

# How is the z/VM and Linux Architecture Holding Up?

David Kreuter  
MVMUA  
April 2015  
Jersey City, NJ.

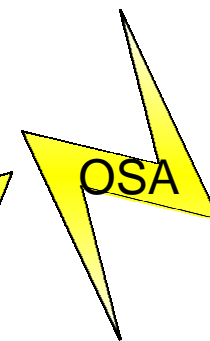
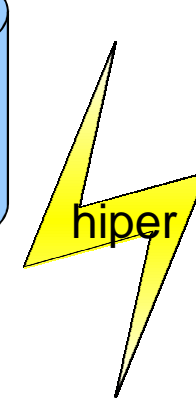
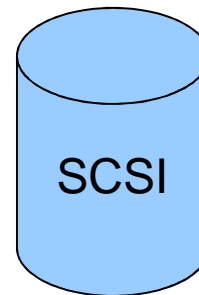


# Now Showing: How is the z/VM and Linux architecture holding up?

Abstract: Update on how my architecture for zvm620 is working in on going z/VM630 situations. The highlights of the architecture will be discussed along with the real shop use of the evolving architecture. Deployment and operational strategies will be discussed along with lessons learned.

# Presentation Goals

- Architecture revisited with real life client situations
- Evolution
- Achievements

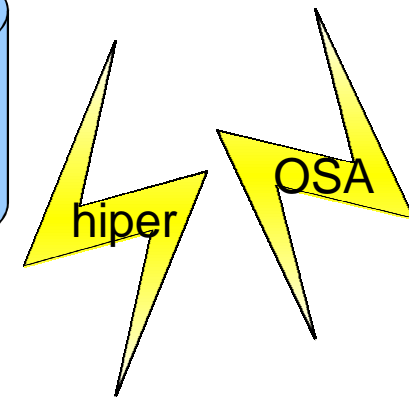
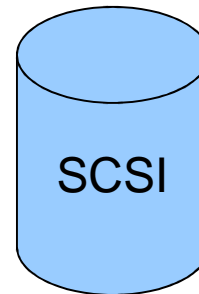


# Architecture Revisited

Client profiled is a large government enterprise.

Other shops mentioned too:

- Small shop (software vendor)
- Burgeoning (academic environment)
- FCP Only site
- Wannabe architecture sites



# Service Zone Evolution

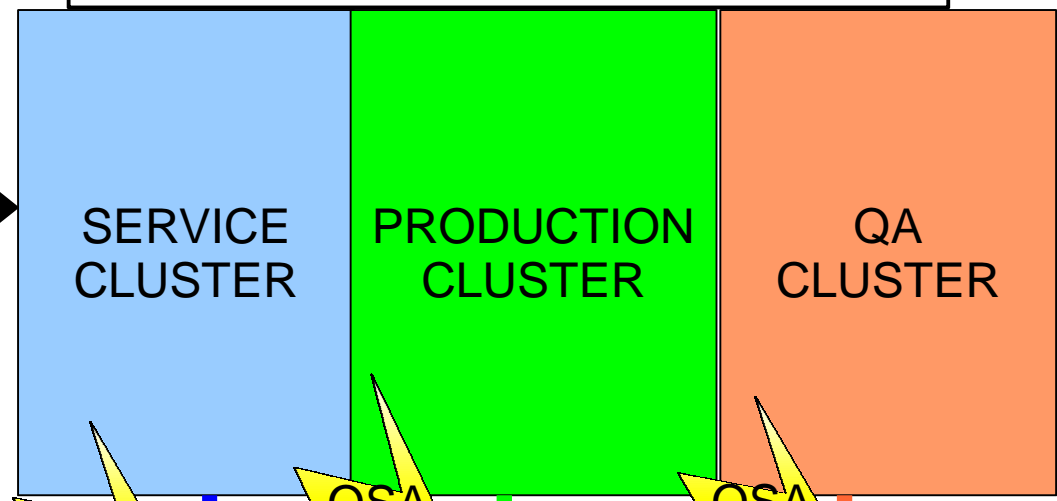
- Second level VM systems are a starting point.
- Mixed with a Systems Programming test system
  - clearly not a sandbox though
- Advantages over 2<sup>nd</sup> level:
  - Proving ground for real hardware
  - Uses LPAR controls same as other LPARs
- While second level still provides:
  - Better flexibility
  - Ease of use
  - On the fly tailoring

# Service Zone Requirements

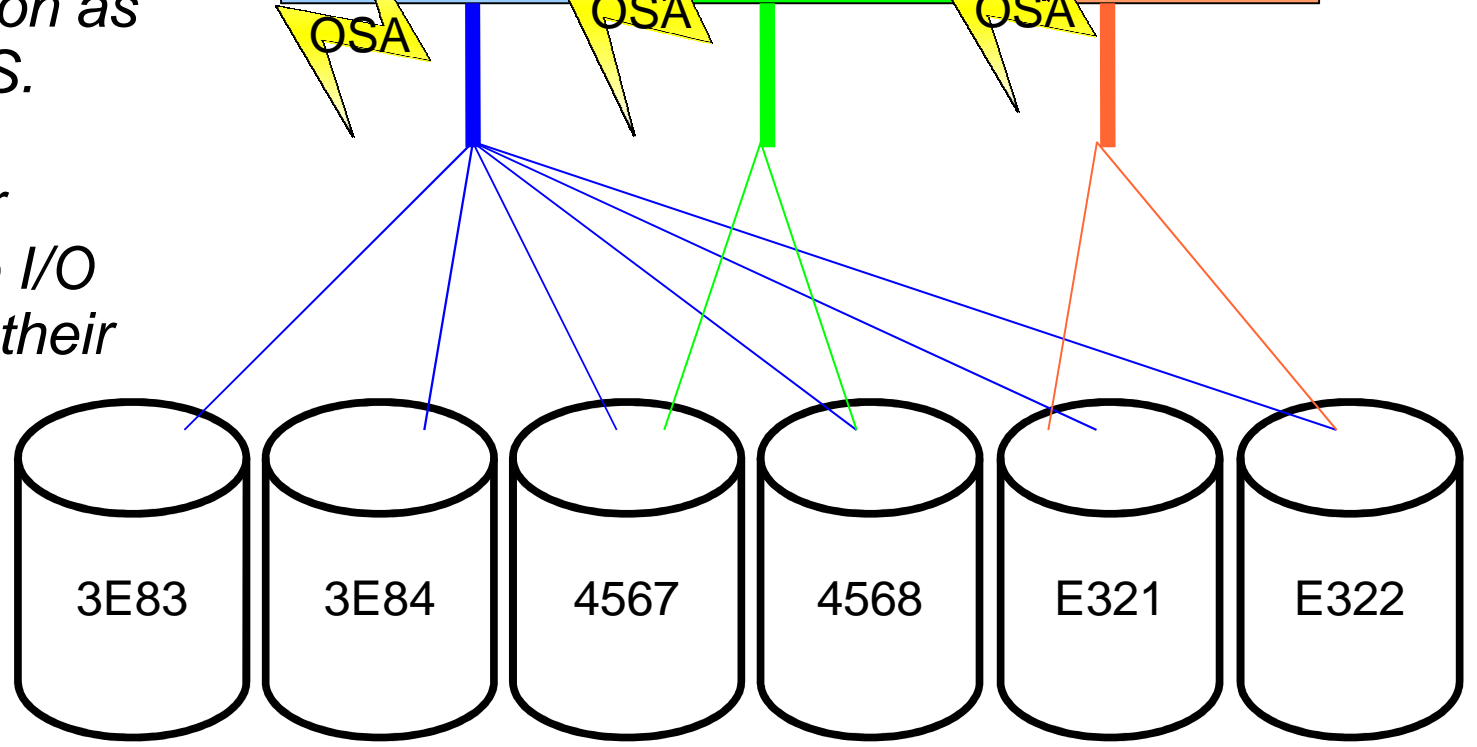
- In 620 and beyond two services zones form a service cluster.
  - Second member is cold standby. Not quite ready for prime time take over.
- Connectivity to all other LPARs over hipersocket.
- Must be able to issue remote commands.
- Small memory and IFL footprint.
- Sees entire I/O configuration.
- TN3270 client connectivity from the enterprise.



# HIPERSOCKET

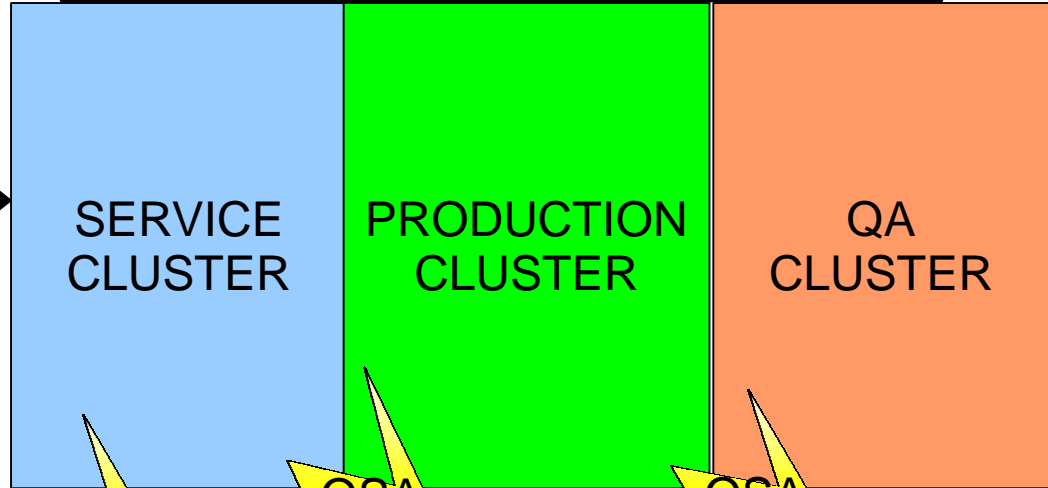


*Service LPAR sees the entire I/O configuration as defined in the IOCDs. Great for systems management. Other clusters only see the I/O devices required for their use.*

