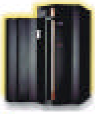
z/OS CS - SHARE in San Francisco - August 2002

Session 3912 - TCP/IP Project

IP Network Design Considerations, Virtual IP Addresses

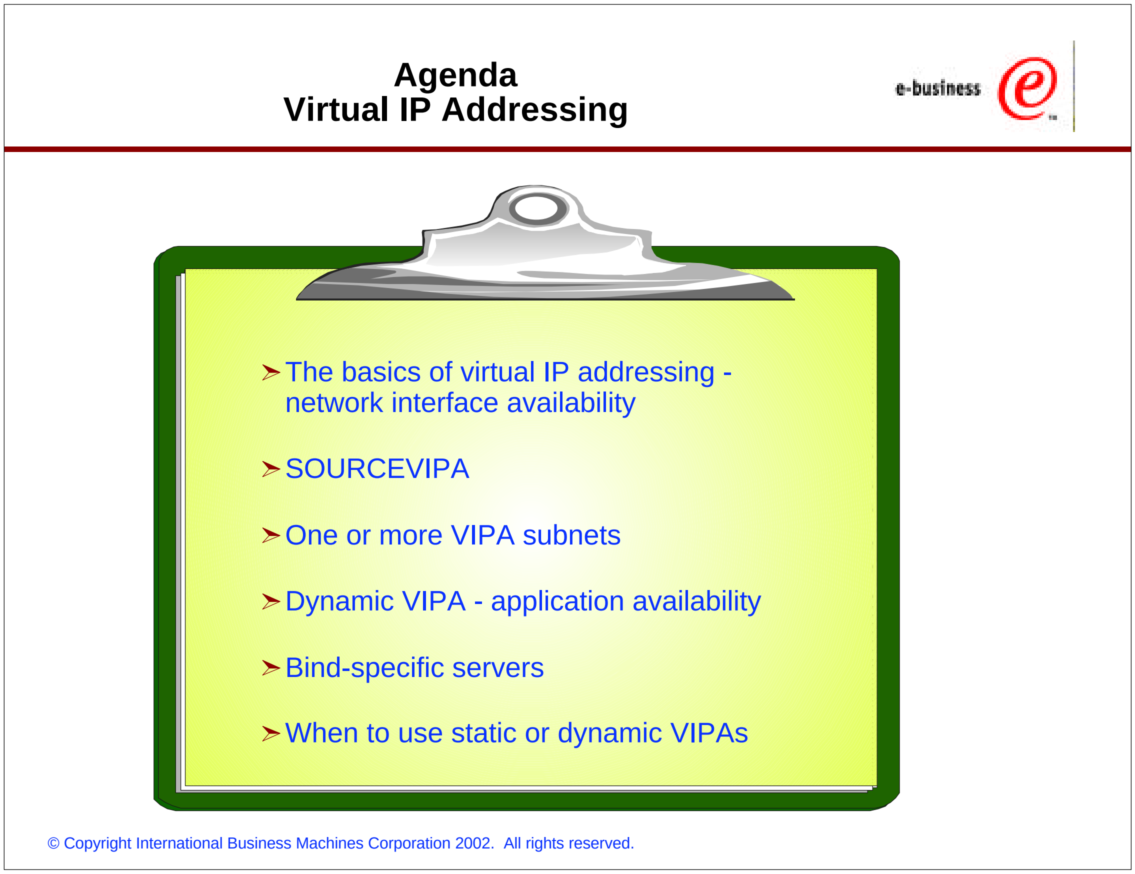
Wednesday, August 21st - 11:00 AM

WebSphere and eServer Networking Solutions, Raleigh
Alfred B Christensen - alfredch@us.ibm.com



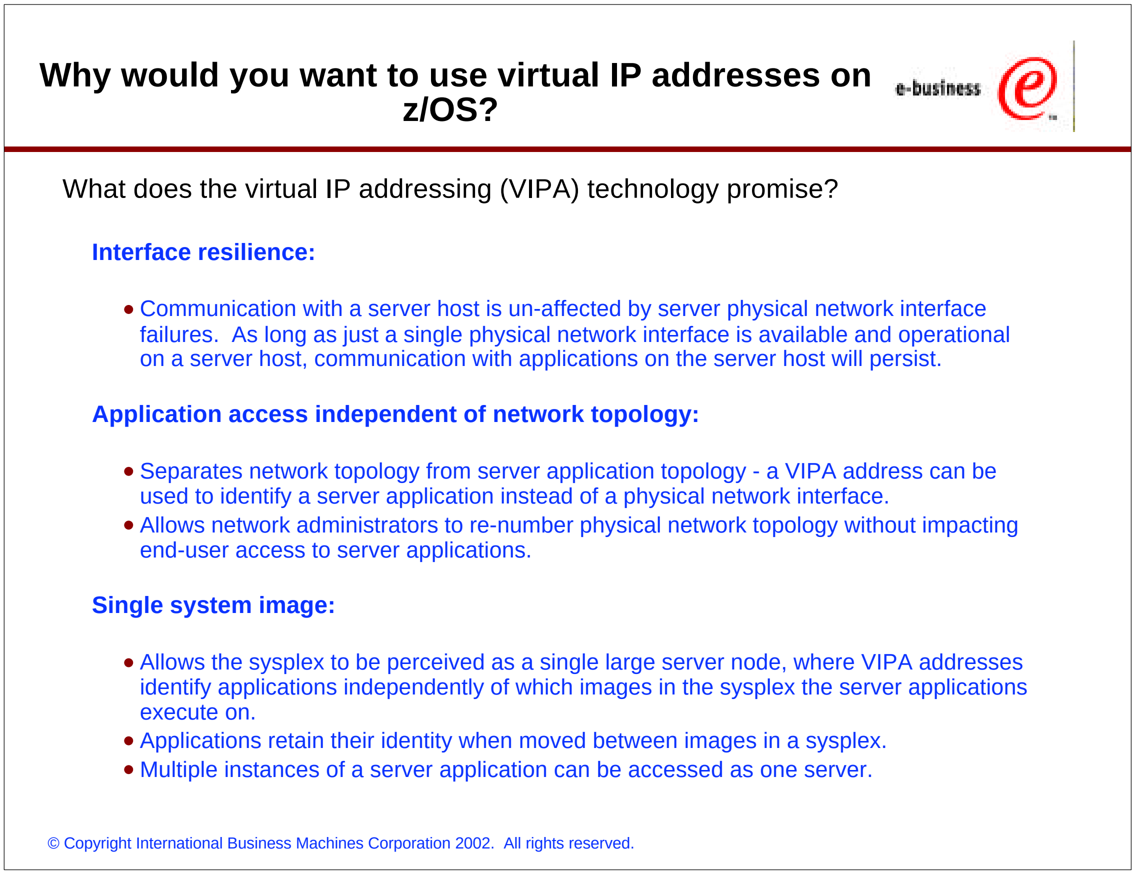
IP Network Design Considerations: VIPA

Session Number:	3912
Date:	Wednesday, 21-August-2002
Time:	11:00 AM
Location:	Parc Fifty-Five - 4th Floor - Dante
Speaker:	Alfred B Christensen, IBM
Chair:	Paul Bouwmeester
Abstract:	Session 3912 will focus on using Virtual IP Addressing to improve availability and ease administration in your network. This session will introduce the basic concept of a virtual IP address and give guidelines on how to assign VIPA addresses. The session will further discuss the recent enhancements to the VIPA concept, including the dynamic VIPA functions of OS/390 V2R8 and the later distributed VIPA functions. Finally the session will discuss the use of the SOURCEVIPA configuration option.



Agenda
Virtual IP Addressing

- ➢ The basics of virtual IP addressing - network interface availability
- ➢ SOURCEVIPA
- ➢ One or more VIPA subnets
- ➢ Dynamic VIPA - application availability
- ➢ Bind-specific servers
- ➢ When to use static or dynamic VIPAs



Why would you want to use virtual IP addresses on z/OS?

What does the virtual IP addressing (VIPA) technology promise?

Interface resilience:

- Communication with a server host is un-affected by server physical network interface failures. As long as just a single physical network interface is available and operational on a server host, communication with applications on the server host will persist.

Application access independent of network topology:

- Separates network topology from server application topology - a VIPA address can be used to identify a server application instead of a physical network interface.
- Allows network administrators to re-number physical network topology without impacting end-user access to server applications.

