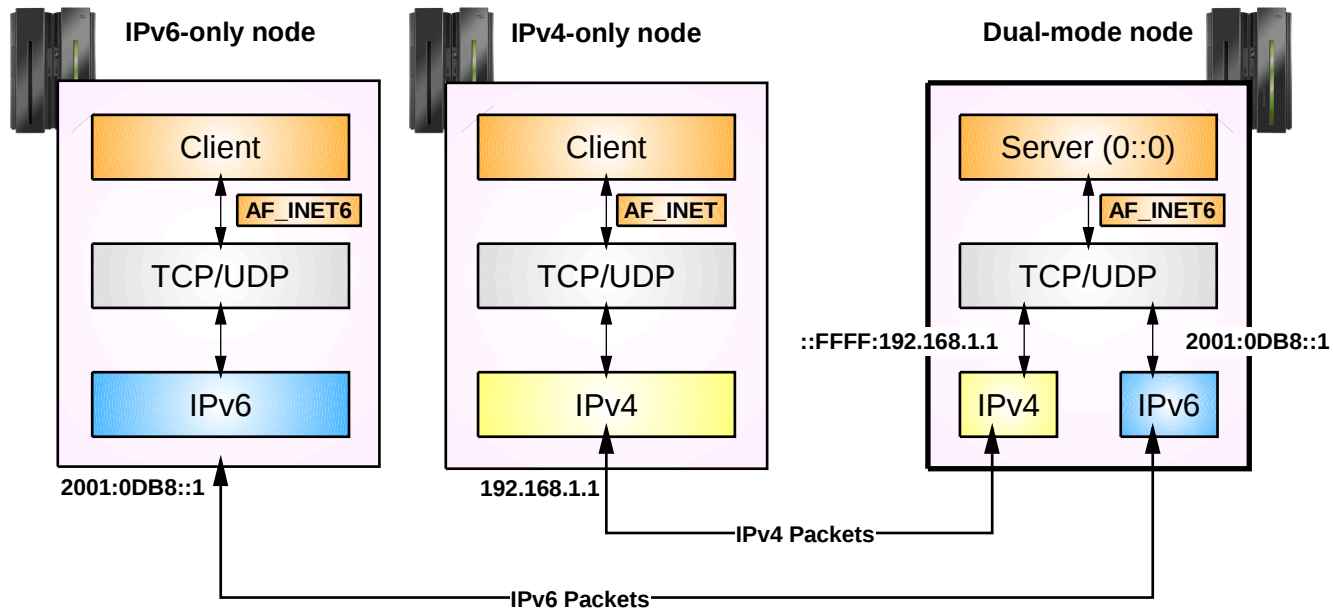this is **illegal** because the fe80::1234 address is not unique on the link
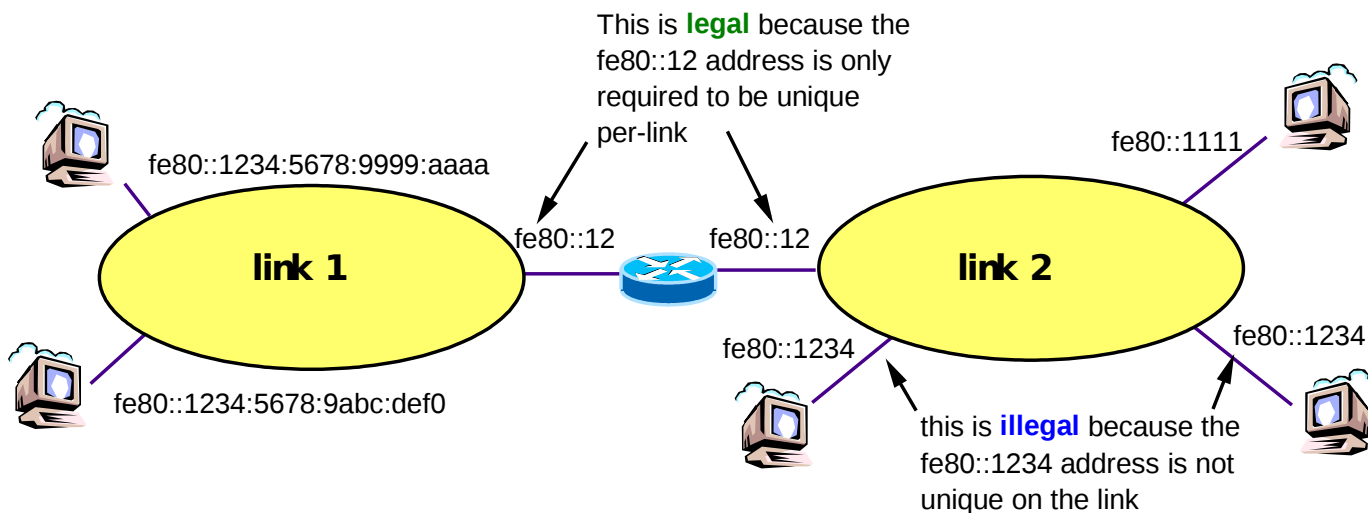
Coexistence and Migration

z/OS TCP/IP is a dual-mode TCP/IP stack

- A dual-mode (or dual-stack) TCP/IP implementation supports both IPv4 and IPv6 interfaces – and both old AF_INET and new AF_INET6 applications.

- The dual-mode TCP/IP implementation is a key technology for IPv4 and IPv6 coexistence in an internet.

- For AF_INET6 applications, the common TCP or UDP transport layer determines per communication partner if the partner is an IPv4 or an IPv6 partner - and chooses the IPv4 or IPv6 networking layer component based on that.

Application migration

IPv4 and IPv6 application

IPv4-only application

AF_INET6 PFS

AF_INET PFS

IPv6 Raw Transport

Common TCP and UDP Transport

IPv4 Raw Transport

IPv6

IPv4

- NeD
- MLD
- Stateless autoconfig

QoS (IDS) IPSec

ICMPv6

QoS IDS IPSec

ARP

IGMP

ICMP

Common DLC Functions

IPv6 DLCs

IPv4 DLCs

z/OS TCP/IP

Network Interface Adapter

IPv4 and IPv6 packets on the same LAN over the same OSA port

- Raw applications make the determination themselves when they choose IPv4 or IPv6 raw transport.

IPv6-enabled application on a dual mode stack

IPv6-only node — Client — AF_INET6 — TCP/UDP — IPv6 — 2001:0DB8::1

IPv4-only node — Client — AF_INET — TCP/UDP — IPv4 — 192.168.1.1

Dual-mode node — Server (0::0) — AF_INET6 — TCP/UDP — ::FFFF:192.168.1.1 — IPv4 — IPv6 — 2001:0DB8::1

IPv4 Packets

IPv6 Packets

- An IPv6-enabled application can communicate with both IPv4 and IPv6 peers
  - A single socket can be used to send or receive traffic from either IPv4 or IPv6 partners
  - IPv4 packets to the IPv4 partner and IPv6 packets to the IPv6 partner
  - No changes need to be made to the partner application
- An IPv6-enabled application uses AF_INET6 sockets for both IPv4 and IPv6 partners
  - An IPv4 address is mapped to IPv6 addresses by the Transport Layer in the TCP/IP stack
  - Uses the special address format which identifies the IPv6 address as an IPv4-mapped IPv6 address
  - For example, 192.168.1.1 would be represented as ::FFFF:192.168.1.1