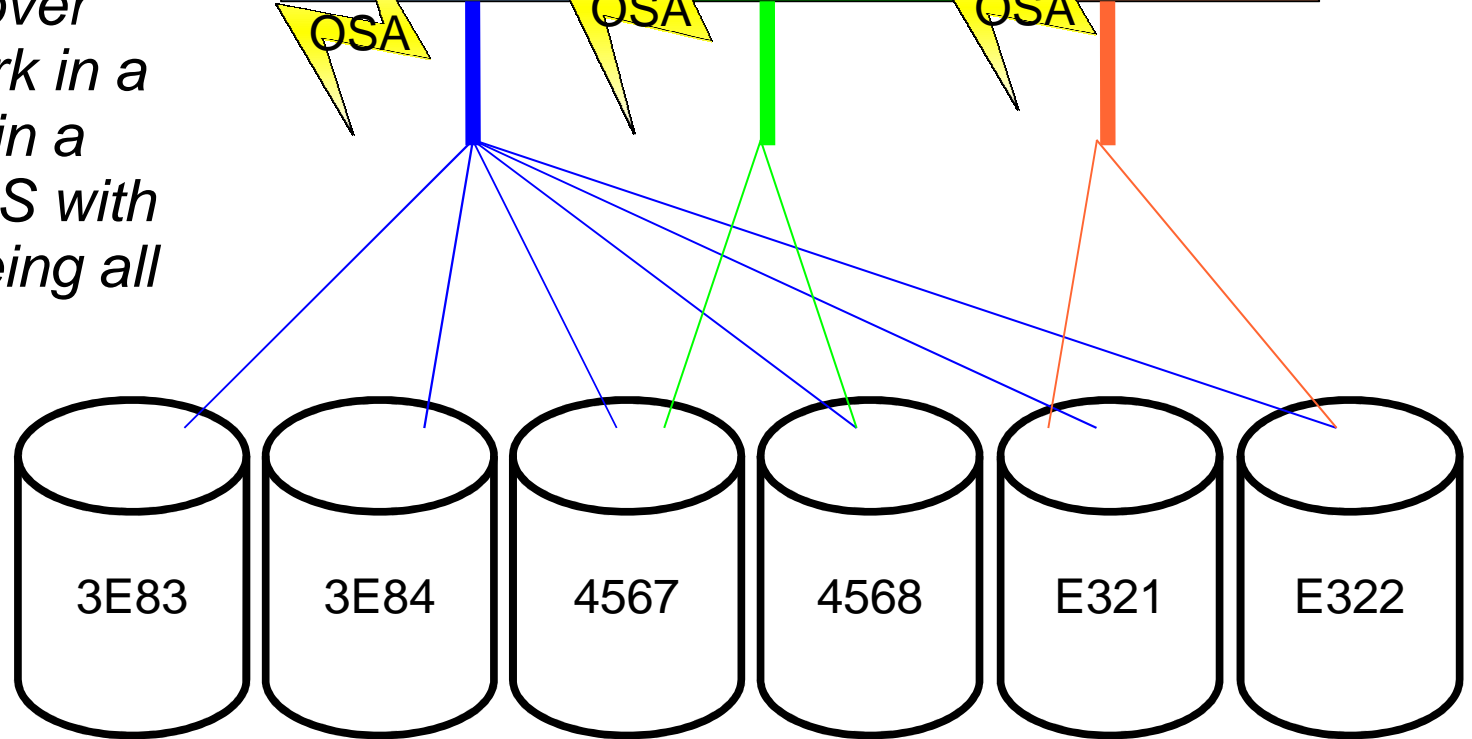




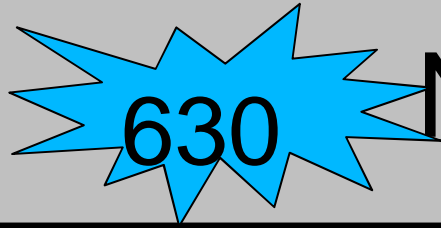
# HIPERSOCKET



*The service zone connects to the other LPARs over hipersockets (network in a box). All LPARs within a CEC share an IOCCDS with the service zone seeing all devices.*



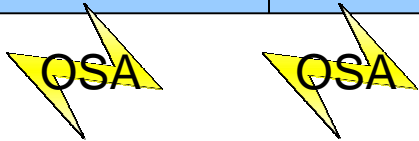
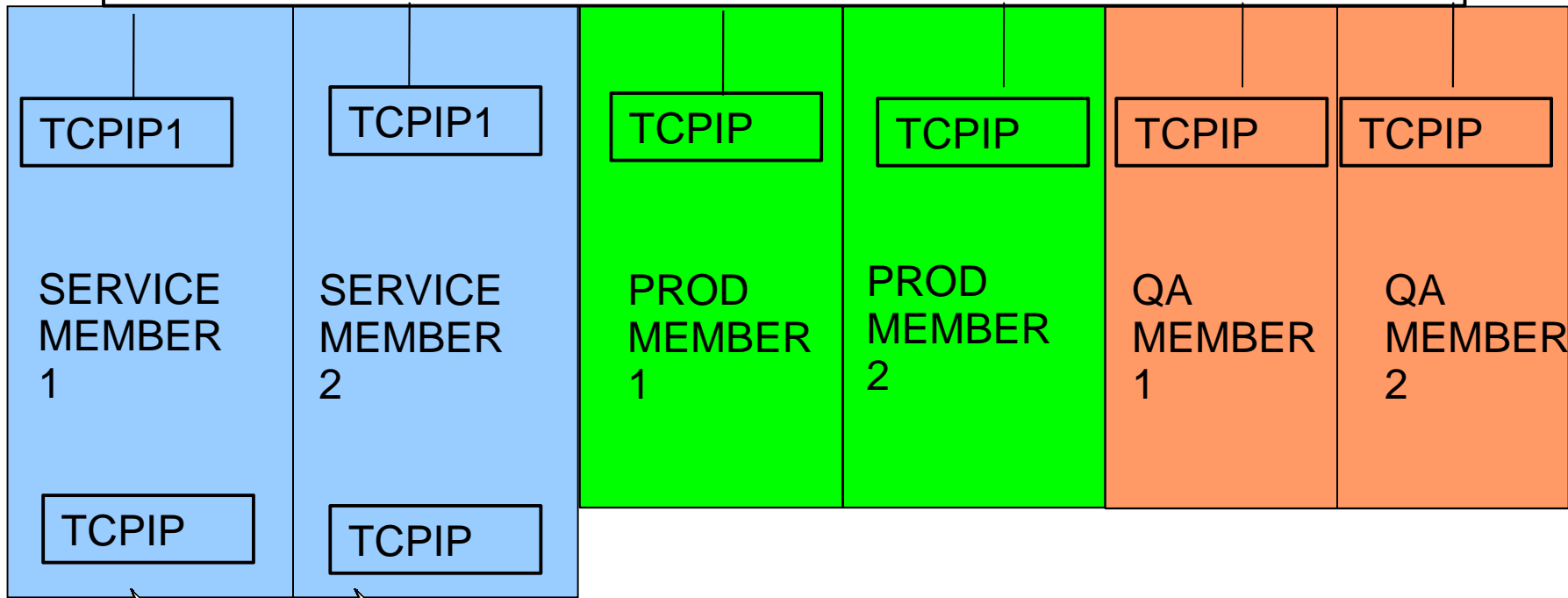




# 630 Network Evolution

- Hipersocket network connects all LPARs in the CEC.
  - 2 sites with one CEC each.
- Additional CEC added in each site
  - 2 sites with 2 CECs
  - Each CEC with its own hipersocket network
- Using VSWITCH with hipersocket bridge
- 2 sites with 2 CECs
- Each site has one hipersocket network!