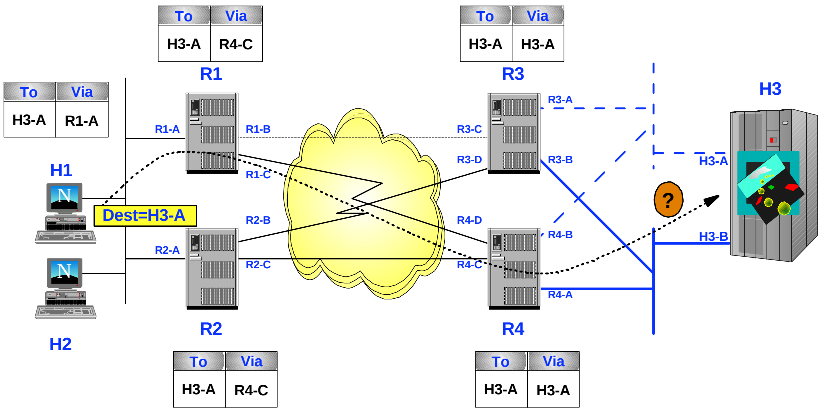
Single system image:

- Allows the sysplex to be perceived as a single large server node, where VIPA addresses identify applications independently of which images in the sysplex the server applications execute on.
- Applications retain their identity when moved between images in a sysplex.
- Multiple instances of a server application can be accessed as one server.

Even dynamic routing has limitations!

- If segments of the backbone network fail, routers in the network are able to re-route IP traffic around the failed segments in such a way that end-users do not notice any disruption of service.

- If a server network interface, such as an OSA-adapter or a channel-attached router fails and clients in the network are connected to the server using the IP address of the failed network interface, these end-users will notice a disruption of service and they will have to try and re-connect to an alternate network interface (alternate IP address) of the server host, which often involves new name server resolution requests.



H1:

To	Via
H3-A	R1-A

R1:

To	Via
H3-A	R4-C

R3:

To	Via
H3-A	H3-A

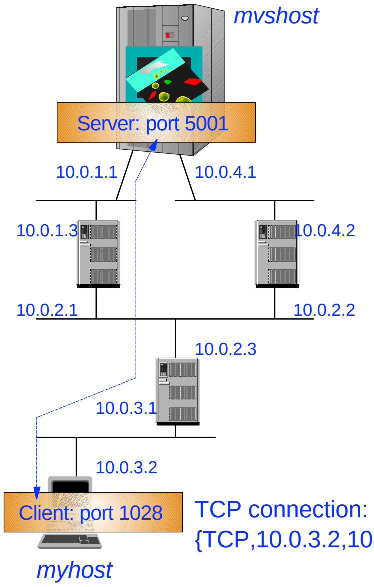
R2:

To	Via
H3-A	R4-C

R4:

To	Via
H3-A	H3-A

TCP connection recovery without VIPA



TCP connection:
{TCP,10.0.3.2,1028,10.0.1.1,5001}

Name server:
mvshost: 10.0.1.1 and 10.0.4.1

10.0.1.1 interface on *mvshost* fails:

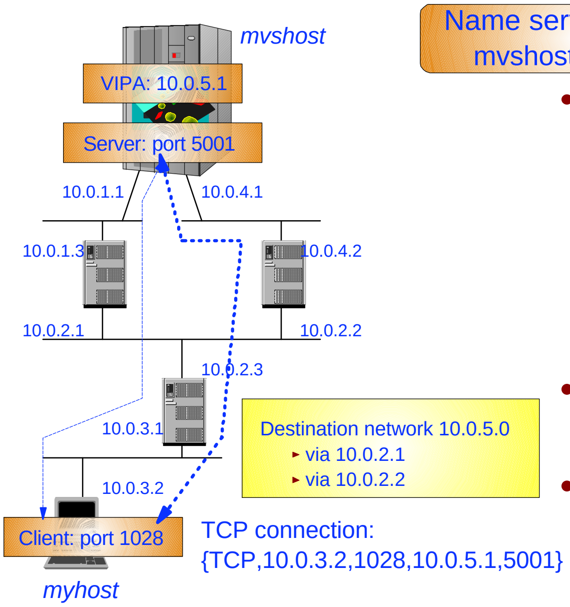
1. TCP layer on *myhost* times out
2. TCP layer on *myhost* retransmits
3. All TCP retransmits fail
4. TCP connection times out and breaks
5. Client manually establishes new connection to 10.0.4.2

The router at 10.0.3.1 does not update its routing table entry for the 10.0.1.0 subnet; that subnet is still reachable through the 10.0.2.1 router.

Subnet mask:
255.255.255.0

Inbound TCP connection recovery with VIPA

Name server:
mvshost: 10.0.5.1



Destination network 10.0.5.0
- via 10.0.2.1
- via 10.0.2.2

TCP connection:
{TCP,10.0.3.2,1028,10.0.5.1,5001}

- 10.0.1.1 interface on *mvshost* fails:

  1. TCP layer on *myhost* times out
  2. TCP layer on *myhost* retransmits
  3. Router at 10.0.3.1 accepts a new route to the 10.0.5.1 network via 10.0.2.2
  4. TCP layer retransmissions succeed

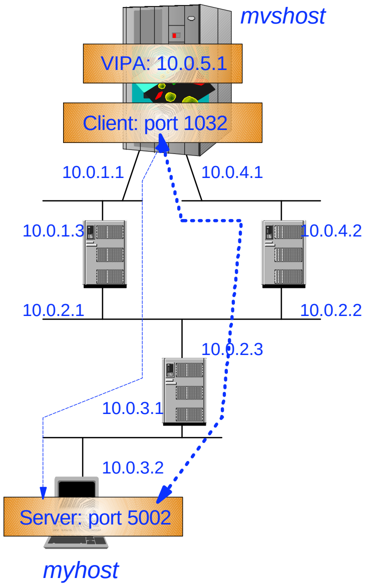
- Routing tables on myhost did not have to be updated.

- Routers' routing tables must be updated before TCP times out the connection (can be a concern with RIP dynamic routing protocols).

Subnet mask:
255.255.255.0

Outbound TCP connection recovery with VIPA (SourceVIPA)

HOME LIST in mvshost:
10.0.5.1 VIPALINK
10.0.1.1 INTFA
10.0.4.1 INTFB



- Client application on *mvshost* connects to server application on *myhost*:

- Default behavior:
  - Local IP address of socket is chosen based on the real interface over which the SYN segment is sent - in this case INTFA. The TCP connection will be based on {TCP,10.0.1.1,1032,10.0.3.2,5002} and is not recoverable.

- With **SOURCEVIPA** specified:
  - Local address of socket is chosen based on most recent VIPA link in the HOME list for the real interface over which the SYN segment is sent: {TCP,10.0.5.1,1032,10.0.3.2,5002} and the connection is recoverable.

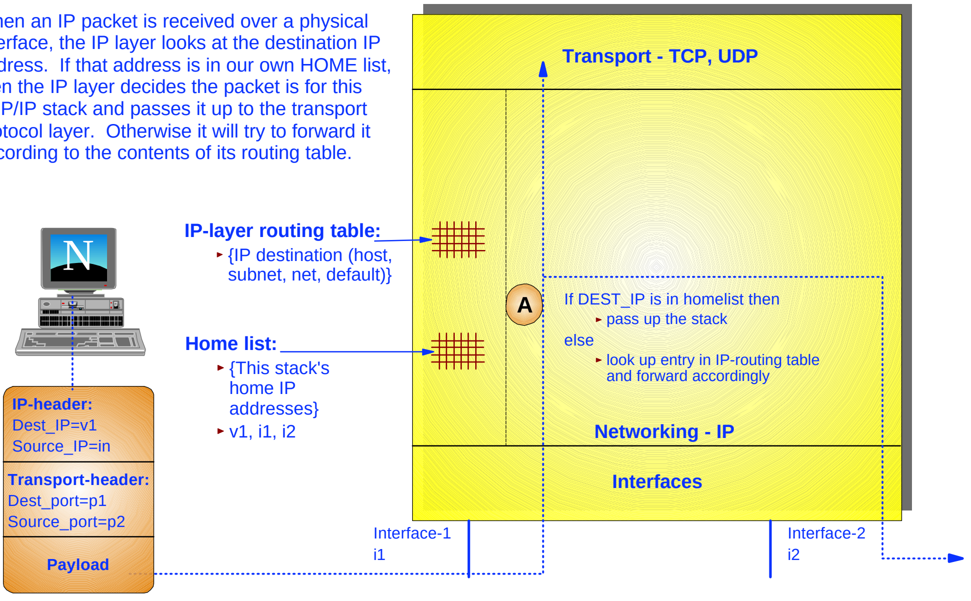
Subnet mask:
255.255.255.0

Virtual IP addressing (VIPA)

- A VIPA network is not associated with any physical network interface and therefore can not fail due to hardware malfunction - a VIPA interface is always available
- A VIPA address frees other hosts from dependence on a particular network attachment to TCP/IP on z/OS and the IP addressing of those real network interfaces
- Multiple VIPA networks and addresses can be defined for a CS z/OS TCP/IP stack
- Name servers must be configured to return the VIPA address(es) as the only IP address(es) or the first IP address(es) of a z/OS TCP/IP host
- VIPA can be used to establish fault-tolerant IP connectivity to CS z/OS TCP/IP
- Fault-tolerant implementation generally depends on dynamic routing to maintain routes in CS z/OS and to inform routers on directly connected networks about routes to the VIPA network(s).