IPv4-only application on a dual-mode stack
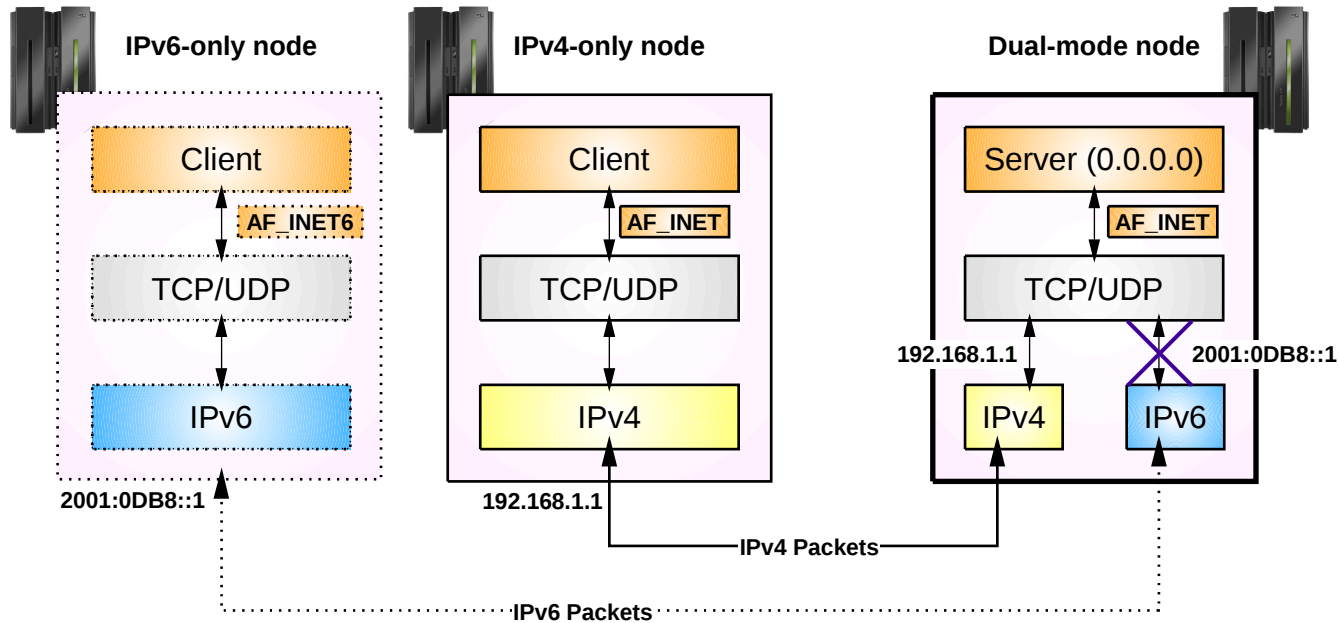
- An IPv4 application running on a dual-mode stack can communicate with an IPv4 partner
  - The source and destination addresses will be native IPv4 addresses
  - The packet which is sent will be an IPv4 packet
- If partner is IPv6 running on an IPv6 only stack, then communication fails
  - If partner was on dual-mode stack, then it would fit in previous page discussion
  - The partner only has a native IPv6 address, not an IPv4-mapped IPv6 address
  - The native IPv6 address for the partner cannot be converted into a form the AF_INET application will understand

Accessing IPv4-only applications through an IPv6 application layer gateway (ALG)
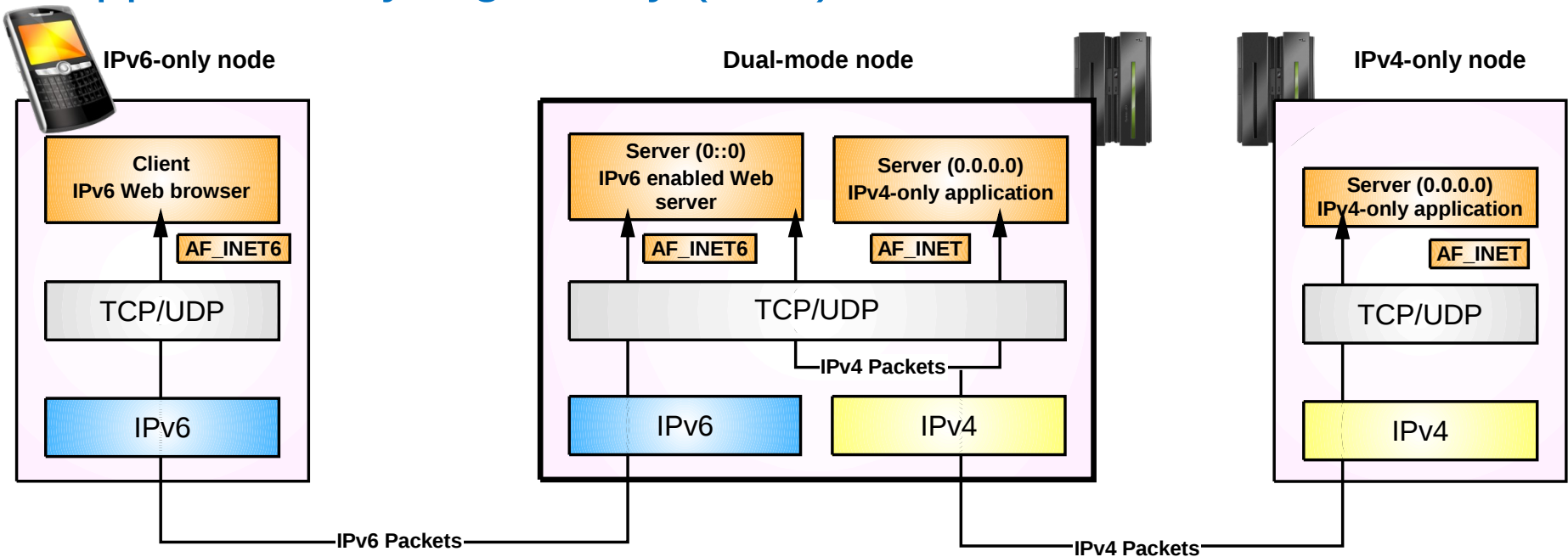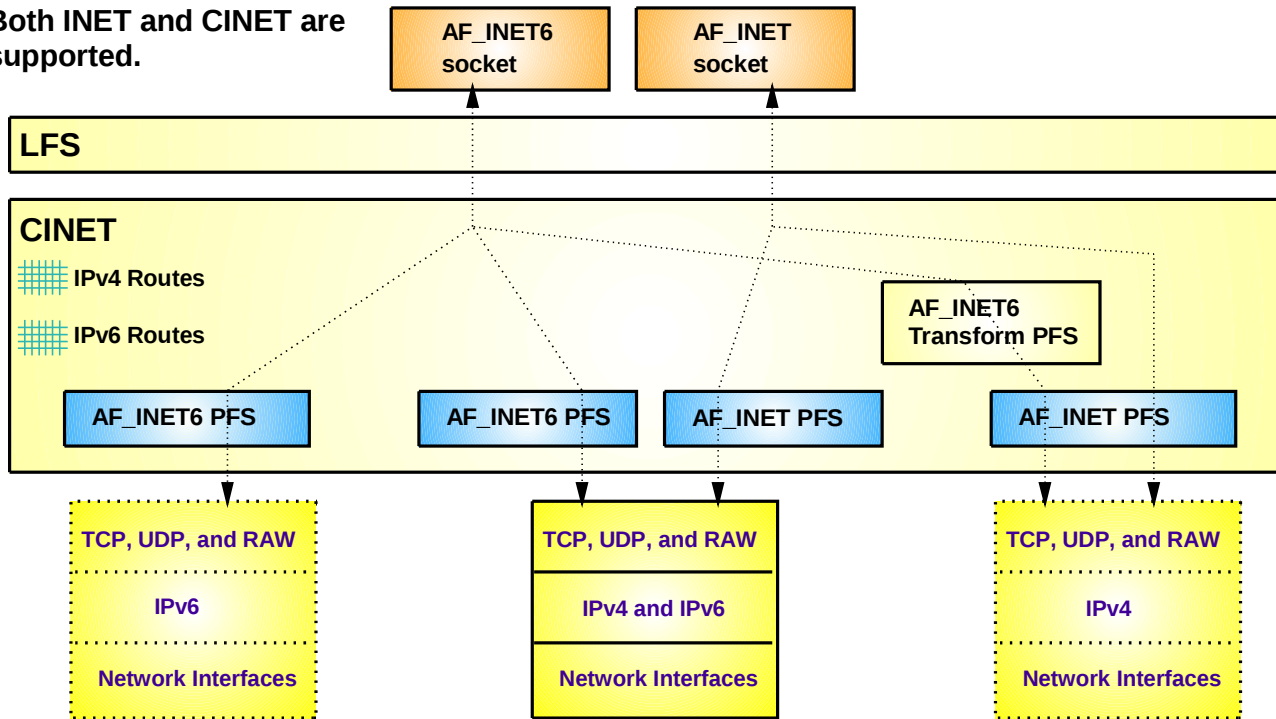

- An IPv6-only client can access IPv4-only servers via an IPv6 "proxy"

  - The IPv6 proxy communicates with the IPv6-only client using IPv6, and accesses the IPv4-only server using IPv4

  - The IPv4-only server may be on the same TCP/IP stack as the IPv6 proxy, or may reside on a different stack

  - The use of a backend IPv4-only server is, in most cases, completely transparent to the IPv6 client

Enabling IPv6 on z/OS

Enabling IPv6 support on z/OS

IPv6 is enabled at an LPAR level via an option in BPXPRMxx to enable AF_INET6 support.
Both INET and CINET are supported.