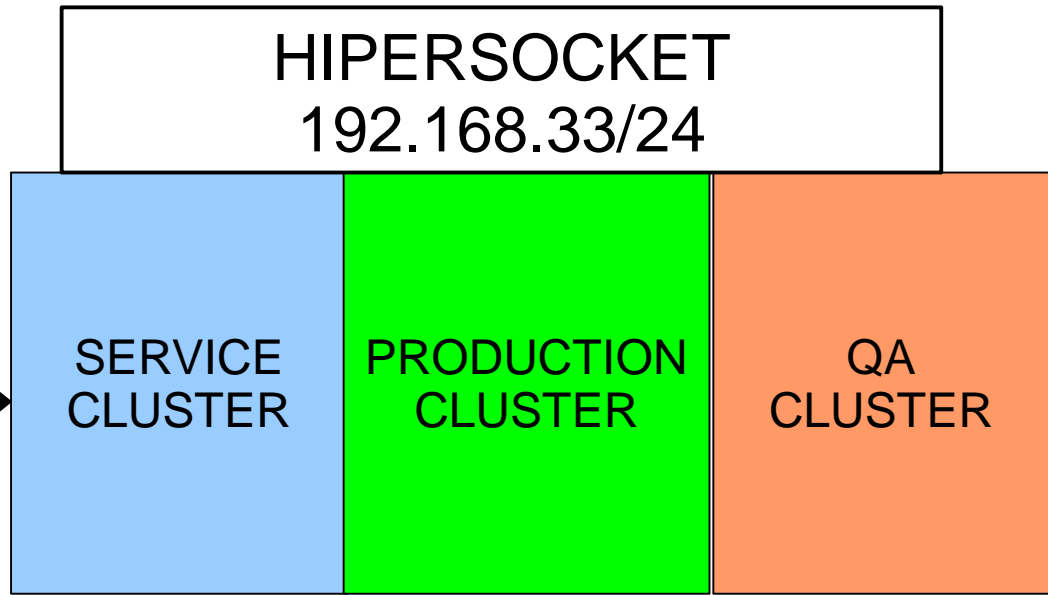
# HIPER SOCKET 192.168.33/24



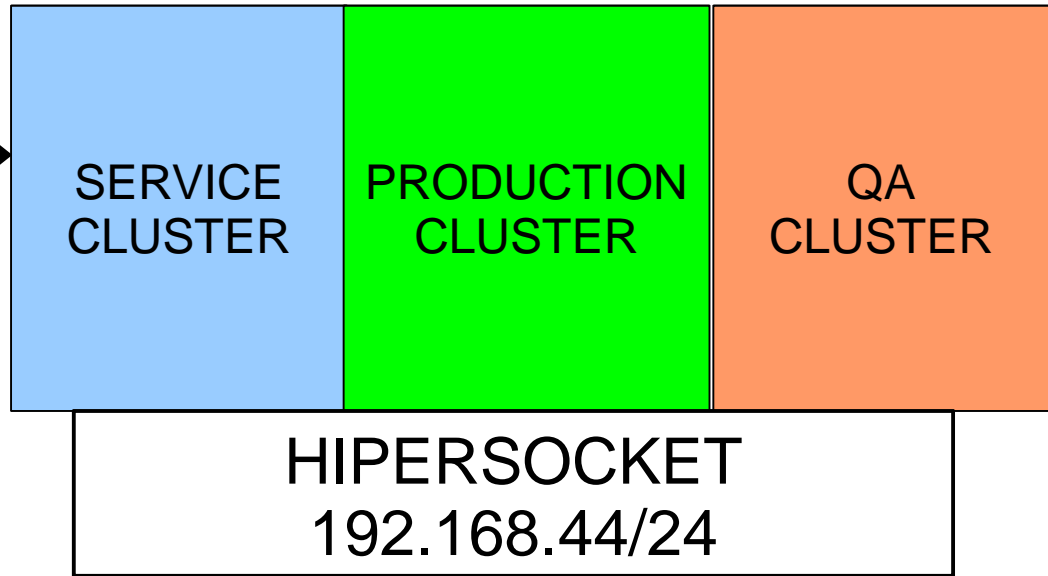
*VM "Network in a Box" on hipersocket. Service zones all have a stack on OSA for enterprise connectivity. TN3270 using OSA only into the service zone. VMLINK to TCPIP1 on hipersocket to use CMS TN3270 client to logon onto other LPARs. Note: these networks do not route to each other.*



630



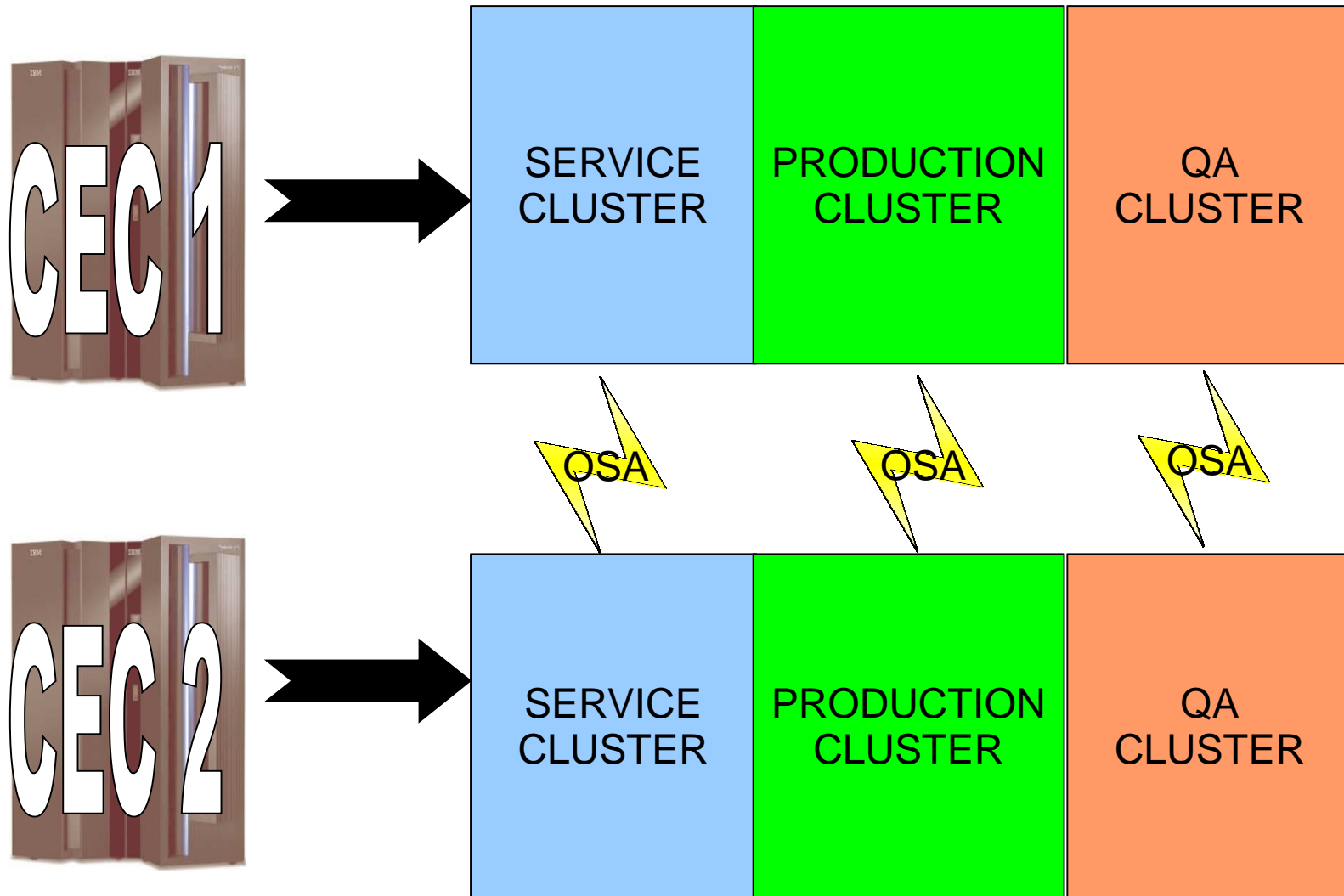
CECs in same site with "net in a box" hipersockets.



# Using VSWITCH with hipersocket bridge all nodes are on same network!

630

192.168.34/24



# VSWITCH with HIPER bridge as defined in SYSTEM CONFIG structure

630

(so it is created at IPL time)

```
VZ4SRV01: VMLAN MACPREFIX 020F01  
VZ4SRV02: VMLAN MACPREFIX 020B02
```

```
DEFINE VSWITCH VSWSRVTR,  
TYPE QDIO ETHERNET,  
UPLINK RDEV 920 930,  
BRIDGEPORT RDEV C40,  
VLAN 440 NATIVE 99
```

PMAINT  
CF0

Must assign MACID prefix so you can have unique MAC addresses. Used the LPAR number as the second byte.

# QUERY VSWITCH VSWRVTR

```
VSWITCH SYSTEM VSWSRVTR Type: QDIO      Connected: 1      Maxconn: INFINITE
PERSISTENT RESTRICTED ETHERNET          Accounting: OFF
USERBASED
VLAN Aware Default VLAN: 0440      Default Porttype: Access GVRP: Enabled
Native VLAN: 0099      VLAN Counters: OFF
MAC address: 02-0B-02-00-00-03      MAC Protection: Unspecified
IPTimeout: 5      QueueStorage: 8
Isolation Status: OFF      VEPA Status: OFF
Uplink Port:
State: Ready
PMTUD setting: EXTERNAL      PMTUD value: 8992
RDEV: 0920.P00 VDEV: 0606 Controller: DTCVSW2 ACTIVE ←
EQID: SRVOSA04
RDEV: 0930.P00 VDEV: 0606 Controller: DTCVSW1 BACKUP ←
EQID: SRVOSA04
Bridge Port:
Role: Primary      Status: Active      Active LPAR: VZ4SRV02
State: Ready
RDEV: 0C40      VDEV: 0609 Controller: DTCVSW2
```

# SMSG RSCS QUERY LINK \* SHOW PARM NAME

Link

Name	Parm Text
VZ4SRV01	HOST=192.168.44.10
VZ4TDC01	HOST=192.168.44.11
VZ4TDC02	HOST=192.168.44.111
VM200P	HOST=192.168.44.15
VM200Q	HOST=192.168.44.115
VZ4GPN01	HOST=192.168.44.12
VZ4GPN02	HOST=192.168.44.112
VZ4PRD01	HOST=192.168.44.13
VZ4PRD02	HOST=192.168.44.113
VZ4GOP01	HOST=192.168.44.14
VZ4GOP02	HOST=192.168.44.114
VZ4PBR01	HOST=192.168.44.17
VZ4PBR02	HOST=192.168.44.117
*NOTHERE	MSGFILE=NOTHERE PURGE=2
*UNKNOWN	MSGFILE=UNKNOWN PURGE=2

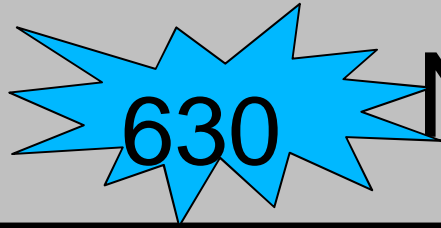
15 links found

630

Two CECs.

One network.