Is IP forwarding required in order to use VIPA?

When an IP packet is received over a physical interface, the IP layer looks at the destination IP address. If that address is in our own HOME list, then the IP layer decides the packet is for this TCP/IP stack and passes it up to the transport protocol layer. Otherwise it will try to forward it according to the contents of its routing table.



IP-layer routing table:
- {IP destination (host, subnet, net, default)}

Home list:
- {This stack's home IP addresses}
- v1, i1, i2

IP-header:
Dest_IP=v1
Source_IP=in

Transport-header:
Dest_port=p1
Source_port=p2

Payload

Transport - TCP, UDP

A

If DEST_IP is in homelist then
- pass up the stack

else
- look up entry in IP-routing table and forward accordingly

Networking - IP

Interfaces

IP forwarding is ***not*** required in order to use VIPA addresses!

SOURCEVIPA

When is the SOURCEVIPA address actually being used?

- When the SOURCEVIPA option has been enabled in the IPCONFIG statement - **AND**
- An outbound connection or UDP association is being established from OS/390 - **AND**
- The application has not bound the local socket to a specific interface IP address - **AND**
- The application has not disabled the use of SOURCEVIPA through a setsockopt call

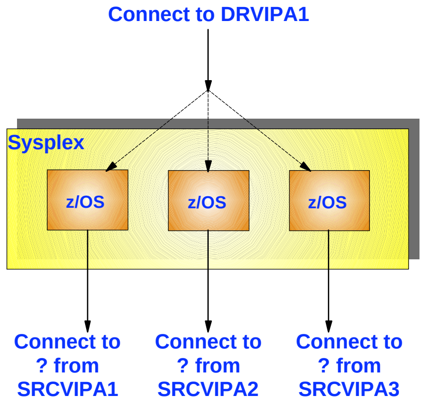
SOURCEVIPA is not being used when outbound data is sent on a connection that was established inbound to z/OS (such as data sent as a response on a TN3270(E) connection that was established from a remote TN3270(E) client to the TN3270(E) server on z/OS). Such IP packets are sent from the IP address the connection was established to.

An FTP outbound data connection is always established from a socket that was bound to the same server IP address as to where the control connection was directed.

The SOURCEVIPA address to use is selected based on the order of the HOME list:

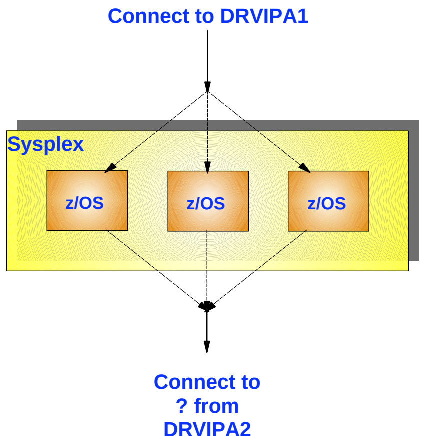
```
10.0.0.1    VIPALINK1
192.168.1.1 REALLINKA  ; Will use 10.0.0.1 as SOURCEVIPA
10.0.0.2    VIPALINK2
192.168.2.1 REALLINKB  ; Will use 10.0.0.2 as SOURCEVIPA
```

Single system image from an IP perspective in the sysplex

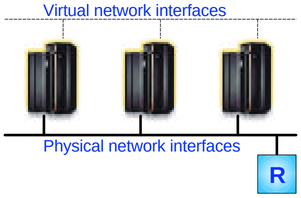

Connect to DRVIPA1
Sysplex
z/OS
z/OS
z/OS
Connect to ? from SRCVIPA1
Connect to ? from SRCVIPA2
Connect to ? from SRCVIPA3

- We have single system image capability for inbound connections where a single distributed VIPA address can represent all images in the sysplex - and remote users do not need to select a specific image when connecting to their server application.
- But if we establish outbound conections from the images in the sysplex, each image has its own source VIPA address - so there is no single system image from an outbound connection perspective - which has implications in firewall filter setup, etc.

- z/OS V1R4 will introduce new capabilities that will allow a single sysplex-wide source VIPA address to be used for outbound TCP connections by all images in the sysplex - resulting in single system image capabilities for both inbound and outbound connections.



Which IP addresses to use as VIPA

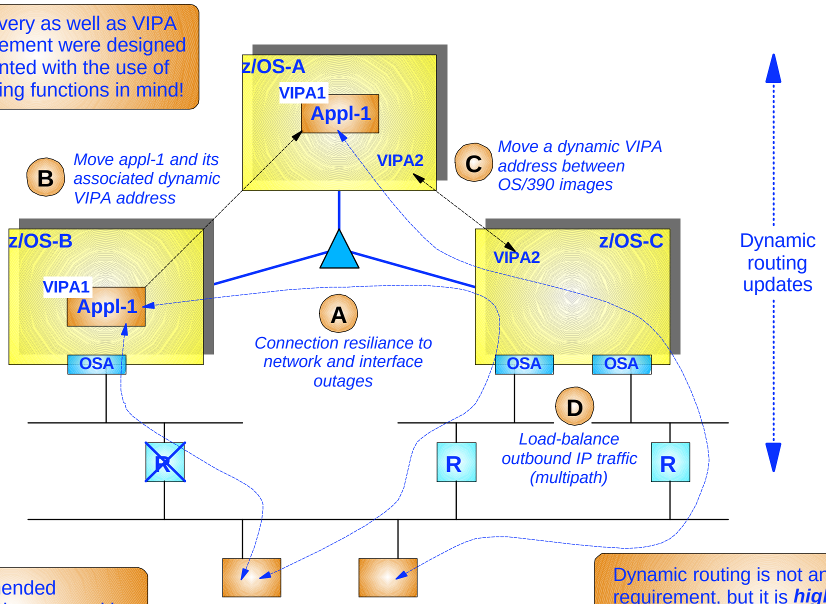


You can pictorially think of the VIPA addresses as addresses that belong to a non-existing network behind the z/OS images on which the VIPA addresses are defined.

- ➢A VIPA address is a normal IPv4 address. It has a link name and it is included in the HOME list of a z/OS TCP/IP stack.
- ➢A z/OS node may have multiple VIPA addresses defined.
- ➢VIPA addresses in a sysplex may all come out of one and the same subnet - or each member in the sysplex may initially be assigned individual subnets - there is no difference in behavior, both approaches will work perfectly fine.
- ➢When dynamic VIPA addresses are used, don't expect to be able to maintain one subnet per z/OS image.
- ➢You may use multiple subnets for VIPA addresses in a sysplex; they do not need all to come out of the same subnet.
- ➢A physical network interface and a VIPA interface should not use the same subnet if dynamic IP routing is enabled.
- ➢VIPA addresses must be advertized by the z/OS routing daemons as 32-bit prefix destinations (host routes). RIP requires the -h flag to do so, OSPF always does so by default.

Is dynamic routing protocols required on z/OS in order to use VIPA?

Base IP recovery as well as VIPA address movement were designed and implemented with the use of dynamic routing functions in mind!

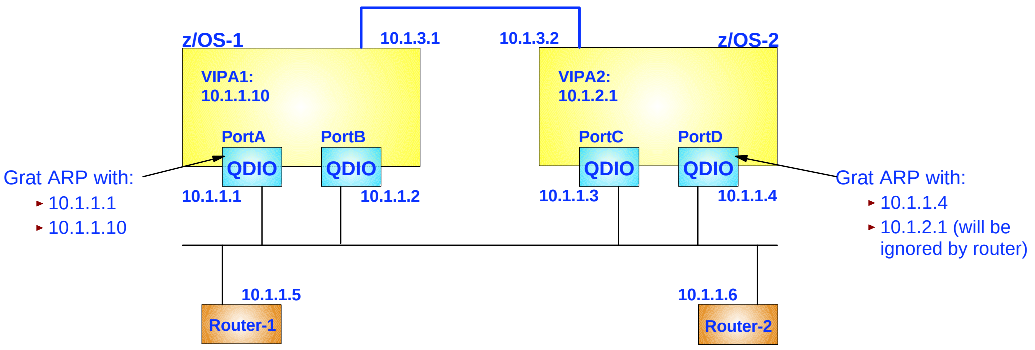

z/OS-A
VIPA1
Appl-1
VIPA2
B
Move appl-1 and its associated dynamic VIPA address
C
Move a dynamic VIPA address between OS/390 images
z/OS-B
VIPA1
Appl-1
OSA
z/OS-C
VIPA2
OSA
OSA
A
Connection resiliance to network and interface outages
D
Load-balance outbound IP traffic (multipath)
R
R
R
Dynamic routing updates

The recommended dynamic routing protocol in the sysplex is **OSPF**.