When IPv6 is enabled, a z/OS TCP/IP stack will always have an IPv6 Loopback interface. You can define real IPv6 interfaces in addition to the loopback interface.

AF_INET6 socket	AF_INET socket

LFS

CINET

▦ **IPv4 Routes**

▦ **IPv6 Routes**

AF_INET6 Transform PFS

AF_INET6 PFS	AF_INET6 PFS	AF_INET PFS	AF_INET PFS

TCP, UDP, and RAW	TCP, UDP, and RAW	TCP, UDP, and RAW
IPv6	IPv4 and IPv6	IPv4
Network Interfaces	Network Interfaces	Network Interfaces

IPv6-only TCP/IP Stack

This will not be the case on z/OS for the foreseeable future! An AF_INET6 stack is required to also support AF_INET!

Dual Mode TCP/IP Stack

A z/OS TCP/IP stack will always come up as dual-mode if AF_INET6 is enabled in BPXPRMxx

IPv4-only TCP/IP Stack

(such as an OEM TCP/IP stack)

► Existing AF_INET sockets programs will continue to work as they always did - no difference in behavior or support.

► AF_INET6 enabled sockets programs will be able to communicate with IPv4 partners (just as before they were changed to support IPv6), but in addition to that they will also be able to communicate with IPv6 partners.

Netstat output format LONG or SHORT

- When IPv6 is enabled, most netstat reports will look different because of the potential for long IPv6 addresses.
  - Without IPv6 enabled, netstat uses what is known as a SHORT report format
    - It is possible to have both local and remote IPv4 address in one 80-character line
    - You can override the SHORT format by coding IPCONFIG FORMAT LONG
  - With IPv6 enabled, netstat uses a LONG report format
    - Each IPv6 address may potentially be up to 45 characters long, which makes it impossible to have both local and remote IPv6 addresses in a single 80-character line
- Make sure you update any netstat screen-scraping REXX programs you might have developed in the past!

```
MVS TCP/IP NETSTAT CS V1R12        TCPIP Name: TCPCS          12:50:02
User Id  Conn      State
-------  ----      -----
MYINETD1 00000025 Listen
  Local Socket:   9.42.104.161..23
  Foreign Socket: 0.0.0.0..0
TN3270A  00000045 Listen
  Local Socket:   ::..23
  Foreign Socket: ::..0
TN3270A  00001B5E Establsh
  Local Socket:   ::ffff:9.42.105.45..23
  Foreign Socket: ::ffff:9.50.52.109..58646
  Application Data:  EZBTNSRV TCPABC81 TSO10001 ET B
```

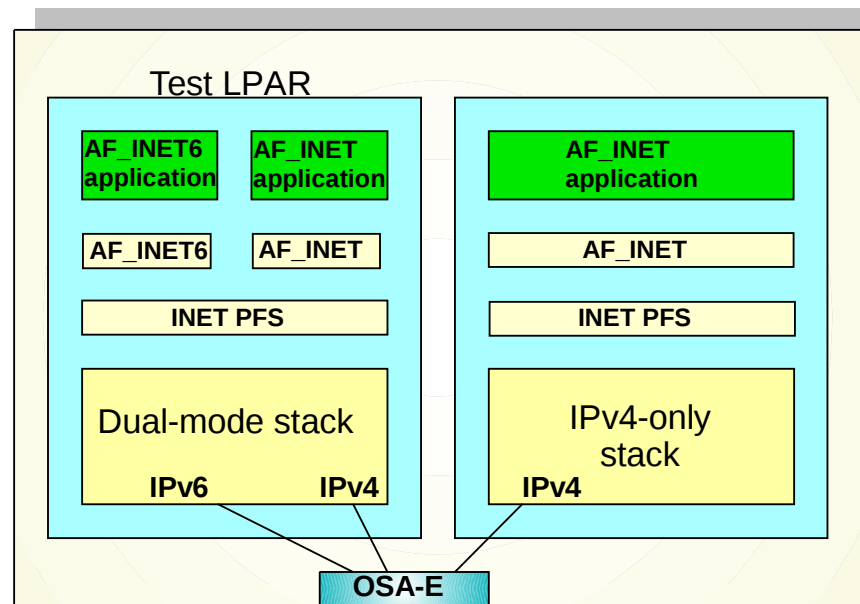# Accessing z/OS over IPv6 from a remote site

Remember:

You can enable IPv6 today on z/OS without impact to your existing IPv4 users.
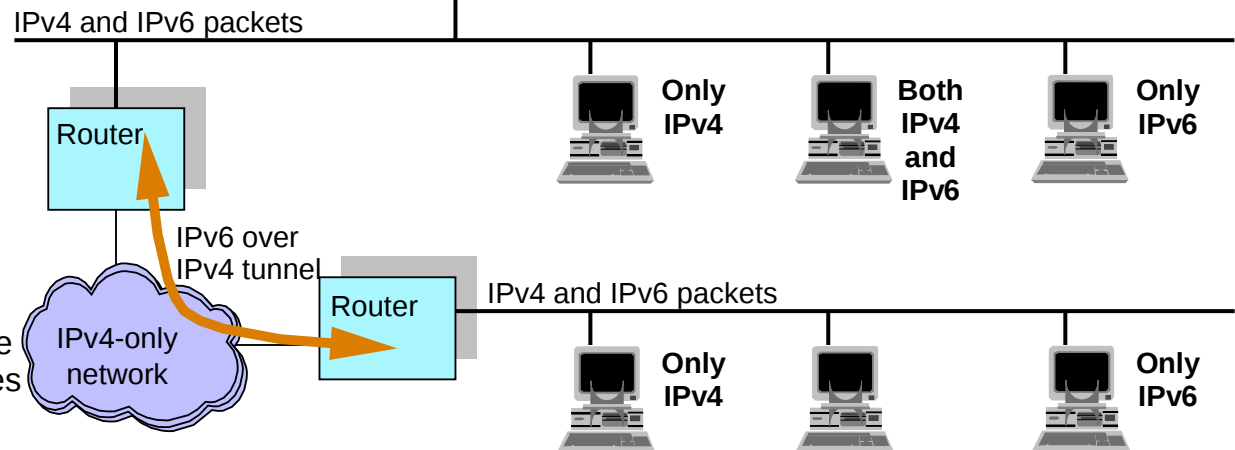
Do it on your test system – initially without defining any IPv6 interfaces.

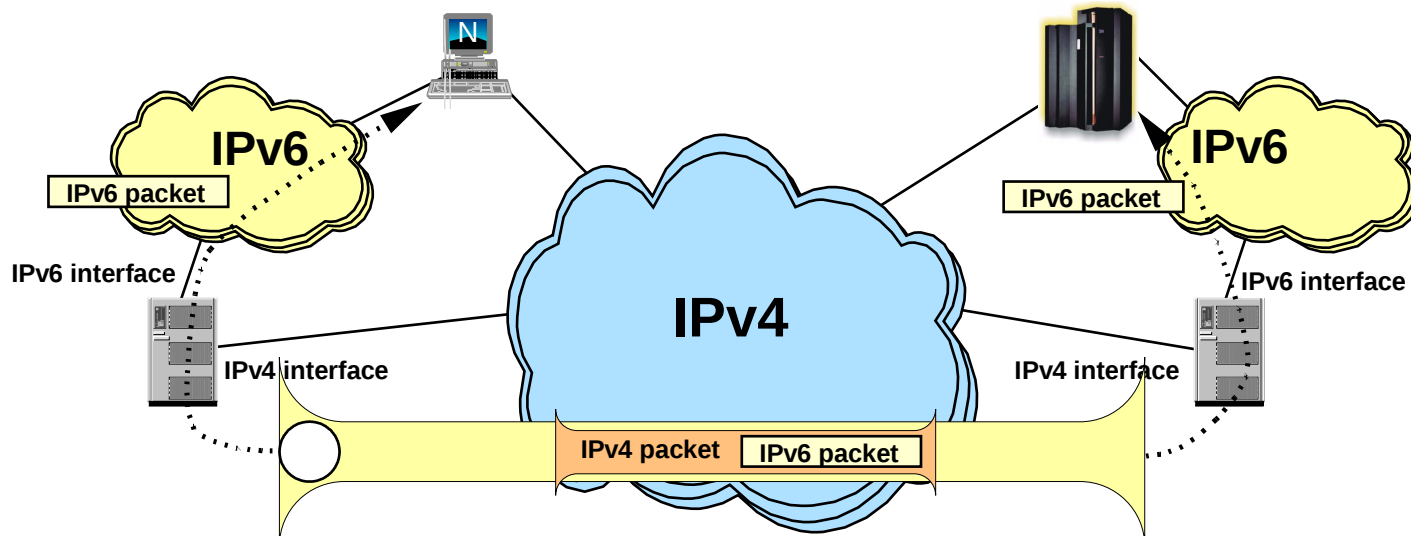All IPv4 communication continues to work as before.