14 VM  
systems (!2  
LPARs and 2  
second level  
systems).

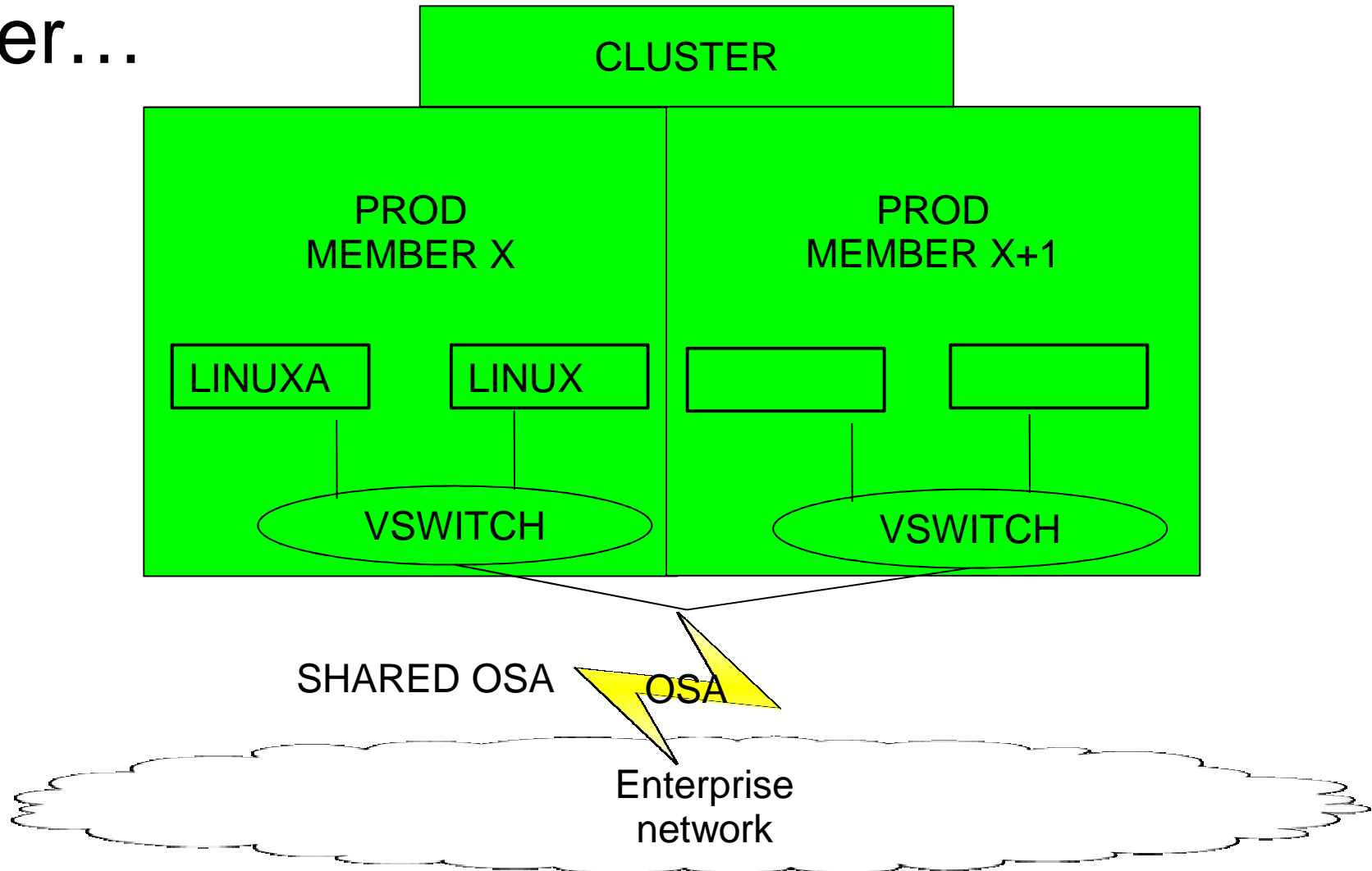


# 630 Network Evolution

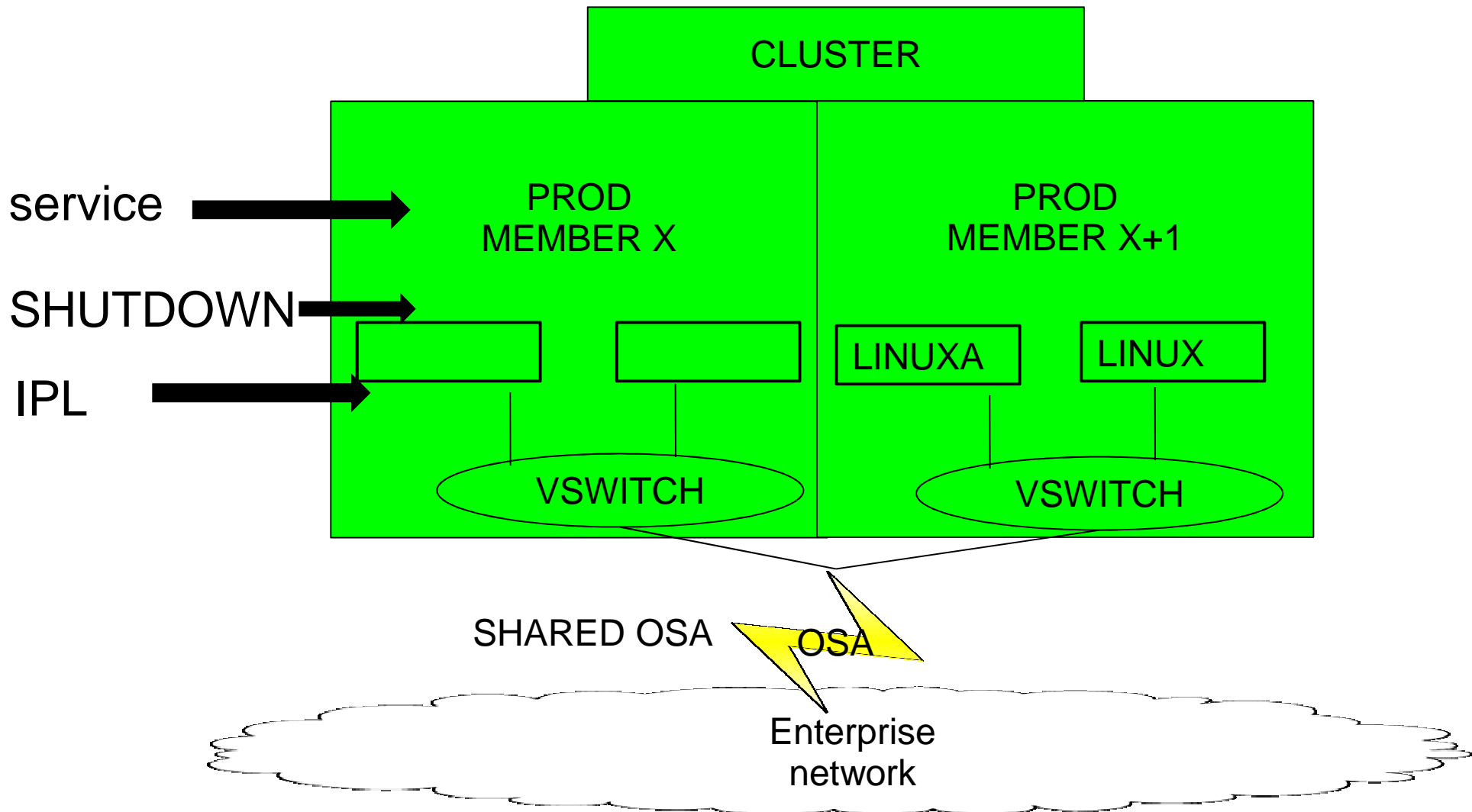
- Hipersocket network connects all LPARs in the CEC.
  - 2 sites with one CEC each.
- Additional CEC added in each site
  - 2 sites with 2 CECs
  - Each CEC with its own hipersocket network
- Using VSWITCH with hipersocket bridge
- 2 sites with 2 CECs
- Each site has one hipersocket network!



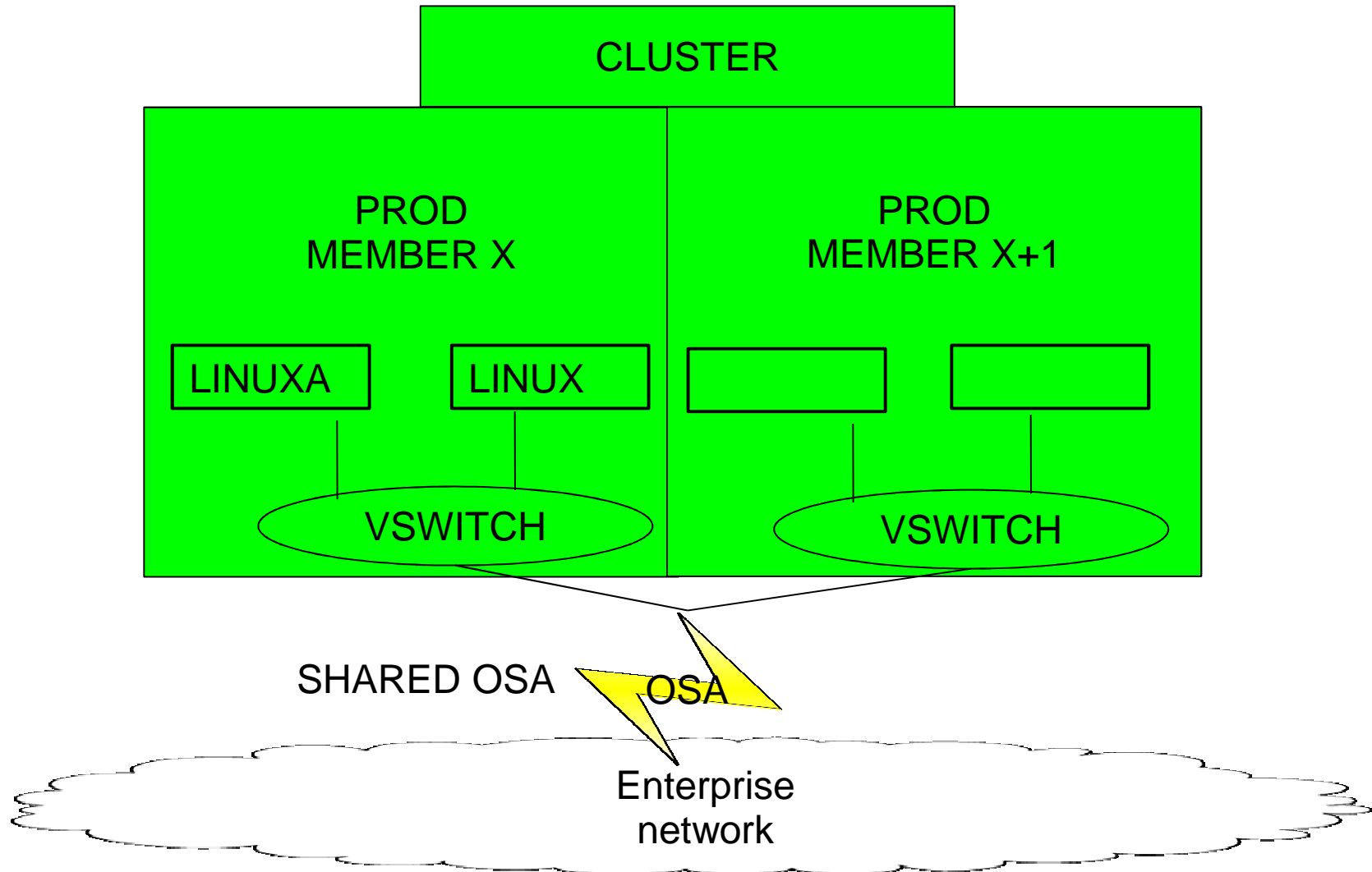
Lesson learned: With 620 LGR Linux workload can be moved from one member to another...