Connect to VIPA-1 (Appl-1)

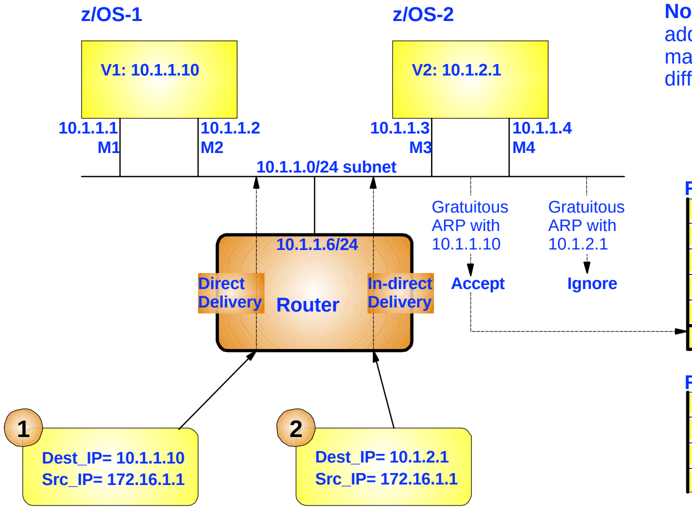
Dynamic routing is not an absolute requirement, but it is ***highly recommended*** when using VIPA addresses (it makes life a whole lot easier)!

Some QDIO basics with respect to VIPA addresses



- All HOME IP addresses will be registered in the OATs dynamically and the OAT content will be changed as the HOME lists change due to movement of IP addresses.
- When an IP address is registered, the adapter will do a gratuitous ARP if the address belongs to the same network as to which the adapter is attached (in this example the 10.1.1.0/24 subnet) or if the address is a VIPA address (independent of which subnet the VIPA address is defined on).
- Gratuitous ARPs are done for two purposes:
  1. to enable down-stream routers to update their ARP cache if an adapter malfunctions and the TCP/IP stack decides to move an address to another adapter (example: if PortA fails, then 10.1.1.1 will be moved to PortB and PortB will grat ARP 10.1.1.1) - Note that down-stream routers normally will ignore gratuitous ARPs for IP addresses that do not belong to the subnet on that physical network (in this example the 10.1.1.0/24 subnet)
  2. to check for duplicate IP addresses on the subnet - will continue for up to 15 seconds, but the adapter will accept incoming packets for the new address immediately

QDIO interfaces seen from a down-stream router's perspective



Note: Some router vendors may have added capabilities in this area that make their product work slightly different from what is described here!

Router's ARP Cache

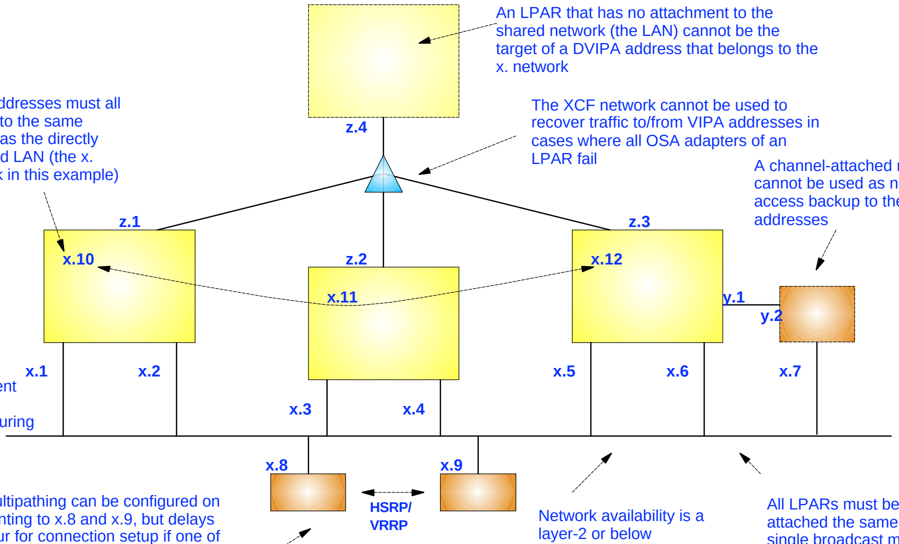
IP address	Mac address
10.1.1.1	M1
10.1.1.2	M2
10.1.1.3	M3
10.1.1.4	M4
10.1.1.10	M1

Router's IP Routing Table

Destination	Via
10.1.1.0/24	Direct delivery
10.1.2.0/24	10.1.1.3
Default	n.n.n.n

z/OS VIPA addresses in a flat network configuration without dynamic routing should be alloacted out of the same subnet as the directly attached network that all members of the Sysplex are attached to - in this example, the 10.1.1.0/24 subnet.

Considerations for use of VIPA addresses in a Sysplex without use of dynamic routing



An LPAR that has no attachment to the shared network (the LAN) cannot be the target of a DVIPA address that belongs to the x. network

VIPA addresses must all belong to the same subnet as the directly attached LAN (the x. network in this example)

The XCF network cannot be used to recover traffic to/from VIPA addresses in cases where all OSA adapters of an LPAR fail

A channel-attached router cannot be used as network access backup to the VIPA addresses

Must be QDIO to support DVIPA movement w/o reconfiguring OAT

Static multipathing can be configured on z/OS pointing to x.8 and x.9, but delays may occur for connection setup if one of the routers turns into a dead gateway (use HSRP or VRRP between the routers to increase availability)

Network availability is a layer-2 or below responsibility (parallel trunked L2 switches).

All LPARs must be attached the same single broadcast media (the same LAN)

x, y, and z in this diagram denotes three different network prefixes (three different subnets)

The main rules for use of VIPA without dynamic routing on OS/390 or z/OS

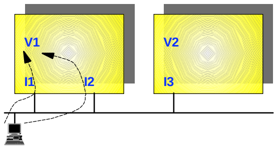
VIPA-based application availability functions implemented on top of layer-2 network attachment availability functions without use of dynamic routing on OS/390 or z/OS can be implemented given the following design and implementation restrictions are met:

1. Although the layer-2 network attachment availability functions are implemented for both LCS and QDIO - only QDIO interfaces are recommended when dynamic VIPA addresses are used.
2. All images in the sysplex (to which a DVIPA may move) must be attached to the same broadcast media - an image that is only attached through XCF or MPC links without attachment to the broadcast media cannot be the target of a DVIPA movement.
3. All VIPA addresses should be assigned to the same (sub)net as the broadcast media. OS/390 and z/OS has some support for responding to ARP requests for VIPA addresses on (sub)nets other than the broadcast media, but experience has shown that some router vendors have problems when OS/390 or z/OS sends out gratuitous ARPs for (sub)nets that the routers didn't initially consider to be on that broadcast media. To avoid problems with such routers, all VIPA addresses should be assigned out of the same (sub)net as the broadcast media uses.
4. Alternative network attachment technologies, such as channel-attached routers, can not participate as network attachment backup in case the adapters to the broadcast media fail.
5. The immediate downstream network (the broadcast media) must be one logical network (one broadcast media) to which all the participating OS/390 or z/OS images are attached. For network attachment availability purposes, each image should have two real interfaces to that single broadcast media.
6. Availability of the broadcast media becomes essential to network availability and must be addressed through appropriate layer-1 and layer-2 functions (such as, multiple switches trunked together). If that broadcast media becomes unavailable as an entity, there is no network connectivity to any of the OS/390 or z/OS images.
7. Without dynamic routing, all routes from OS/390 and z/OS must be configured using static routing tables. Care must be taken if functions such as Cisco's MNLB is used. Functions like MNLB use some "hidden" IP addresses for communication of control information (CASA protocol-based). These addresses need to be identified and included in the static routing tables of OS/390 and z/OS if in use.
8. Without dynamic routing, care must also be taken if use of the MULTIPATHING feature is implemented on OS/390 or z/OS.

With dynamic routing use on OS/390 or z/OS - none of the above restrictions apply!

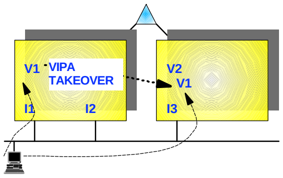
The evolution of VIPA
static VIPA - dynamic VIPA

TCP/IP V3R2 VIPA Support



- VIPA addresses primarily used to represent the OS/390 host - some limited deployment of application-specific VIPA, but no specific support in place for that use.
- Connection resilience to failure of network interface.
- If an application was to be moved from one OS/390 image to another, the DNS could be updated to point to V2 instead of V1.
- In TCP/IP V3R2, you could manually (through OBEYFILE commands) move a static VIPA address from one stack to another - the concepts were identical to dynamic VIPA, but the movement was completely manual.

CS for OS/390 V2R8 Dynamic VIPA Support



- A VIPA address can either represent an OS/390 host or an individual application where the name server is updated to include resource records that identify individual applications, such as, myCICS.xyz.com at IP address V1.
- VIPA still addresses connection resilience but now also addresses application recovery. If an OS/390 image is taken down, Dynamic VIPA backup policies can be used to define where the associated DVIPAs move within the sysplex. Dynamic VIPA support also allows for manual movement of applications and associated DVIPA addresses.