Use IPv6 over IPv4 tunneling when native IPv6 connectivity does not exist

Test LPAR

AF_INET6 application	AF_INET application		AF_INET application
AF_INET6	AF_INET		AF_INET
INET PFS			INET PFS
Dual-mode stack			IPv4-only stack
IPv6	**IPv4**		**IPv4**

OSA-E

IPv4 and IPv6 packets

z/OS CS does not support being a tunnel endpoint, although it can route traffic through an intermediate tunnel

Router

IPv6 over IPv4 tunnel

IPv4-only network

Router

IPv4 and IPv6 packets

Only IPv4

Both IPv4 and IPv6

Only IPv6

Only IPv4

Only IPv6

IPv6 paths are preferred over IPv4

- IPv6 connectivity is preferred over IPv4
  - In many cases, only if one of the nodes does not support IPv6 will IPv4 be used
  - Can lead to undesirable connection paths in the network
    - Data may be tunneled over the IPv4 network even when a native IPv4 path exist
- May lead to longer connection establishment to an AF_INET application on a dual-stack node
  - IPv6 addresses will be tried before attempting to connect via IPv4
  - A "well behaved" client will cycle through all addresses returned and may try the IPv4 address
    - But this takes time and network resources
    - And not all clients are "well behaved"

Use of distinct IPv4 and IPv6 host names
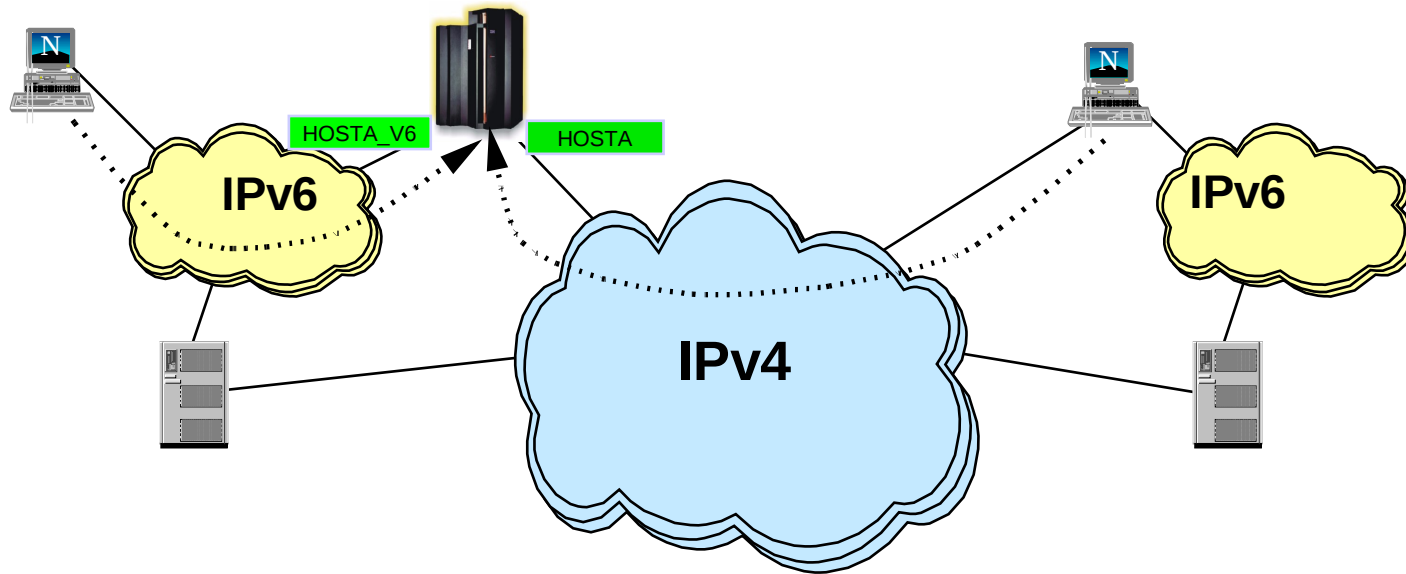
To avoid undesirable tunneling, configure two host names in DNS
- Continue to use the existing host name for IPv4 connectivity
- Create a new host name to be used for IPv6 connectivity
- Optionally, a third host name which may be used for both IPv4 and IPv6 can be configured

Client chooses type of connection based on host name
- Using the existing host name results in IPv4 connectivity
- Using the new host name results in IPv6 connectivity

Note: Use of distinct host names is only necessary during the initial transition phases when native IPv6 connectivity does not exist

What functions are IPv6-enabled?

Defining IPv6 interfaces

- IPv6 interfaces are defined using an INTERFACE statement in the TCP/IP profile
    - Combines the definitions of DEVICE, LINK and HOME into one statement
    - In order for one physical device to support both IPv4 and IPv6 traffic, DEVICE, LINK and HOME statements (or INTERFACE statement) have to be specified in the profile to define the IPv4 side and an INTERFACE statement must be specified to define the IPv6 side
- A single IPv6 interface may have one or more IPv6 addresses at any given time
    - There will always be a link-local address, which is automatically assigned during interface activation
    - There may be 0-n local-unicast and/or global IPv6 addresses as well
- For physical interfaces, IP addresses (except for the link-local address) may be manually configured or may be autoconfigured

```
INTERFACE OSAQDIO15 DEFINE IPAQENET6 PORTNAME OSAQDIO1

INTERFACE OSAQDIO25 DEFINE IPAQENET6 PORTNAME OSAQDIO2
IPADDR  FC00::9:67:115:5
        2001:0DB8::9:67:115:5
```

Netstat DEVLINKS/-d

```
MVS TCP/IP onetstat CS V1R12      TCPIP Name: TCPCS              12:55:20
DevName: OSAQDIO4          DevType: MPCIPA
  DevStatus: Ready
  LnkName: OSAQDIOLINK        LnkType: IPAQENET    LnkStatus: Ready
    NetNum:  0   QueSize: 0   Speed: 0000000100
    IpBroadcastCapability: No
    CfgRouter: Non                   ActRouter: Non
    ArpOffload: Yes                  ArpOffloadInfo: Yes
    ActMtu:  1492
    VLANid:  1260                    VLANpriority: Enabled
    ReadStorage: GLOBAL (8064K)      InbPerf: Balanced
    ChecksumOffload: Yes
  BSD Routing Parameters:
    MTU Size: 00000           Metric: 00
    DestAddr: 0.0.0.0         SubnetMask: 255.255.255.192
  Multicast Specific:
    Multicast Capability: Yes
    Group              RefCnt
    -----              ------
    224.0.0.1          0000000001
  Link Statistics:
    BytesIn                      = 11476
    Inbound Packets              = 10
    Inbound Packets In Error     = 0
    Inbound Packets Discarded    = 0
    Inbound Packets With No Protocol = 0
    BytesOut                     = 6707
    Outbound Packets             = 10
    Outbound Packets In Error    = 0
    Outbound Packets Discarded   = 0
```