....MEMBER X can be taken of the air without application impact



# Workload can be moved back to original member



# Real Life Experiences

- Customer received four new z13's December 2014 replacing four z12s.
  - Due to contract issues the z12s had to be off the floor by January 31<sup>st</sup>, 2015.
  - Usually would unfold and progress over months not weeks.
- No Linux outages with LGR strategy.
- HCD on z/VM
  - One of the z13s VM only.

# December 2014



ANY Cluster Member 1



ANY Cluster Member 2



# January 2015



ANY Cluster Member 1



ANY Cluster Member 2



# 0. Starting point



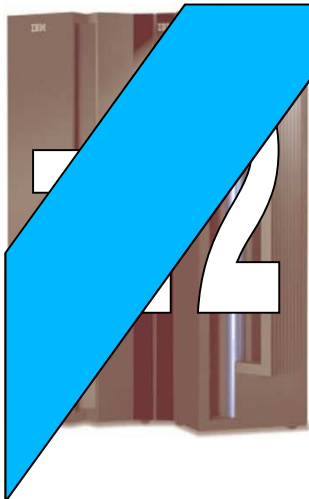
ANY Cluster Member 1



ANY Cluster Member 2



# 1. LGR all workload



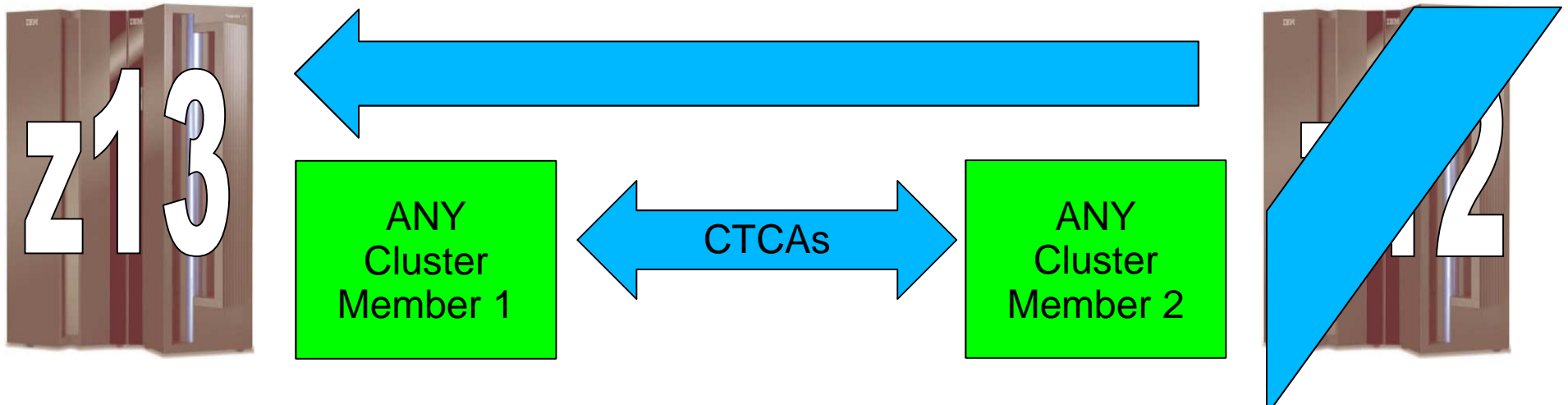
ANY Cluster Member 1



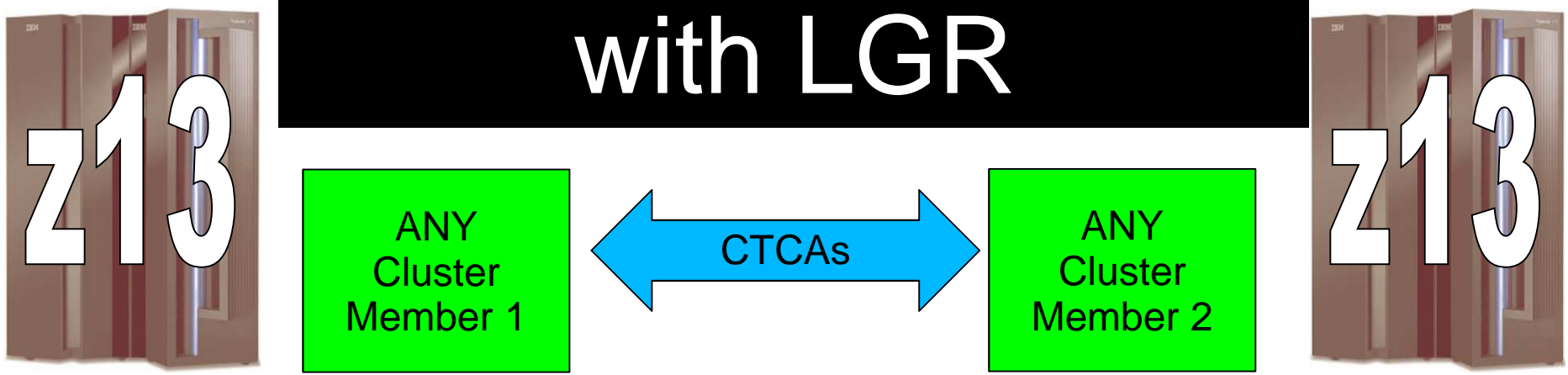
ANY Cluster Member 2



## 2. LGR to z13

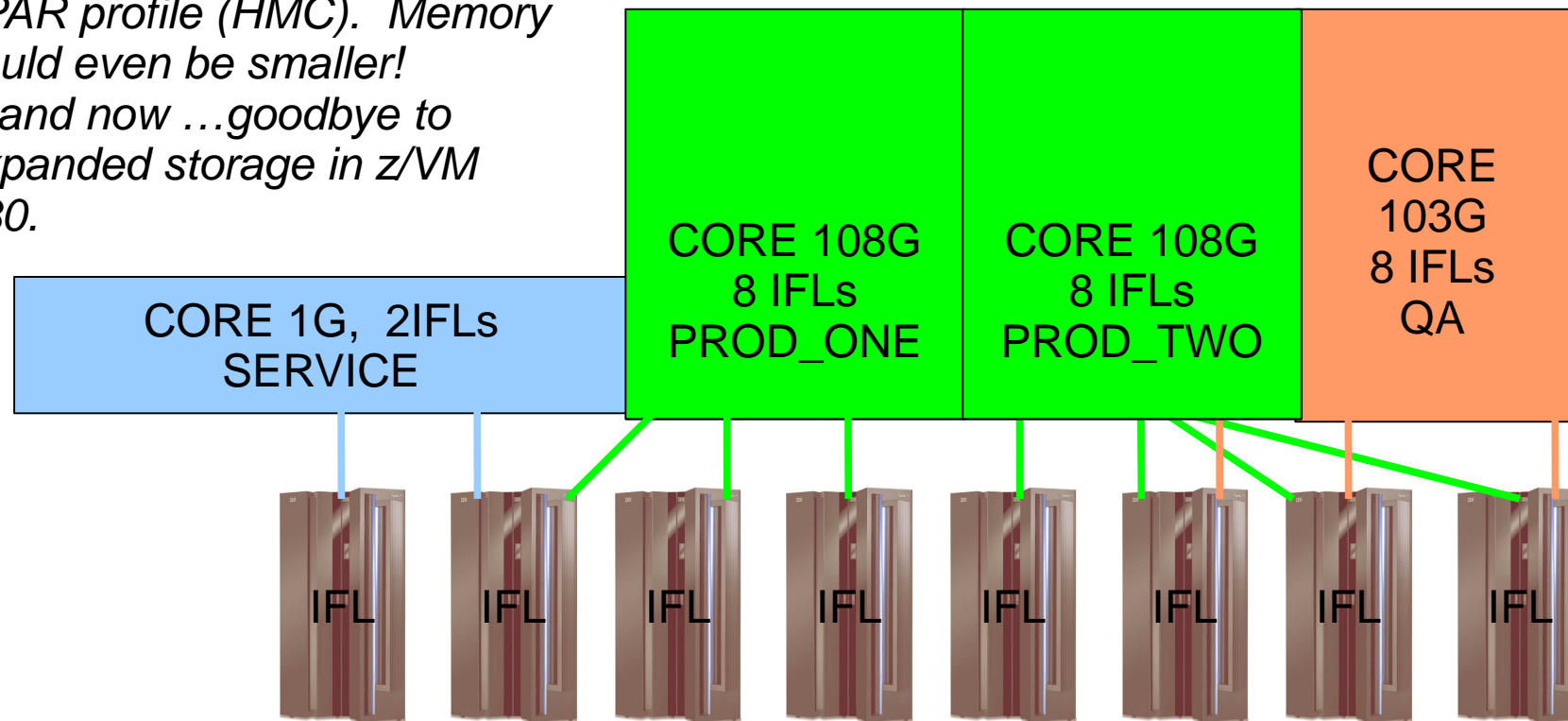


## 3. Balance workload with LGR



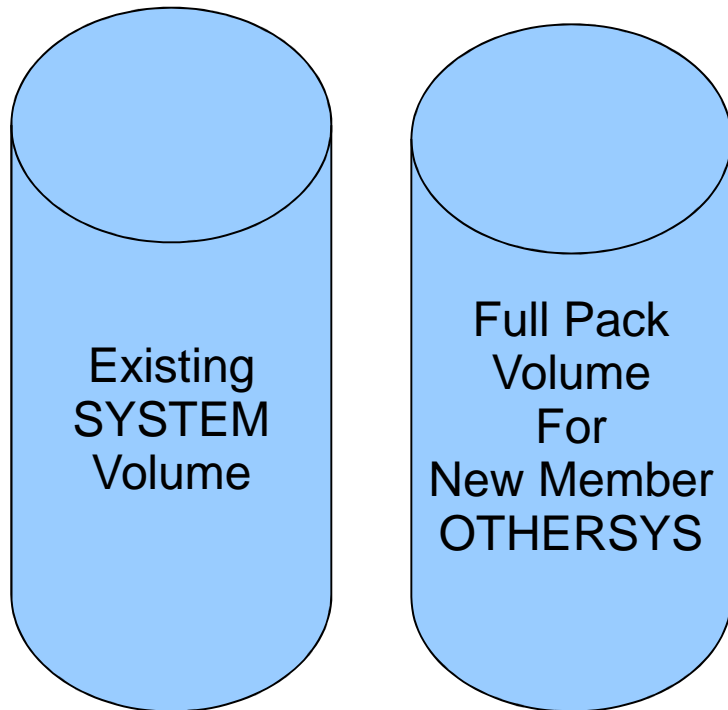
# Memory and IFL Definition at New Shop

*Do not need many resources to be great! The two IFLs for the service zone should also be given a low weight in the LPAR profile (HMC). Memory could even be smaller!  
...and now ...goodbye to expanded storage in z/VM 630.*