In CS for OS/390 V2R8, only one stack at any point in time owns a specific VIPA address - in V2R10 and later, a VIPA address may exist on multiple images in the sysplex concurrently (but only one will advertize it).

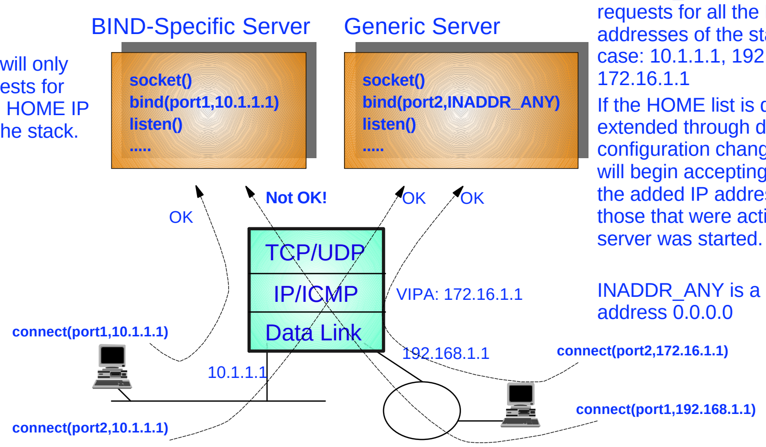
Dynamic VIPA usage

A dynamic VIPA address has all the attributes of a static VIPA address. In addition to those it has the ability to move between TCP/IP stacks in a sysplex based on certain events - without operator intervention in terms of configuration changes.

When do you want the dynamic VIPA to move?	What's the type of DVIPA to use?	How do you define it?	Application requirements	Typical use
Move to a backup stack, when the currently owning stack goes down or is taken down.	**A stack-managed DVIPA**	VIPADEFINE on primary owner - VIPABACKUP on potential backup stacks.	Applications bind to INADDR_ANY.	Multiple instances of server runs on multiple stacks and can back each other up.
Move along with a specific server application that binds its listening socket to the dynamic VIPA address.	**An application-specific DVIPA**	VIPARANGE	Applications must bind to the specific dynamic VIPA address (alternatively use BIND specific on port reservation)	Single instance application that is moved between stacks - planned or unplanned.
Move when instructed to do so by executing a utility (moddvipa) or by an authorized application (using an ioctl call)	**A command-activated DVIPA**	VIPARANGE	No special requirements, but typically application binds to INADDR_ANY.	Single instance applications that cannot be controlled via bind specific functions.

Bind-specific or generic servers

This server will only accept requests for the 10.1.1.1 HOME IP address of the stack.



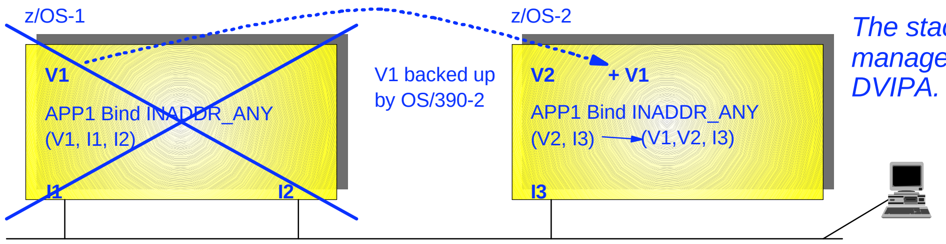
A generic server (also known as an INADDR_ANY server) will accept requests for all the HOME IP addresses of the stack, in this case: 10.1.1.1, 192.168.1.1, and 172.16.1.1

If the HOME list is dynamically extended through dynamic configuration changes, the server will begin accepting requests for the added IP address in addition to those that were active when the server was started.

INADDR_ANY is a symbol for IP address 0.0.0.0

If you try to start two or more servers that both bind to the same port number on the same IP address (incl. INADDR_ANY), only the first server will succeed in starting up, the others will end with address_in_use errors. Multiple servers can bind to the same port number, if they bind to different specific HOME IP addresses.

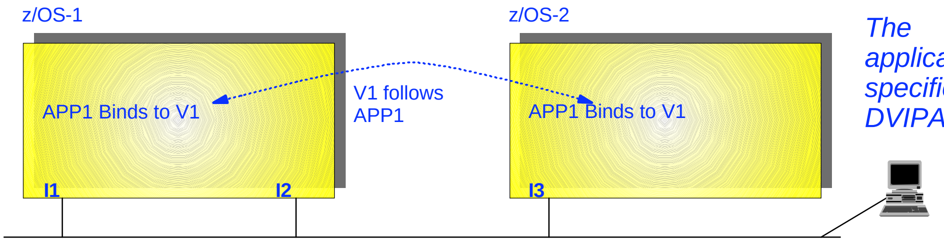
VIPADEFINE/VIPABACKUP use with INADDR_ANY applications



The stack-managed DVIPA.

- Configured through VIPADEFINE on primary stack, and VIPABACKUP on backup stack(s).
- When primary stack (the one with VIPADEFINE) is started and the address is not active in the sysplex, the address wil be activated on the primary stack during start-up.
- APP1 binds to INADDR_ANY (all IP addresses of a stack).
- If APP1 is started on both z/OS-1 and z/OS-2, it will initially respond to connections to V1 on z/OS-1 and to V2 on z/OS-2.
- If z/OS-1 fails, existing connections with V1 will break, but z/OS-2 will automatically initiate a takeover of V1 and V1 will be dynamically added to the addresses that APP1 on z/OS-2 will accept connection requests for, and the APP1 instance on z/OS-2 will thereafter serve connection requests for both V1 and V2 (picking up the workload from z/OS-1).
- An alternative to having APP1 already started on z/OS-2 would be to have automation start APP1 on z/OS-2 when z/OS-1 fails (either based on ARM policies or message automation).
- When a server can execute in multiple instances, this can be used to merge all work from a failed instance to another active instance on another member of the sysplex - one obvious example is the TN3270 server.

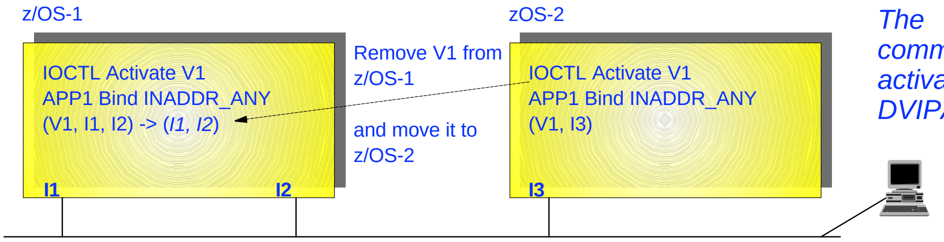
Have a dynamic VIPA stick with a single-instance application



The application-specific DVIPA.

- Configured through VIPARANGE for reservation of allowed dynamic VIPA addresses.
- Addresses defined in VIPARANGE are not automatically activated during stack startup.
- APP1 binds to a specific IP address, such as V1.
- This support initially required that the application either already has bind-specific support or that you have access to the source code so you can change the application to include bind-specific support. By combining this with the new Bind-specific definitions in the PORT reservation section of your TCP/IP Profile, this support can be used by applications that currently bind to INADDR_ANY.
- When APP1 is started on an image in the sysplex, V1 will be activated on that image depending on the definition of V1 (MOVEABLE DISRUPTIVE/NONDISRUPTIVE) and the state of V1 on other images in the sysplex.
- An application can be moved from z/OS-1 to z/OS-2 by having an operator stop it on z/OS-1 and restart it on z/OS-2. V1 will be moved with the application to z/OS-2.
- If z/OS-1 fails, the application can be immediately started on z/OS-2.
- An application that can only be running in a single instance in a sysplex can be moved between images using this support. An example of such an application could be a CICS application that uses CICS resources that cannot be shared with other CICS regions.

IOCTL or command-activated dynamic VIPA movement



The command-activated DVIPA.

- Also configured through VIPARANGE for reservation of allowed dynamic VIPA addresses.
- For some reason, APP1 cannot use bind specific support, but you want to ensure that one specific dynamic VIPA address is associated with APP1 when it is started no matter which z/OS image it is started on.
- To activate the specific dynamic VIPA address you either add a job step to the JCL that starts the server or you change the server code itself to issue the IOCTL call. CS for z/OS supplies a utility program called EZBXFDVP (or UNIX shell command: moddvipa) to be used as a separate job step, which based on your runtime options will activate the dynamic VIPA you need. This utility can also be used to deactivate dynamic VIPA addresses on-demand.
- A program that issues this IOCTL call must either be APF authorized or execute with UNIX root authority.
- When the IOCTL call is issued, the stack immediately activates the DVIPA address. Depending on how V1 is defined, this is disruptive to any existing connections using this VIPA address on other images in the sysplex. The dynamic VIPA address is immediately added to the stack that processes the IOCTL call, and the succeeding bind to INADDR_ANY from the server program will include the newly activated dynamic VIPA address.