Netstat DEVLINKS/-d *(continued)*

```
IntfName: OSAQDIO46        IntfType: IPAQENET6 IntfStatus: Ready
  NetNum:  0   QueSize: 0   Speed: 0000000100
  MacAddress: 000629DC21BC
  SrcVipaIntf: VIPAV6
  DupAddrDet: 1
  CfgRouter: Pri                    ActRouter: Pri
  RtrHopLimit: 5
  CfgMtu:  4096                     ActMtu:    1492
  VLANid:  1261                     VLANpriority: Enabled
  IntfID:   0000:0000:0000:0001
  ReadStorage: GLOBAL (8064K)       InbPerf: Balanced
Packet Trace Setting:
  Protocol: *               TrRecCnt: 00000000  PckLength: FULL
  SrcPort:  *               DestPort: *
  IpAddr/PrefixLen: 9::44/128
Multicast Specific:
  Multicast Capability: Yes
  RefCnt        Group
  ------        -----
  0000000001  ff02::1:ff15:5
  0000000001  ff02::1:ff00:2
Interface Statistics:
  BytesIn                      = 12655
  Inbound Packets              = 12
  Inbound Packets In Error     = 0
  Inbound Packets Discarded    = 0
  Inbound Packets With No Protocol = 0
  BytesOut                     = 4590
  Outbound Packets             = 11
  Outbound Packets In Error    = 0
  Outbound Packets Discarded   = 0
```

IPv6 VIPA and SOURCEVIPA

- Static VIPAs are defined on a VIRTUAL6 interface
    - Each VIRTUAL6 interface must be manually configured with one or more IPv6 addresses

- Use the SOURCEVIPAINTerface parameter to associate a physical interface to a specific VIRTUAL6 interface
    - No ordering considerations like DEVICE/LINK/HOME for IPv4
    - The TCP/IP stack will choose the "best" address as the source IP address using the Default Address Selection algorithms defined by the IETF

    - More than one physical interface can point to the same VIRTUAL6 interface
- IPCONFIG6 SOURCEVIPA definition makes the SOURCEVIPA function available for all

    IPv6 interfaces configured with SOURCEVIPAINTerface.

```
IPCONFIG6 SOURCEVIPA

INTERFACE VIPAV61 DEFINE VIRTUAL6
IPADDR  FC00::9:67:115:5  2001:0DB8::9:67:115:5

INTERFACE VIPAV62 DEFINE VIRTUAL6
IPADDR  FC00::9:67:115:6  2001:0DB8::9:67:115:6

INTERFACE OSAQDIO16 DEFINE IPAQENET6 PORTNAME OSAQDIO1
SOURCEVIPAINT VIPAV61

INTERFACE OSAQDIO26 DEFINE IPAQENET6 PORTNAME OSAQDIO2
SOURCEVIPAINT VIPAV62

INTERFACE OSAQDIO36 DEFINE IPAQENET6 PORTNAME OSAQDIO3
```

Sockets-related AF_INET6 enablement

- Currently supported
  - LE provided C/C++ API
  - Unix System Services callable BPX API
  - REXX
  - CICS
  - EZASOKET callable
  - EZASMI assembler

- Not supported
  - TCP/IP provided C API
  - Pascal

z/OS Communications Server Applications IPv6-enabled

- IPv6-enabled applications in z/OS
  - inetd
  - ftp and ftpd
  - telnetd
  - USS rshd and rexecd servers
  - USS rexec client
  - ping
  - tracert
  - netstat
  - tftpd (trivial file transfer server)
  - syslogd
  - dcas (digital certificate access server)
  - sntpd (simple network time protocol server)
  - sendmail
  - MVS rshd/rexecd server
  - TSO rsh/rexec clients
    - Updated version that can be used in all z/OS environments (batch, TSO, REXX, etc.)
  - New UNIX rsh client that is IPv6-enabled from start
  - CICS Listener (including CICS socket APIs)

FTP

- FTP server
  - To enable IPv6 support in the FTP server, activate IPv6 stack support
    - No new configuration commands are provided or needed to enable IPv6 support
  - User Exit routines
    - Update server exit routines for IPv6 addressing
  - Trace and Extended Trace
    - Update DUMP IPADDR() and DEBUG IPADDR() as needed
  - NETRC data set
    - Update with IPv6 addresses as needed
  - SMF recording
    - Update SMF statements in client and server FTP.DATA commands
- FTP client
  - For the client, you may specify the host as an IPv4 address, a hostname, an IPv4 mapped IPv6 address, or as an IPv6 address
  - Examples:

        ftp fc00:197:11:105::1
        ftp 9.67.21.33 and ftp ::ffff:9.67.21.33 are equivalent
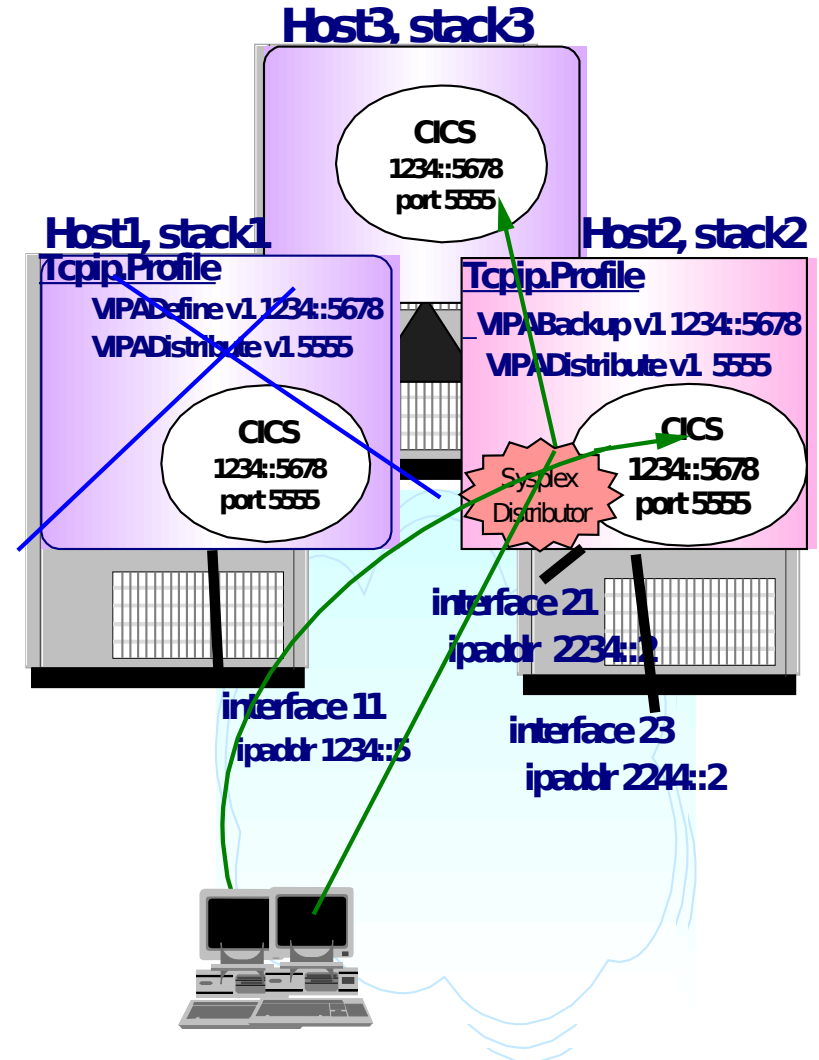   *ftp linuxipv6.tcp.raleigh.ibm.com*

TN3270

- To enable IPv6 support in the TN3270 server, activate IPv6 stack support
  - No new configuration commands are provided or needed to enable IPv6 support

- IPv6-enable the TN3270 server
  - Support clients with IPv6 addresses
  - Support IPv6 addresses in USS messages, displays, command responses, etc.
  - Support IPv6 addresses as client identifiers for all mapping statements in TN3270 server configuration that allows an IP address client identifier
  - Includes SSL/TLS support
  - Changes made to VTAM to support TN3270 visibility when clients are IPv6 clients
    - IPv6 addresses are passed to VTAM
    - VTAM displays that include IP addresses are enhanced to accommodate IPv6 addresses