# Want to Explain Minidisk Mapping



Both of these volumes contain minidisks used by their respective systems. Tools will map these minidisks to be used on the service system via directory definition or DEFINE MDISK.  
Brilliant for tailoring and copying!

# Brilliant for Tailoring and Copying!

USER DIRECT 1<sup>st</sup> Level:

```
USER PMAINT WD5JU8P 1G 1G G
```

```
MDISK CF0 3390 1 120 VMVOL1
```

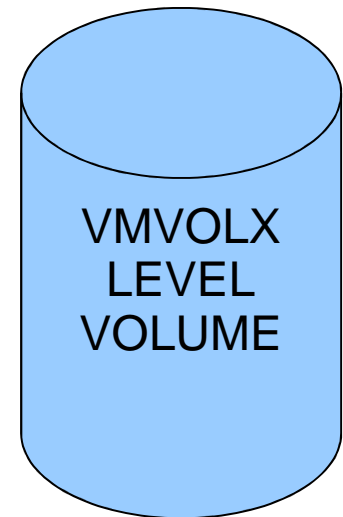
```
:
```

```
USER OTHERLPR NOLOG
```

```
MDISK 0000 3390 1 120 VMVOLX MR OTHERSYS PMAINT CF0
```

*Use the read and  
write link password  
to show owning  
machine and  
address*

**PMAINT CF0 starts on  
cylinder 1 for 120 cylinders**

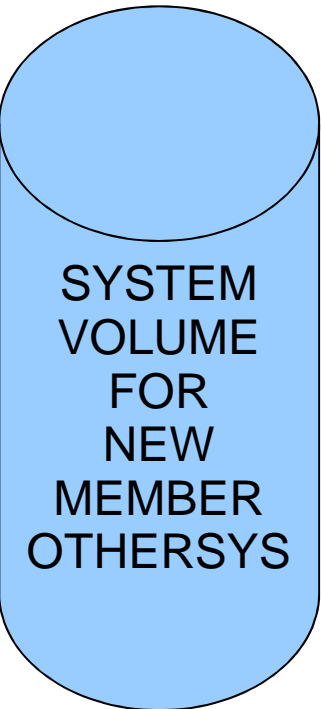


# Brilliant for Tailoring and Copying: DEFINE MDISK COMMAND

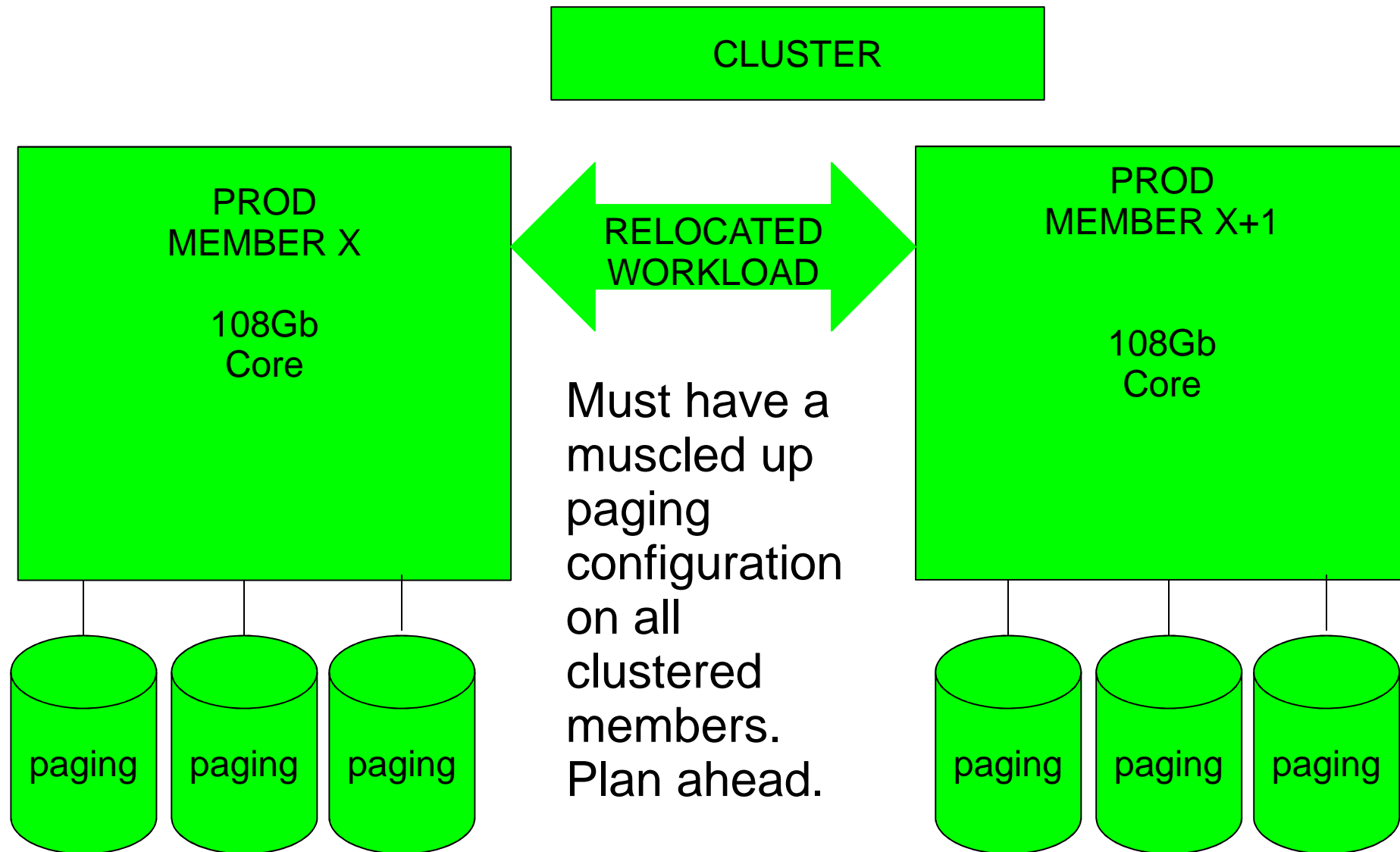
FROM MAINT on 1<sup>st</sup> level:

```
CP DEFINE MDISK FCF0 1 120 VMVOLX  
ACCESS FCF0 J
```

PMAINT CFO on other system  
SYSTEM CONFIG .....



# Must have a Robust Paging Farm



# Best Practices for Steady State

- Steady state
- Adhere to architecture
- Standards
- Get to know:
  - Network
  - Storage devices
- Observer
- Monitor
- Detect
- Report
- Automate