Dynamic VIPA definitions

e-business

```
VipaDynamic
 VipaDefine Moveable WhenIdle 255.255.255.0 192.168.101.1
 VipaBackup 1 192.168.102.2
 VipaBackup 1 192.168.103.3
 VipaRange  Define Moveable Nondisruptive 255.255.255.0 192.168.110.21
 VipaDefine Moveable Immediate 255.255.255.0 9.67.116.32
 VipaDistribute Define 9.67.116.32 Port 23 DestIP All
EndVipaDynamic
```

- Activate 192.168.101.1 during initialization - allow it to be moved when there are no more connections..
- Act as backup for 192.168.102.2 and 192.168.103.3 - if the owning stack(s) go down, these IP addresses will be activated on this stack.
- Allow applications to bind to 192.168.110.21 or to activate the address via an ioctl() function.
- Activate 9.67.116.32 and distribute connections to port 23 to all images in the sysplex

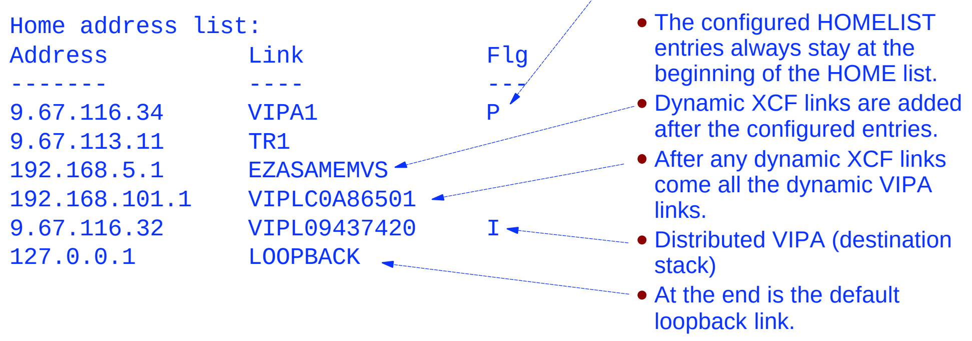
Why does my HOMELIST look strange?

e-business

My configured HOMELIST:

```
HOME
  9.67.116.34    VIPA1        ; Static VIPA
  9.67.113.11    TR1          ; Tokenring
```

Output from NETSTAT HOME:



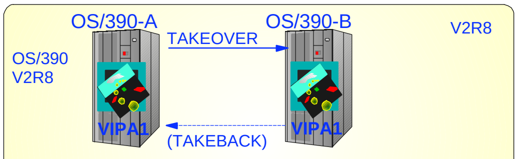
```
Home address list:
Address          Link             Flg
-------          ----             ---
9.67.116.34      VIPA1            P
9.67.113.11      TR1
192.168.5.1      EZASAMEMVS
192.168.101.1    VIPLC0A86501
9.67.116.32      VIPL09437420     I
127.0.0.1        LOOPBACK
```

The only purpose of the PRIMARYINTERFACE statement is to specify which IP address will be returned on a gethostid() call. If no PRIMARYINTERFACE statement is specified, it will always be the IP address of the first entry in the HOME list.

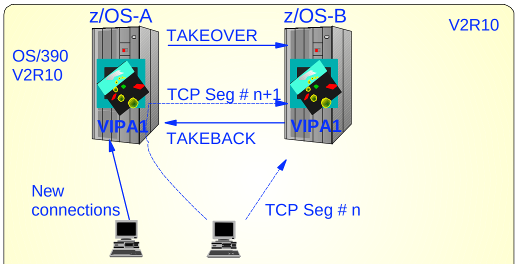
The P flag tells you which IP address will be returned on a gethostid() call:

- The configured HOMELIST entries always stay at the beginning of the HOME list.
- Dynamic XCF links are added after the configured entries.
- After any dynamic XCF links come all the dynamic VIPA links.
- Distributed VIPA (destination stack)
- At the end is the default loopback link.

Dynamic VIPA non-disruptive movement in CS for OS/390 V2R10



1. B takes over VIPA1 when A goes down.
2. When A comes back up, a request for getting VIPA1 back is queued to B.
3. B doesn't give VIPA1 back until there are no more connections with VIPA1 on B - which may take some time.



1. When A comes back up again, it immediately takes VIPA1 back.
2. New connections to VIPA1 are accepted by A.
3. IP packets to old connections are routed via A to B for as long as such connections remain.

Gets workload back to where it rightfully belongs

- New connections using VIPA are handled by primary owner

Non-disruptive to connections established to backup

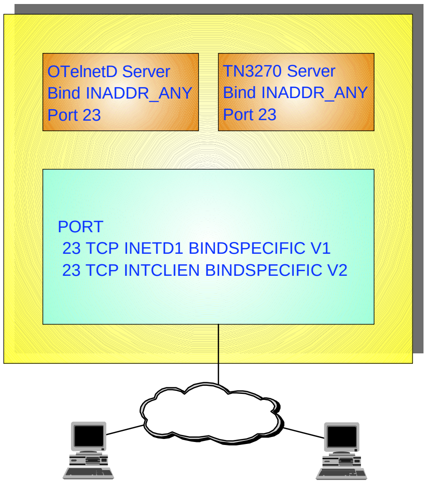
- Connection data forwarded to backup by primary owner
  - Uses internal Sysplex Distributor function
  - No additional configuration externals required
- Can be used in conjunction with Sysplex Distributor so that workload can be distributed to multiple backup servers during primary owner outage
  - Reduce impact of primary owner outage
  - Connection data forwarded to appropriate backup by primary owner
- Allows movement of application server without impacting existing workload
- Data for existing connections continue to be forwarded to old location

When is the DVIPA actually moved?

VIPADEFINE / VIPABACKUP	Initial activation on primary owner	Primary goes down, backup activates	Primary comes back up and tries to activate
Moveable IMMEDIATE	Successful	Successful	Successful - keep active on backup until connections terminate, then deactivate on backup
Moveable WHENIDLE	Successful	Successful	Delay activation until no more connections on the backup

VIPARANGE	Initial activation (not active elsewhere in sysplex) - bind activation or MODDVIPA activation	Application on other stack binds to address that is already active elsewhere in sysplex	IOCTL or MODDVIPA activation of address that is already active elsewhere in the sysplex
Moveable DISRUPTIVE	Successful	New bind fails.	1. Deactivate on current owning stack (break connections) 2. Successful activation on new stack
Moveable NONDISRUPTIVE	Successful	Successful - keep active on old stack until connections terminate	Successful - keep active on old stack until connections terminate

BIND for INADDR_ANY servers



OTelnetD Server
Bind INADDR_ANY
Port 23

TN3270 Server
Bind INADDR_ANY
Port 23

PORT
23 TCP INETD1 BINDSPECIFIC V1
23 TCP INTCLIEN BINDSPECIFIC V2

Telnet to
os390unix.xyz.com
(Resolves to V1).

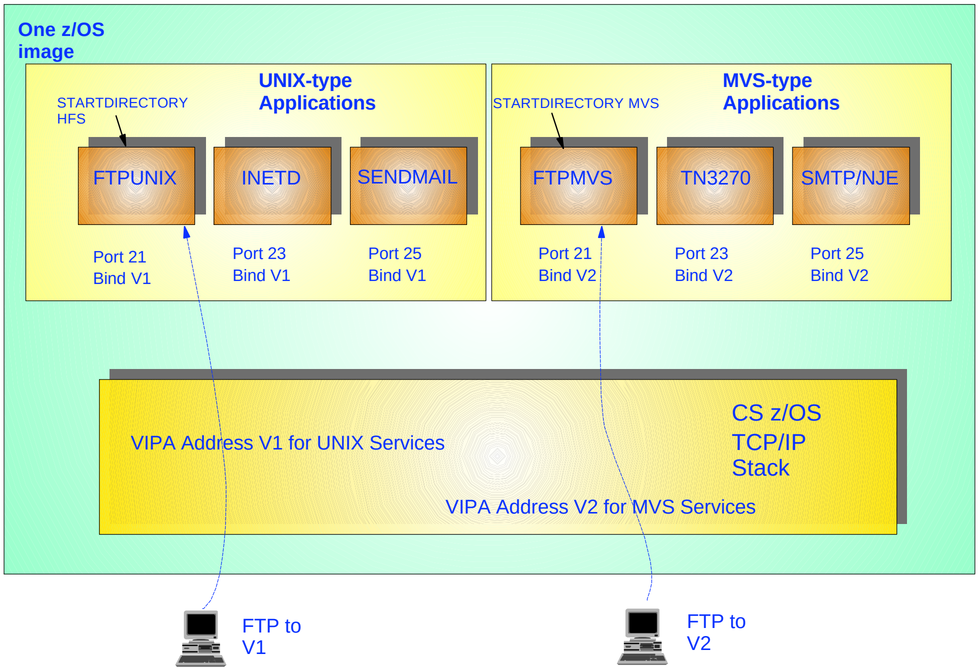
Telnet to
os390mvs.xyz.com.
(Resolves to V2)

Different INADDR_ANY servers, same application, same TCP/IP

- Currently requires different TCP/IP stacks or running some servers on nonstandard port numbers
- Multiple server pair examples
  - TN3270, OTELNET
  - RSH (MVS and UNIX versions)
  - REXEC (MVS and UNIX versions)
- Provide new TCP configuration option on PORT statement
  - Designate different VIPA address to each server
    - Identified by job name
  - Convert BIND(INADDR_ANY,PORT) to BIND(specificIPAddr,PORT)
    - No more conflict or overlap
  - Improved bind-specific dynamic VIPA use

One less reason for running multiple stacks!