Enterprise Extender

- Enterprise Extender has support for IPv6 in z/OS
    - Allows Enterprise Extender to exploit an IPv6-enabled network
    - Architectural changes needed since HPR passes IP addresses in protocol data and is supported on multiple platforms
    - Changes to VTAM exits to pass IPv6 addresses, hostnames, and port numbers:
        - SME (Session Management exit)
        - Login exit

- IPv6 support requires use of the HOSTNAME keyword (start option, GROUP, path definition)
    - Existing IPADDR keywords (start option, path definition in SMN) are IPv4-only

- EE Connection networks are IPv4-only or IPv6-only
    - Nodes supporting both IPv4 and IPv6 must define an IPv4 VRN and an IPv6 VRN
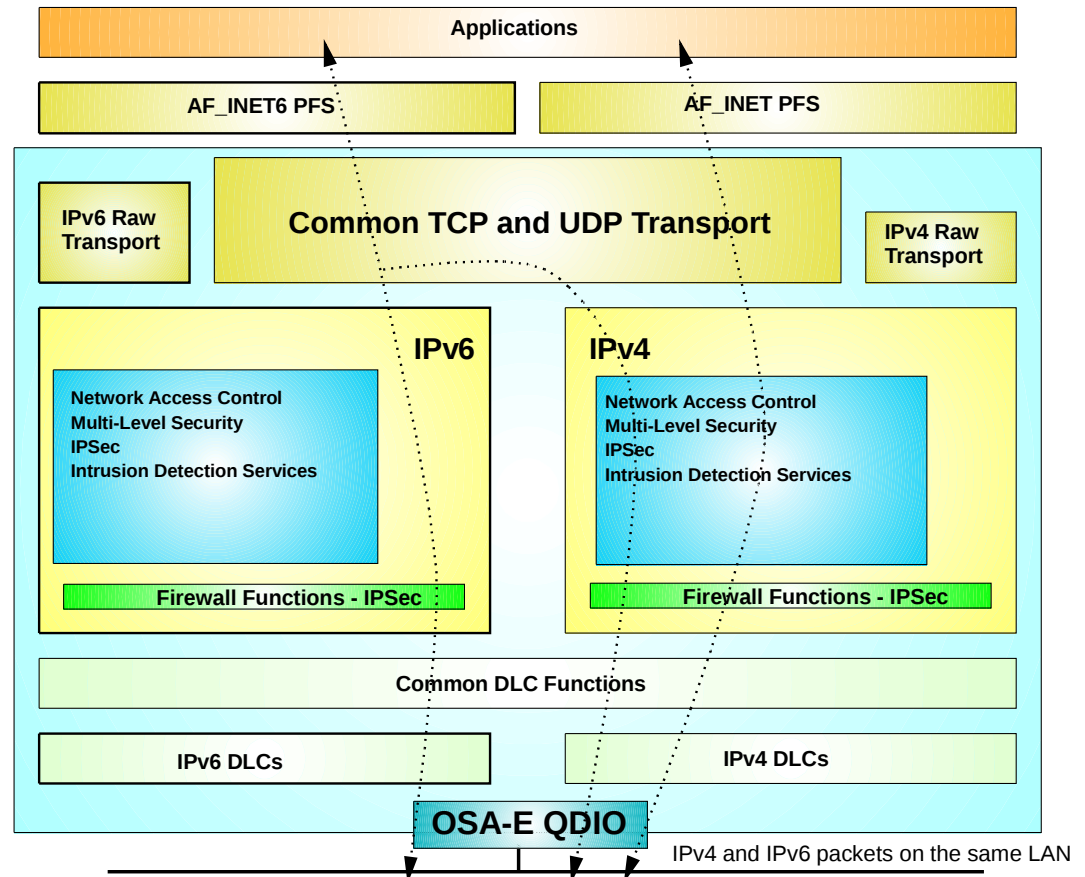
Sysplex functions that support IPv6

- Almost all Sysplex functions support IPv6
  - Dynamic VIPA (DVIPA)
  - Dynamic VIPA Takeover
  - Sysplex Distributor
  - Sysplex Sockets
  - TCPSTACKSOURCEVIPA
  - Sysplexports
  - Fast Connection Reset after System Failure
  - Enhance Workload Distribution (Application Server Affinity)
  - Dynamically Assign Sysplex Ports
  - Activation of DVIPAs through VIPABACKUP
  - DYNAMICXCF and SOURCEVIPAINT
  - Sysplex Distributor load balancing algorithms
  - Sysplex Distributor Policy
  - z/OS Load Balancing Advisor

Host3, stack3

CICS
1234::5678
port 5555

Host1, stack1

Tcpip.Profile
VIPADefine v1 1234::5678
VIPADistribute v1 5555

CICS
1234::5678
port 5555

Host2, stack2

Tcpip.Profile
VIPABackup v1 1234::5678
VIPADistribute v1 5555

Sysplex Distributor

CICS
1234::5678
port 5555

interface 21
ipaddr 2234::2

interface 11
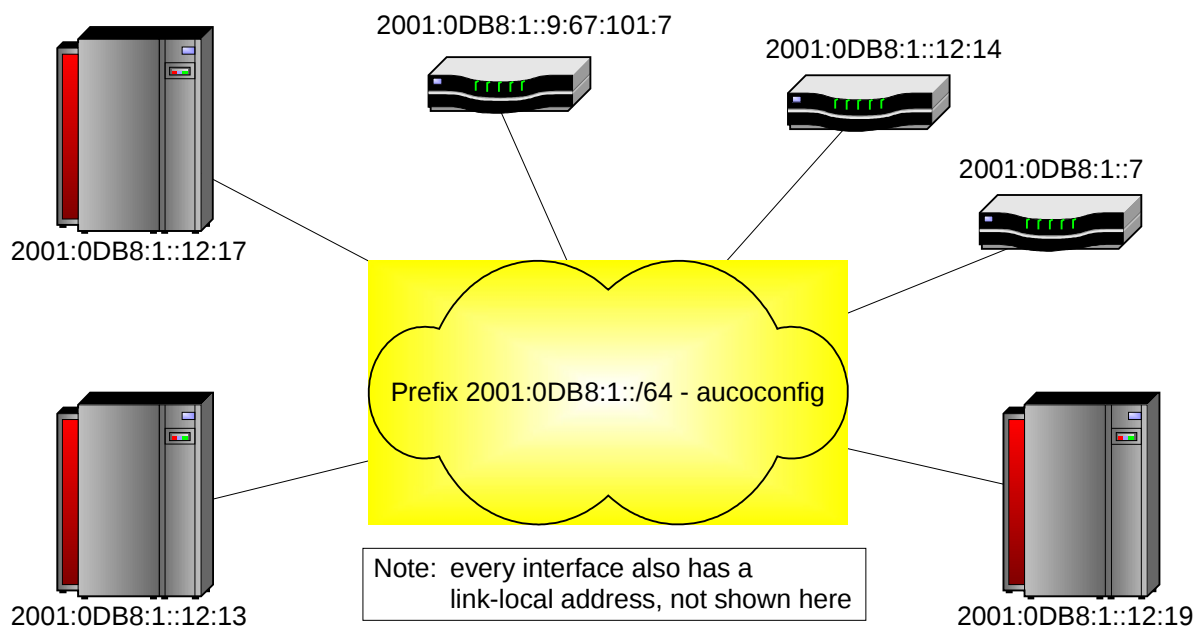ipaddr 1234::5

interface 23
ipaddr 2244::2

Securing your IPv6 network

- z/OS provides key security features for the IPv6 transport
  - Includes support for both Network Access Control and Multi-Level Security
  - Application Transparent TLS (ATTLS) for both IPv4 and IPv6
  - IPSec and Firewall filters support for both IPv4 and IPv6
  - IKEv2 support for both IPv4 and IPv6 (shipped in V1R12)
  - IPv6 IPSec currently does not support Sysplex Wide Security Associations (SWSA)
  - Intrusion Detection Services (IDS) support for IPv6 (shipped in z/OS V1R13)