630

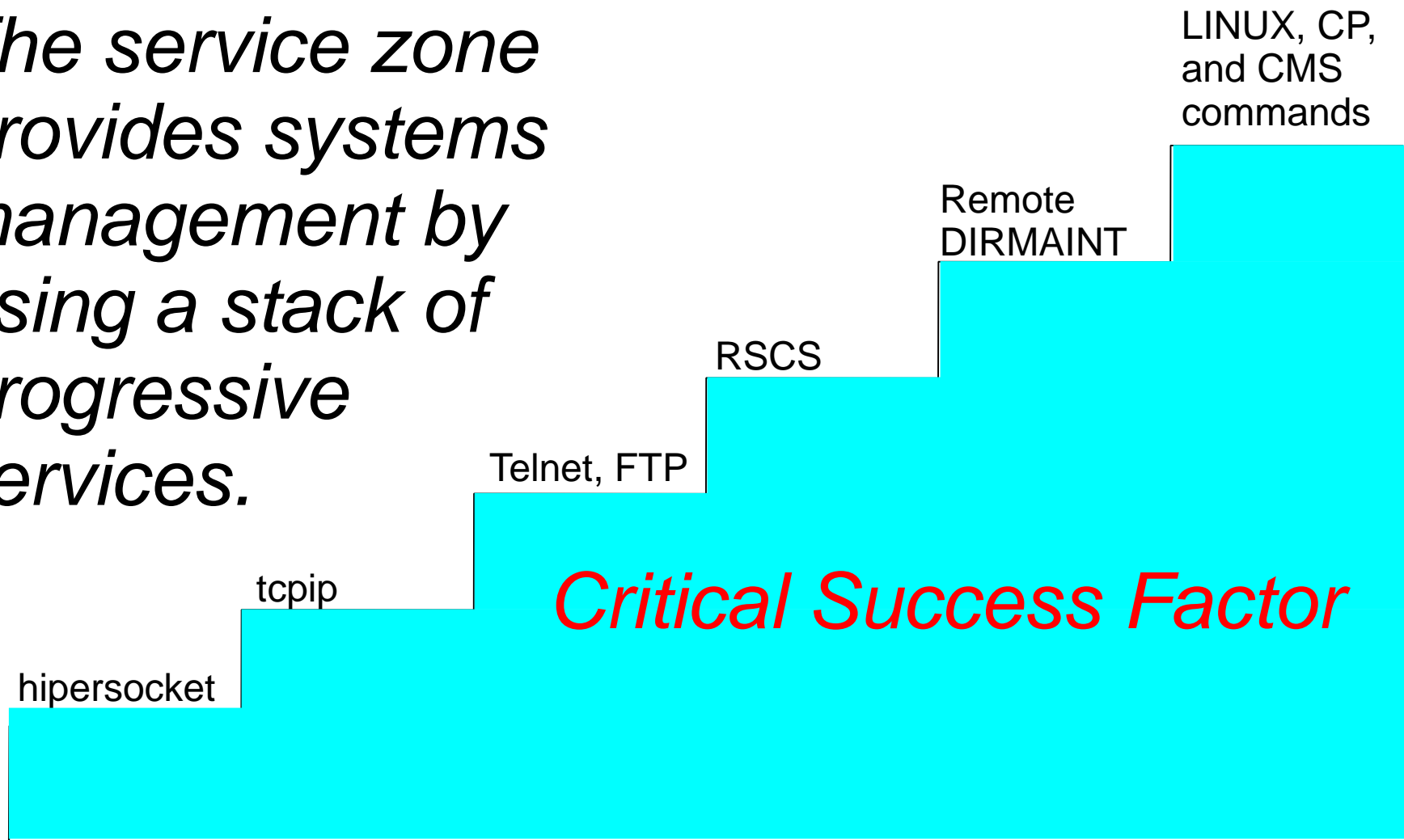
# Lessons Learned

- Changing local tools on 24 LPARs is time consuming!
- Get used to EQIDs for LGR
- Let VSWITCH handle VLAN unless Linux really needs to manage VLANs
- Monitor your systems
- Cleanup your mess
- Go with common core base
- Keep performance data lest you be blamed for every problem



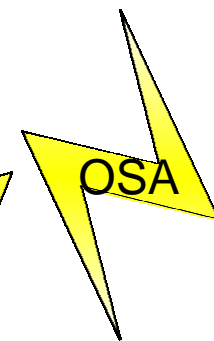
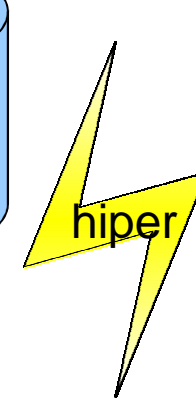
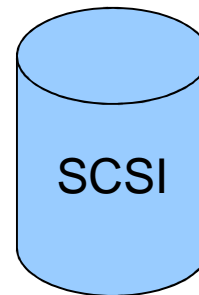
# Network Services Stacking

*The service zone provides systems management by using a stack of progressive services.*



# Presentation Goals

- Architecture revisited with real life client situations
- Evolution
- Achievements



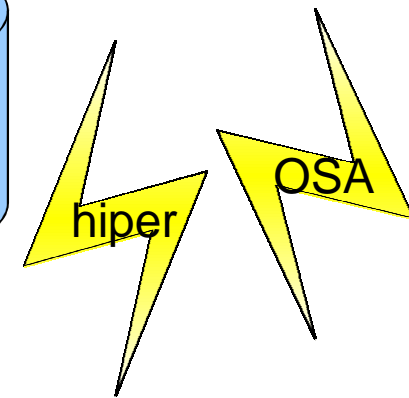
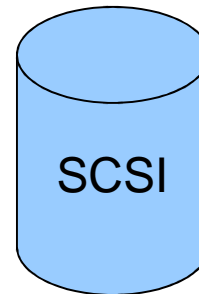
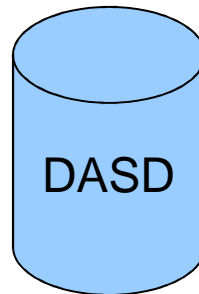


# Architecture Revisited

Client profiled is a large government enterprise.

Other shops mentioned too:

- Small shop (software vendor)
- Burgeoning (academic environment)
- FCP Only site
- Wannabe architecture sites



# The Future

- ...is now! It is upon us!
- Clients with large memory.
- Will there be a world without 3390s?
- 10Gb OSA exploitation.
- Vertical vs. Horizontal CPUs
- Hiper dispatch



# The Mysterious World of FCP on z/VM

- Departure from 3390 technology
- z/VM and Linux use of FCP
- Meet your friendly storage administrator
- The voodoo of FCP terminology
- Experiences with FCP-only VM customer
- The future is now
- How I learned to accept my fate and love modern storage
- EDEV for opsys data
- DEDICATED FCP for payload data

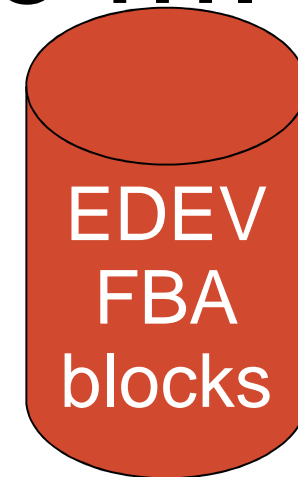
FBA  
EDEV  
WWPN  
NPIV  
WWID  
Multipath

# Departure from 3390 technology



From “Round and Brown and spins around” to ....

Simulated FBA....



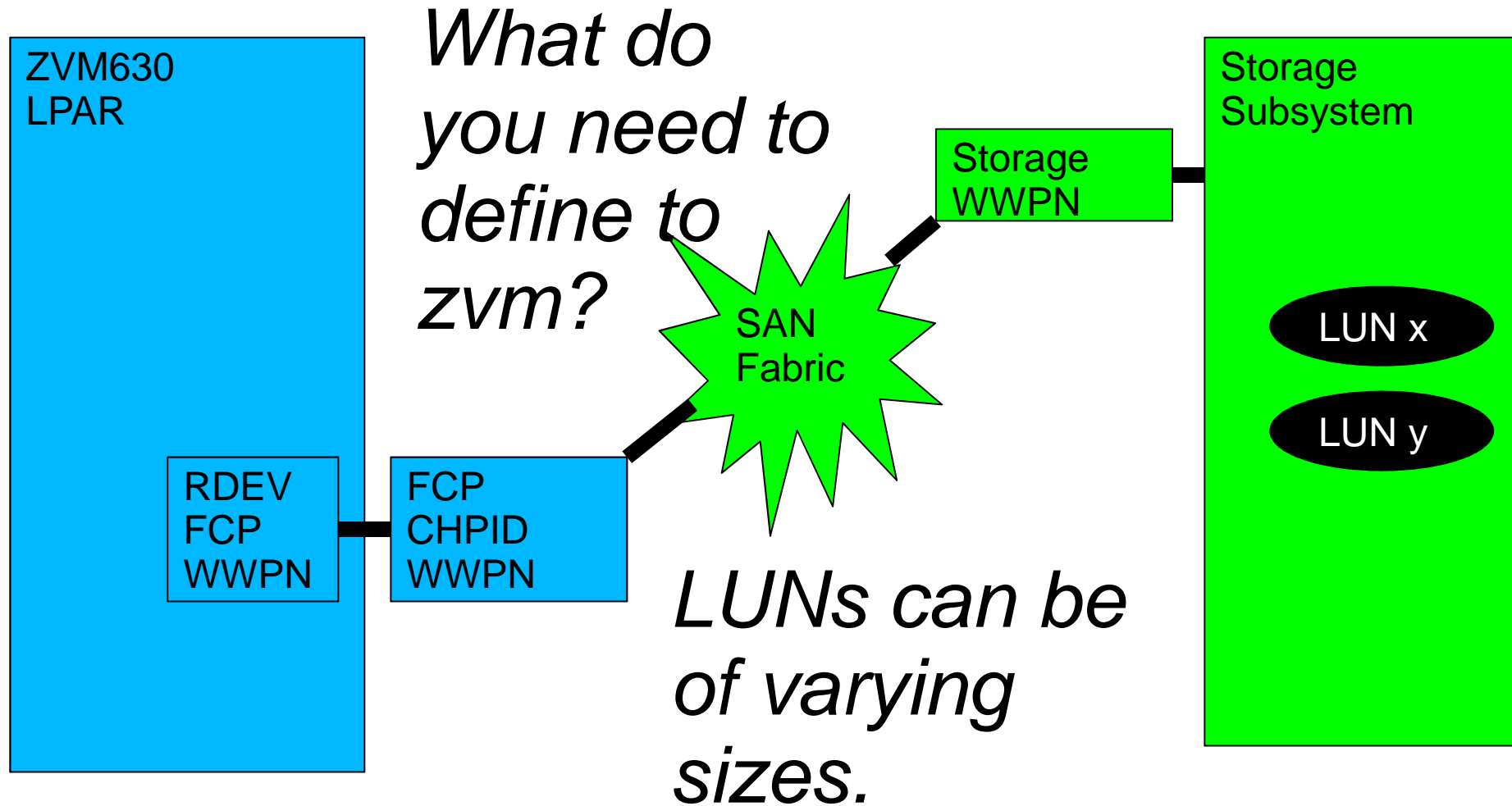
LINUX with  
DEDICATED  
FCP

To DEDICATED FCP...