Multiple applications bound to same port number on different IP addresses



One z/OS image

UNIX-type Applications

STARTDIRECTORY HFS

FTPUNIX — Port 21 Bind V1

INETD — Port 23 Bind V1

SENDMAIL — Port 25 Bind V1

MVS-type Applications

STARTDIRECTORY MVS

FTPMVS — Port 21 Bind V2

TN3270 — Port 23 Bind V2

SMTP/NJE — Port 25 Bind V2

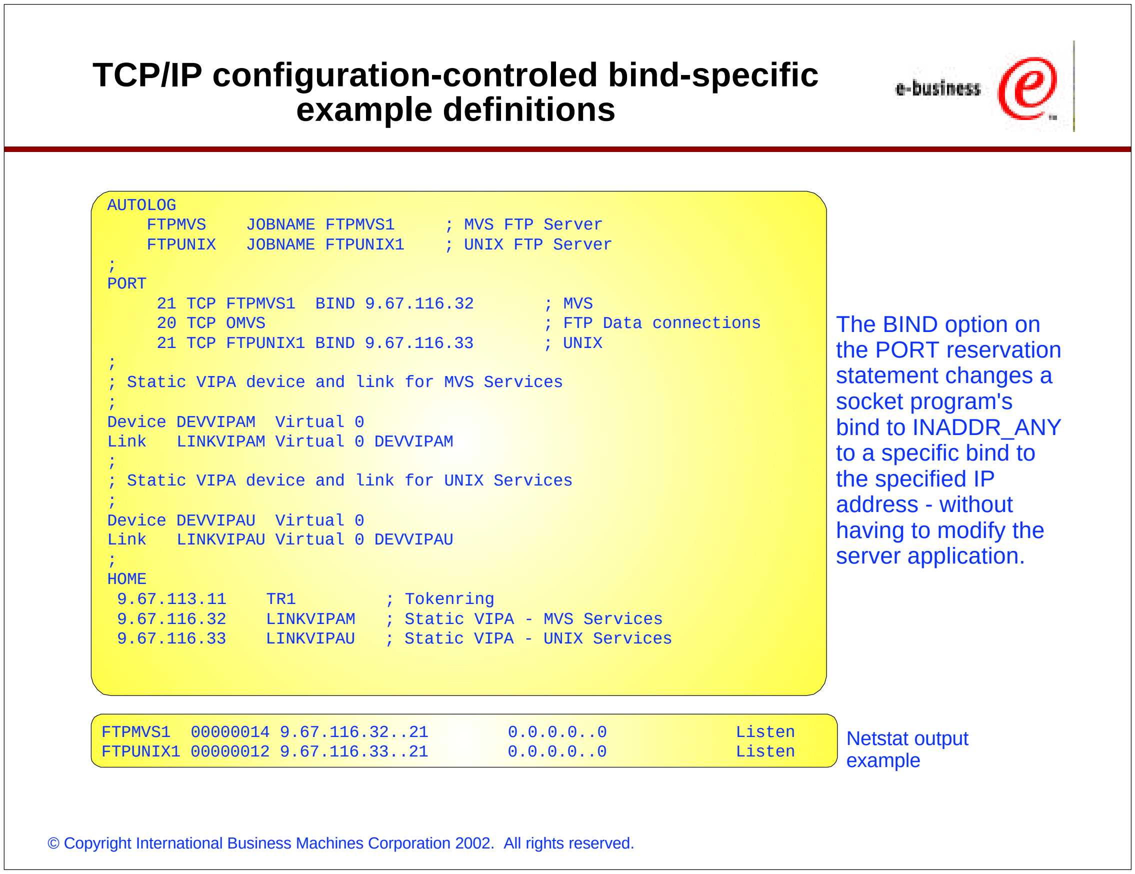
VIPA Address V1 for UNIX Services

CS z/OS TCP/IP Stack

VIPA Address V2 for MVS Services

FTP to V1

FTP to V2



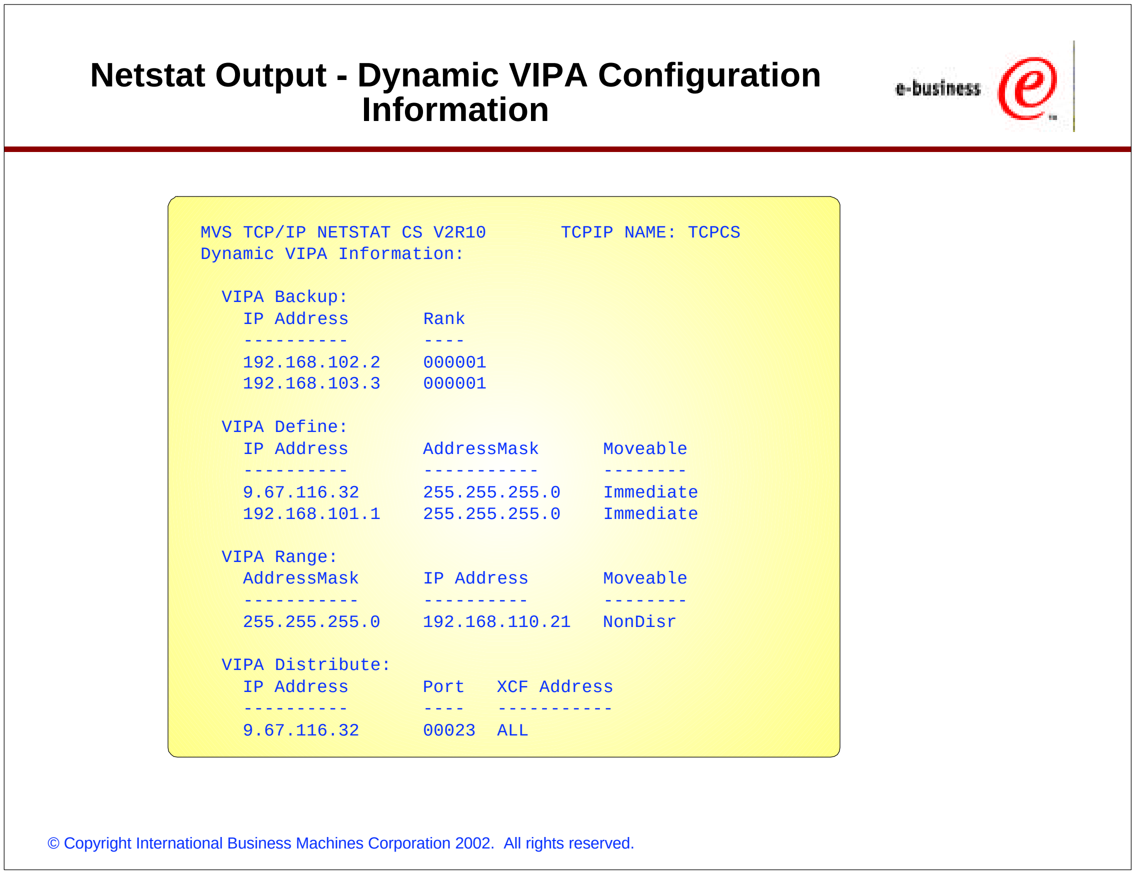
TCP/IP configuration-controled bind-specific example definitions

```
AUTOLOG
    FTPMVS    JOBNAME FTPMVS1     ; MVS FTP Server
    FTPUNIX   JOBNAME FTPUNIX1    ; UNIX FTP Server
;
PORT
     21 TCP FTPMVS1  BIND 9.67.116.32       ; MVS
     20 TCP OMVS                            ; FTP Data connections
     21 TCP FTPUNIX1 BIND 9.67.116.33       ; UNIX
;
; Static VIPA device and link for MVS Services
;
Device DEVVIPAM  Virtual 0
Link   LINKVIPAM Virtual 0 DEVVIPAM
;
; Static VIPA device and link for UNIX Services
;
Device DEVVIPAU  Virtual 0
Link   LINKVIPAU Virtual 0 DEVVIPAU
;
HOME
 9.67.113.11    TR1         ; Tokenring
 9.67.116.32    LINKVIPAM   ; Static VIPA - MVS Services
 9.67.116.33    LINKVIPAU   ; Static VIPA - UNIX Services
```