Applications

AF_INET6 PFS

AF_INET PFS

IPv6 Raw Transport

Common TCP and UDP Transport

IPv4 Raw Transport

IPv6

IPv4

Network Access Control
Multi-Level Security
IPSec
Intrusion Detection Services

Network Access Control
Multi-Level Security
IPSec
Intrusion Detection Services

Firewall Functions - IPSec

Firewall Functions - IPSec

Common DLC Functions

IPv6 DLCs

IPv4 DLCs

OSA-E QDIO

IPv4 and IPv6 packets on the same LAN

Dynamic routing

- Support IPv6 RIP (RIPng) and OSPF (OSPFv3)
  - Implementation done in OMPROUTE
    - One and the same daemon for both IPv4 (RIP and OSPF) and IPv6 (RIPng and OSPFv3)
- Based on IPv4 specifications with IPv6-specific updates
  - IPv6 RIP includes minimal changes
    - Replacement for RIPv1 and RIPv2 used in IPv4 networks
  - IPv6 OSPF is protocol independent
    - Separate IP addressing and network topology where possible
- Needed for Sysplex-related functions such as dynamic IPv6 VIPA movement

2001:0DB8:1::9:67:101:7

2001:0DB8:1::12:14

2001:0DB8:1::7

2001:0DB8:1::12:17

Prefix 2001:0DB8:1::/64 - aucoconfig

2001:0DB8:1::12:13

Note: every interface also has a
        link-local address, not shown here

2001:0DB8:1::12:19

Managing your IPv6 network

- Network management SNMP support
  - Support SNMP agent (OSNMPD)
  - DPI 2.0 enabled for AF_INET6 (used between SNMP subagents and SNMP manager)
  - Support TCPIP (stack) subagent
  - osnmp command
  - The trap forwarder daemon enabled for AF_INET6
  - IPv6 MIB support
    - New RFCs have been published that are IP version neutral - support will gradually converge from supporting version-specific MIBs to the new version-neutral MIBs
      - *RFC 2011 (IP and ICMP)*
      - *RFC 2012 (TCP)*
      - *RFC 2096 (IP routes)*
      - *RFC 2333 (Interfaces) - this one is not version neutral*

Testing network connectivity

- Ping and Traceroute support for IPv6
  - IPv6 IP addresses, or host names that resolve to IPv6 IP addresses, can be used for destinations
  - IPv6 IP addresses can be used as the source IP address for the command's outbound packets
  - IPv6 IP addresses or interface names can be used as the outbound interface
  - A new ADDRTYPE/-A command option can be specified to indicate whether an IPv4 or IPv6 IP address should be returned from host name resolution

- IPv4-mapped IPv6 IP addresses are not supported for any option value

Planning how to get there

Steps for moving to an IPv6 Environment

- Network access
  - A LAN can carry both IPv4 and IPv6 packets over the same media
  - An OSA-Express port can be used for both IPv4 and IPv6
  - Update TCP/IP Profile to include the INTERFACE statement(s) for any IPv6 interfaces
  - For LPAR-LPAR communication for IPv6, several options exist:
    - Using QDIO to a shared LAN (or a Shared OSA)
    - MPCPTP6 interfaces (via XCF if in the same sysplex or ESCON CTC links)
    - IPv6 HiperSocket connections (if in the same CEC)

Two logical networks
  ► an IPv4 network
  ► an IPv6 network

IPv4

IPv6

A separate IPv4 network - assign a separate subnet to this IPv4 network
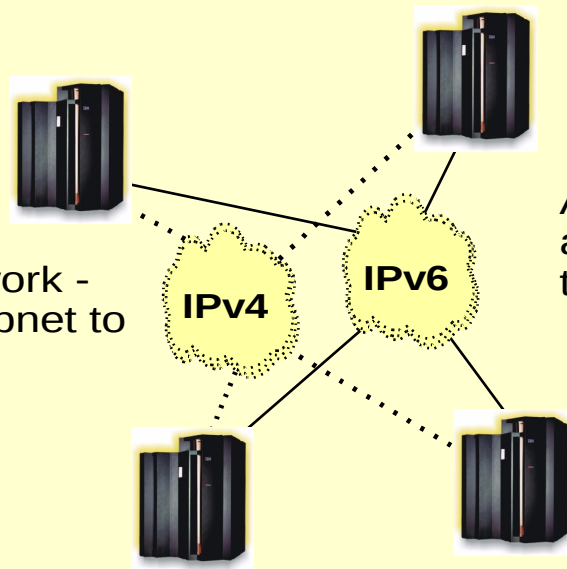
A separate IPv6 network - assign a separate prefix to this IPv6 network

The IPv6 dynamic VIPA functions require that an IPv6 Dynamic XCF network is defined between participating stacks.

Steps for moving to an IPv6 Environment…

- IPv6 address selection
  - Obtain an address block from your ISP or create local IPv6 unicast addresses using the FC00::/7 prefix
    - You provide the MAC address of one of your OSA adapters, and sites such as http://www.kame.net/~suz/gen-ula.html can generate a unique local address for you
    - Remember: IPv6 site local addresses have been deprecated and should not be used
  - IPv6 addresses can be assigned to the IPv6 Interfaces and static VIPAs
  - Addresses can be manually configured on the INTERFACE statement in the TCP/IP Profile or auto-configured using Neighbor Discovery Stateless Auto-configuration
    - HiperSocket interfaces must be manually configured
    - VIPA addresses must be manually configured

Steps for moving to an IPv6 Environment…

- INET or Common INET
  - Both are supported for IPv6, but INET is much simpler
  - Running IPv4-only and dual-mode stacks under CINET is not recommended - run dual-mode stacks in a separate LPAR from IPv4-only stacks
    - All z/OS TCP/IP stacks in an LPAR are either IPv4-only or dual-mode (based on your BPXPRMxx definitions)
    - Only case where this could become an issue is if you start CA's TCPAccess TCP/IP stack side-by-side with a z/OS TCP/IP stack in an LPAR that have been enabled for IPv6 in the BPXPRMxx parmlib member
  - AF_INET6 NETWORK statement must be coded in BPXPRMxx before starting IPv6-enabled stacks

Steps for moving to an IPv6 Environment…

- TCP/IP profile – IP Configuration for IPv6
  - Basic IPv6 configuration is done using the IPCONFIG6 statement
    - Similar to IPCONFIG, which continues to be used for IPv4
    - Separate statements for IPv4 and IPv4 allow different values to be specified for IPv4 and IPv6
  - Most of the defaults on the IPCONFIG6 statement are good choices
    - However, we recommend that you code the SOURCEVIPA parameter
      - SOURCEVIPA allows a VIPA to be used as the source IP address for

      - It is not enabled by default, but is more important in an IPv6 environment

  - You may want to enable IP forwarding using the DATAGRamfwd parameter
    - The default is to **not** forward IP packets, the same as for IPv4

```
IPCONFIG6 SOURCEVIPA
          DATAGRAMFWD
```

Steps for moving to an IPv6 Environment…

- DNS setup
  - A DNS BIND 9 Name Server can be used for both IPv4 and IPv6 resources
  - Continue to use the existing host name for IPv4 connectivity to avoid possible disruption in network connectivity and IPv4-only applications on an IPv6-enabled stack
  - Create a new host name to be used for IPv6 and IPv4 connectivity
  - Optionally, a third host name which may be used only for IPv6 can be configured
  - If using stateless auto-configuration to define IPv6 addresses, static VIPA addresses should be stored in DNS since the auto-configured addresses will change over time

- System Resolver setup
  - Resolver sends queries to DNS server using IPv4 or IPv6 (IPv6 support added in z/OS V1R12)
    - If the IPv4 protocol is used to communicate, it does not affect what type of records are returned (you can still resolve host names to IPv6 addresses and vice-versa)
  - Resolver communication with DNS name servers
    - Name query sends AAAA query to DNS and receives AAAA records in response
    - Reverse query sends PTR query to the 'ip6.arpa' domain and receives results from the 'ip6.arpa' domain
  - Resolver support for caching both IPv4 and IPv6 addresses returned from a DNS query

Steps for moving to an IPv6 Environment…

- Resolver setup….
    - Use /etc/ipnodes as the local host file for IPv6 name queries
        - Local database that associates host names with IP addresses
        - May be used to locate both IPv4 and IPv6 addresses (using the COMMONSEARCH System Resolver option)
        - Extended version of /etc/hosts
            - Uses the same format as /etc/hosts, but may be used to store both IPv4 and IPv6

    - /etc/hosts may continue to be used to store IPv4 addresses
        - But may not be used to store IPv6 addresses (same is true for files created with MAKESITE utility - HOSTS.SITEINFO and HOSTS.ADDRINFO)
    - May want to consider using a local host file for early testing
        - Use the *LOOKUP LOCAL|DNS* resolver directive

```
9.67.43.100          HOSTNAME1
9.67.43.126          HOSTNAME2
9.67.43.222          HOSTNAME3          HOSTNAME3_IPV4
129.34.128.245                HOSTNAME4
9::67:43:126         HOSTNAME2_IPV6
9::67:43:222         HOSTNAME3_IPV6
```

Recommendations when adding IPv6 addresses to DNS

- Add Static VIPAs in DNS
  - You don't need to add addresses assigned to physical interfaces if using VIPA and SOURCEVIPA
  - z/OS auto-configured addresses are not suitable for placement in DNS
    - May (and likely will) change each time a z/OS stack is recycled
    - If you need to place addresses assigned to physical interfaces in DNS, then you should manually configure the addresses

- Configure two (and optionally three) host names in DNS
  - Continue to use the existing host name for IPv4 connectivity
  - Create a new host name to be used for IPv6 and IPv4 connectivity
  - Optionally, a third host name which may be used only for IPv6 can be configured

- Be careful when adding Unique Local Unicast addresses to DNS
  - Unique Local unicast addresses are not globally unique and must not be returned to hosts outside the local site
    - Similar to how private addresses are handled in IPv4

- **Never** add link-local addresses to DNS
  - They can't be used beyond the link on which they are defined, and aren't intended for general-purpose applications

Resolver communication with DNS Name Server

- The results of System Resolver queries varies based on interface availability
  - Resolver may omit IPv4 or IPv6 results if there aren't any physical interfaces which support the network protocol
  - Resolver sorts the addresses returned based on local interface availability
    - Default Address Selection algorithms govern both source address selection and destination address selection
    - Destination Address Selection is performed by Resolver as part of the name-to-address mapping
    - Source Address Selection is performed by the TCP/IP stack after the destination address is chosen
    - RFC 3484 *"Default Address Selection for Internet Protocol version 6 (IPv6)"* defines configurable rules for how parts of the source and destination IP address selection logic works – the default source and destination IP address selection
      - This rule-based logic kicks in after all the existing z/OS TCP/IP logic for selection of source and destination IP addresses has been exhausted

Steps for moving to an IPv6 Environment…

- Selection and placement of IPv6 to IPv4 translators or application gateway
    - z/OS does not implement any functions that will allow IPv6-only nodes to communicate with z/OS-resident AF_INET applications, so an outboard protocol converter or application-layer gateway component may be needed
    - This component will only be needed if the test configuration includes IPv6-only platforms
    - Various technologies are being made available by various vendors

- Connectivity to non-local IPv6 locations
    - Tunneling may be needed between a router connected to the LAN that z/OS is connected to, and a router at another location where IPv6 test equipment is located

HTTP Gateway or Proxy

IPv6
IPv4

Inter-net   DMZ   Appl. zone   Data zone
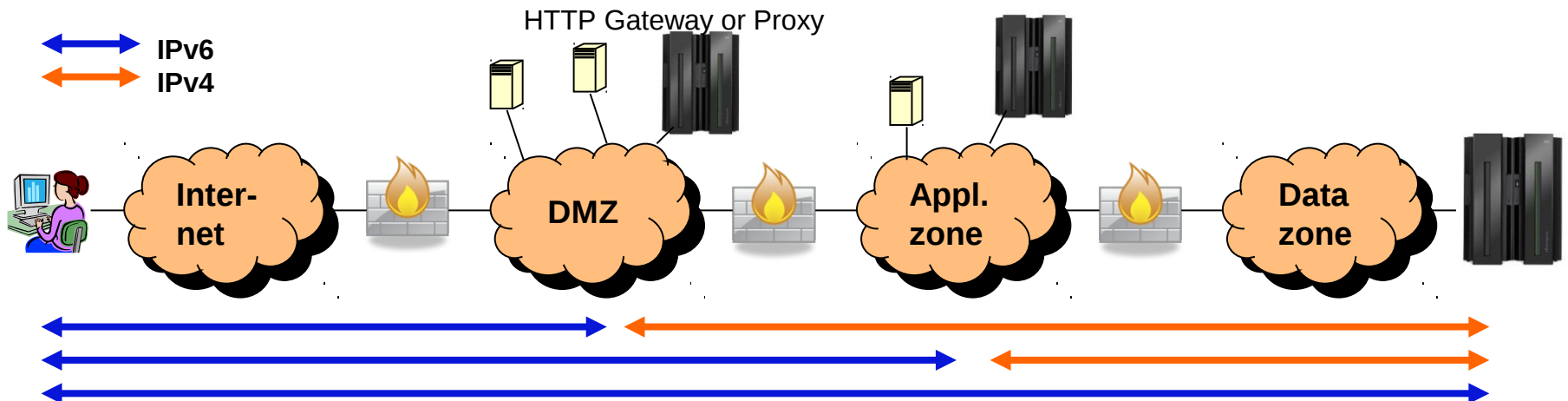
Sockets API considerations when moving to AF_INET6

- IPv6 addresses are 128-bit in size as compared to 32 bits for IPv4
  - Data structures which store IP addresses must be modified to handle the larger size
- DNS Resolver library changes
  - New DNS routines - getaddrinfo() and getnameinfo()
- Textual representation of the IP address has changed
  - IPv4 addresses use dotted-decimal format
  - IPv6 addresses use colon-hex notation
- IPv6 has several scopes for IP addresses
  - An address is only unique within its given scope
  - On multi-homed hosts, an IP address alone may be insufficient to select the interface over which to route
- IP addresses should not be assumed to be permanent
  - Long-term use of an address is discouraged due to renumbering
  - Applications should rely on DNS resolvers to obtain the appropriate IP addresses
- sockaddr_in6
  - Analogous to sockaddr_in, but holds 128-bit IPv6 address, port numbers, plus Flow Label and Interface Identifier
- in6_addr
  - Analogous to 32-bit in_addr, but holds 128-bit IPv6 address
- Socket calls to investigate for possible changes
  - socket(), bind(), connect(), sendmsg(), sendto(), accept(), recvfrom(), recvmsg(), getpeername(), getsockname()
- New IP address conversion routines
  - inet_pton(), inet_ntop()

What can you do today? Start learning, planning, and testing!

- ❑ Develop a multi-step plan
  - Eventual goal is fully IPv6-enabled dual-stack operating environment
- ❑ Choose a target date for being IPv6-enabled
  - Work backwards in developing a timeline on when key steps need to be completed
- ❑ Develop detailed plan for each sub-step
  - To resolve critical dependencies in the necessary timeframe
- ❑ Not too early to begin planning today
  - Need for IPv6 may occur quickly and with little advanced warning
  - Rapid realization that IPv6 is needed
  - Take several years to actually get IPv6 deployed
  - Need to have IPv6 already in use (and tested) before it becomes a requirement that it be used operationally
- ❑ Develop an internal addressing plan for distributing/managing IPv6 addresses
  - Determine how IPv6 addresses will be obtained
  - Either from your ISP, or from a Regional Internet Registry (RIR)
  - Consider whether Unique Local Addresses are appropriate

- ❑ Understand your ISPs IPv6 plans
- ❑ Perform a detailed inventory of all systems
  - Determine what is involved in IPv6-enabling them
  - All network hardware and software
  - All client and server hardware, software and applications
- ❑ Develop plans to ensure all components are IPv6-enabled according to a workable timeline
  - Most vendors have already added IPv6 support for critical components
- ❑ Determine how end users will use IPv6 services
  - Likely involve tunneling initially
  - But need IPv6-capable routers on the edge links where clients connect
  - Need to provide remote IPv6 access
- ❑ Develop plans for IPv6 training, education and consulting

Please fill out your session evaluation

- IPv6 on z/OS
- Session # 11340
- QR Code:

Find us on Facebook at
http://www.facebook.com/IBMCommserver

Follow us on Twitter at
http://www.twitter.com/IBM_Commserver

Read the z/OS Communications Server blog at
http://tinyurl.com/zoscsblog

Visit the z/OS CS YouTube channel at
http://www.youtube.com/user/zOSCommServer