The BIND option on the PORT reservation statement changes a socket program's bind to INADDR_ANY to a specific bind to the specified IP address - without having to modify the server application.

```
FTPMVS1  00000014 9.67.116.32..21        0.0.0.0..0             Listen
FTPUNIX1 00000012 9.67.116.33..21        0.0.0.0..0             Listen
```

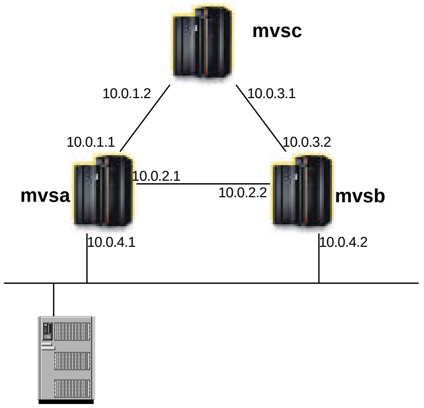
Netstat output example



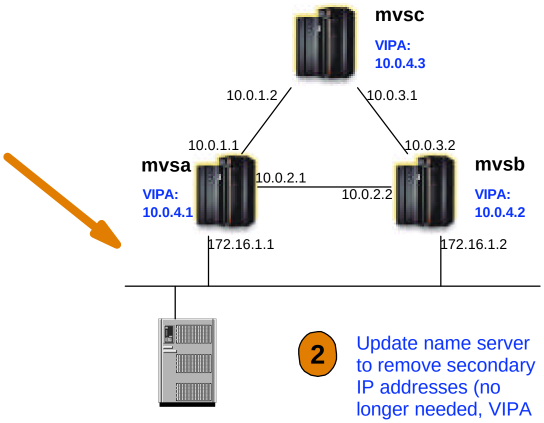
Netstat Output - Dynamic VIPA Configuration Information

```
MVS TCP/IP NETSTAT CS V2R10       TCPIP NAME: TCPCS
Dynamic VIPA Information:

  VIPA Backup:
    IP Address       Rank
    ----------       ----
    192.168.102.2    000001
    192.168.103.3    000001

  VIPA Define:
    IP Address       AddressMask      Moveable
    ----------       -----------      --------
    9.67.116.32      255.255.255.0    Immediate
    192.168.101.1    255.255.255.0    Immediate

  VIPA Range:
    AddressMask      IP Address       Moveable
    -----------      ----------       --------
    255.255.255.0    192.168.110.21   NonDisr

  VIPA Distribute:
    IP Address       Port   XCF Address
    ----------       ----   -----------
    9.67.116.32      00023  ALL
```

A possible approach to start using VIPA addresses



Nameserver:
mvsa: 10.0.4.1, 10.0.1.1, 10.0.2.1
mvsb: 10.0.4.2, 10.0.2.2, 10.0.3.2
mvsc: 10.0.1.2, 10.0.3.1



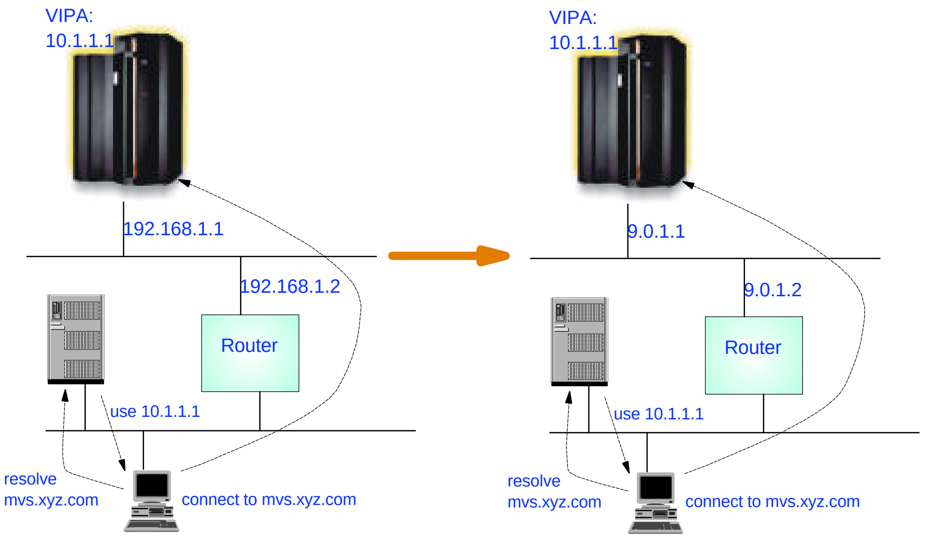
2 Update name server to remove secondary IP addresses (no longer needed, VIPA addresses will always be available if the stack is up).

Nameserver:
mvsa: 10.0.4.1
mvsb: 10.0.4.2
mvsc: 10.0.4.3

1 Promote one or more sets of old physical interface addresses to VIPA addresses, and assign new addresses to the physical interfaces.

Renumbering made easy by VIPA

If you use VIPA addresses on z/OS, then the intermediate network between clients and z/OS up to and including the physical network interfaces on z/OS can be re-numbered without any impact to name servers and/or clients.



The different types of VIPA addresses

Static VIPA

Belongs to one TCP/IP stack. Manual configuration changes needed to move it.

Dynamic VIPA

Stack-managed

- Belongs to one TCP/IP stack, but backup policies governs which TCP/IP stack in the sysplex takes it over if the primary TCP/IP stack leaves the sysplex

Application-specific

- Belongs to an application. Comes active on the TCP/IP stack in the sysplex where the application is started. Moves with the application.

Command-activated

- Belongs to whatever TCP/IP stack in the sysplex on which a command to activate the address has been issued. Moves between TCP/IP stacks based on operator commands.

Distributed

- Used with Sysplex Distributor as a cluster IP address that represents a cluster of equal server instances in the sysplex. From a routing perspective it belongs to one TCP/IP stack. From an application perspective it is distributed among the TCP/IP stacks in the sysplex where an instance of the server application is executing.
- Come to session N37 to hear more about a distributed DVIPA!

Static or dynamic VIPA?

Static VIPA

- Defined during TCP/IP initialization or via OBEY processing (DEVICE, LINK, HOME, BSDROUTINGPARMS)
- Can be used as SOURCEVIPA
- Can be used with Enterprise Extender
- Can be moved to backup stack by executing OBEY commands on the backup stack

Dynamic VIPA

- Defined during TCP/IP initialization or on request (application starting)
- Backup policies can be defined in advance in order to automate movement when owning TCP/IP stack goes/is taken down
- Cannot be used as SOURCEVIPA
- Cannot be used with Enterprise Extender
- Maximum is 64 Dynamic VIPAs per stack
- Can be used for z/OS sysplex workload distribution with Sysplex Distributor in OS/390 V2R10

Recommendations

- Define at least one static VIPA address to be used as SOURCEVIPA address.
- Group applications that normally run together and assign a Dynamic VIPA address to the group. This will allow you to move the group of applications between z/OS images in the sysplex.
- Don't define too many VIPA addresses; it may become difficult to administer and there may be a performance hit. Generally somewhere between 2 and 10 VIPA addresses per z/OS image seems to be sufficient in most configurations.
- If you need to support UNIX and non-UNIX servers, define two static VIPA addresses, use the BIND-specific support to force servers to bind to them, and update your name server with appropriate DNS names for the two addresses.
- If you use Enterprise Extender, define a separate static VIPA address for EE and point to the address in your VTAM start options on the IPADDR keyword.
- Allthough it is not an absolute requirement, it is always recommended to use a dynamic routing update protocol when you use VIPA addresses. In most configurations, interface resiliance and host-independence will not work if you do not use dynamic routing on z/OS.

For More Information....

URL	Content
http://www.ibm.com/servers/eserver/zseries	IBM eServer zSeries Mainframe Servers
http://www.ibm.com/servers/eserver/zseries/networking	Networking: IBM zSeries Servers
http://www.ibm.com/servers/eserver/zseries/networking/technology.html	IBM Enterprise Servers: Networking Technologies
http://www.ibm.com/software/network	Networking & communications software
http://www.ibm.com/software/network/commserver	Communications Server
http://www.ibm.com/software/network/commserver/library	Communications Server white papers, product documentations, etc.
http://www.redbooks.ibm.com	ITSO redbooks
http://www.ibm.com/software/network/commserver/support	Communications Server technical Support
http://www.ibm.com/support/techdocs/	Advanced technical support (flashes, presentations, white papers, etc.)
http://www.rfc-editor.org/rfcsearch.html	Request For Comments (RFC)