# The voodoo of FCP terminology

- FCP – Fibre Channel Protocol. A technology to connect data storage devices, usually SCSI.
- SCSI – Small Computer System Interface. Set of standards for connecting peripherals (we care about disk and tape) to a computer host.
- NODE – endpoint that can be a computer, host, disk, etc. It has a 64-bit address.
- WWPN – World Wide Port Name. It is assigned to a port.
  - 64-bit unique address assigned by the vendor and is persistent.
- NPIV – N\_Port ID Virtualization. Fibre Channel method to allow sharing of one physical node WWPN as if it were multiple virtual ports. Solves major data exposure problem in Z series.
- FBA – old school Fixed Block Architecture for disk addressing.
- WWID – World Wide Identifier. Another Fibre Channel implementation that is used in several storage technologies including multipathing in Linux. It is persistent.
- EDEV – Emulated disk device in z/VM that supports z/VM I/O to FCP as if it were FBA.

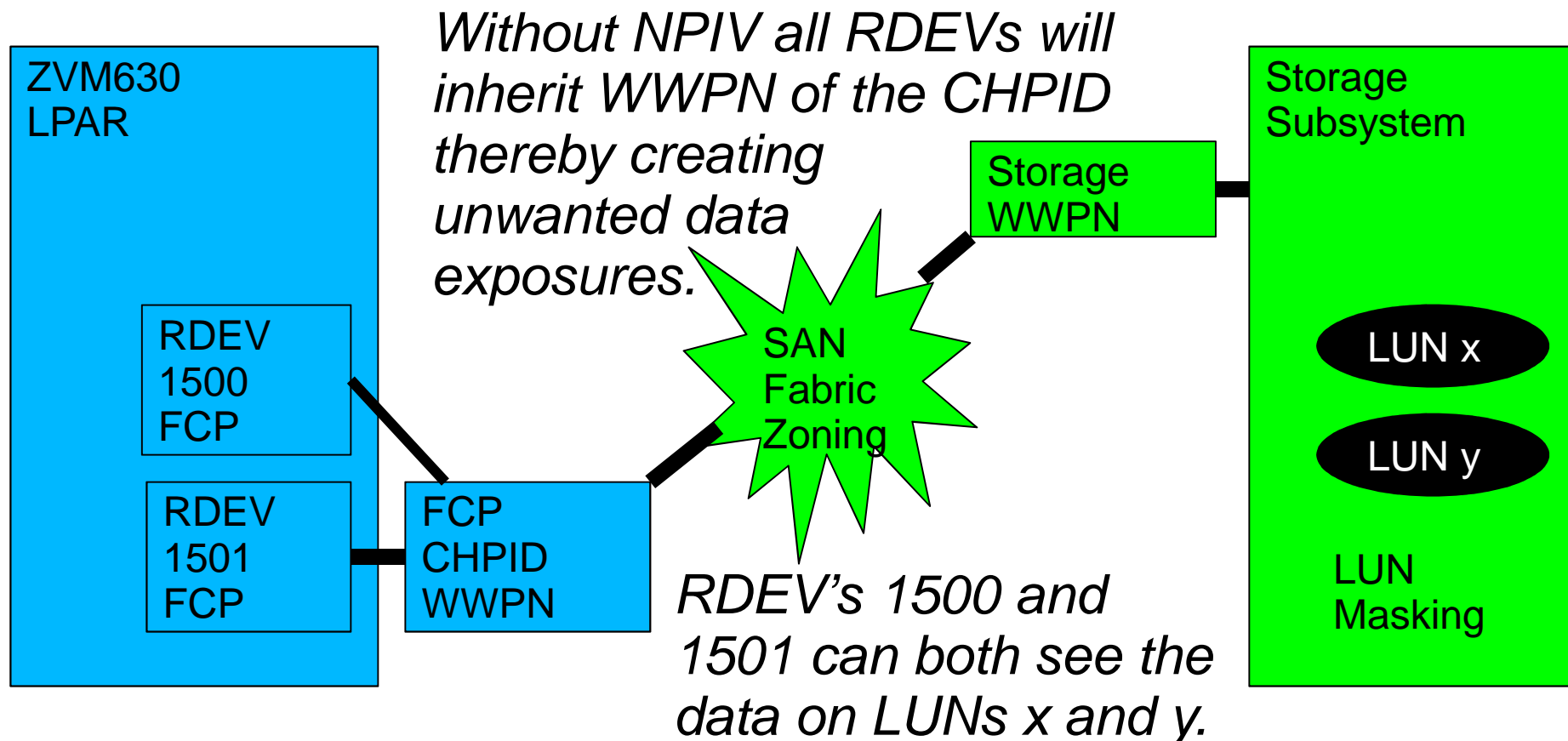
# Simplified Topology



# NPIV solves data exposure problem

## This example shows the undesired state

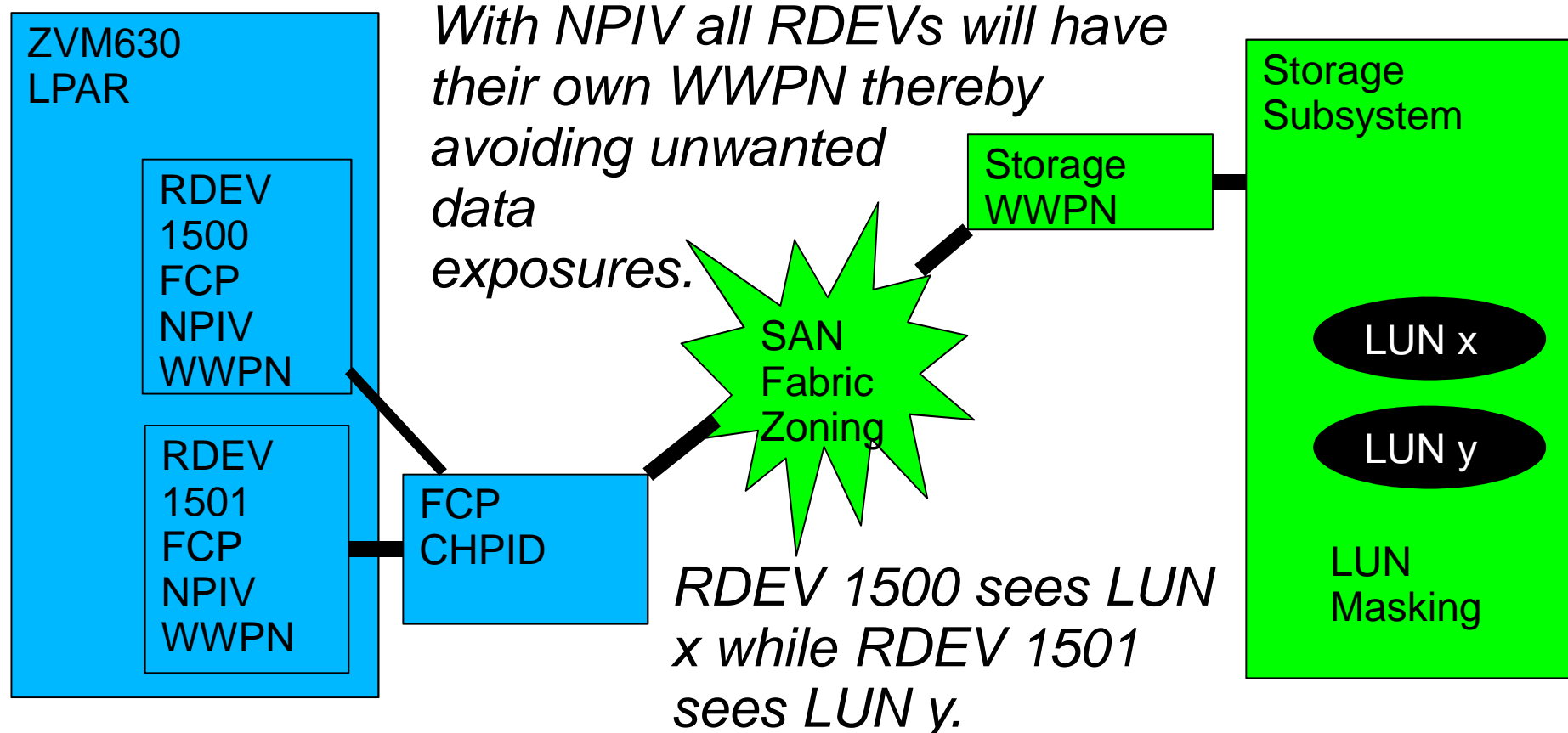
Desired state is RDEV 1500 to see LUN x and RDEV 1501 to see LUN y.



# NPIV solves data exposure problem

## This example shows the desired state

Desired state is RDEV 1500 to see LUN x and RDEV 1501 to see LUN y.





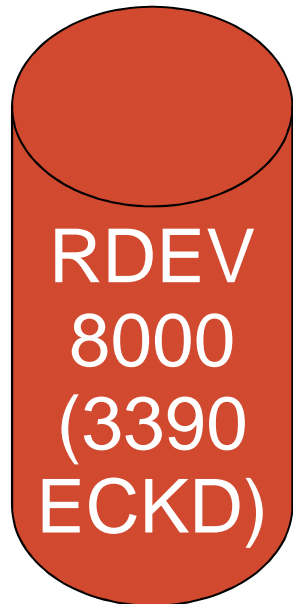
# HMC Display of NPIVs

Partition	CSS	IID	CHPID	SSID	Device Number	WWPN	NPIV Mode	Current Configured
PROD01	00	01	00	00	1500	c05076dcbd000700	On	Yes
PROD01	00	01	00	00	1501	c05076dcbd000704	On	Yes

These are the NPIVs assigned from the HMC. You do not need to specify in z/VM. Note: 1500 and 1501 are the same RDEV on both LPARs but have different NPIV WWPNs! Meaning that they can be zoned to different LUNs! Got it!?

PROD02	00	04	00	00	1500	c05076dcbd0007c0	On	Yes
PROD02	00	04	00	00	1501	c05076dcbd0007c4	On	Yes

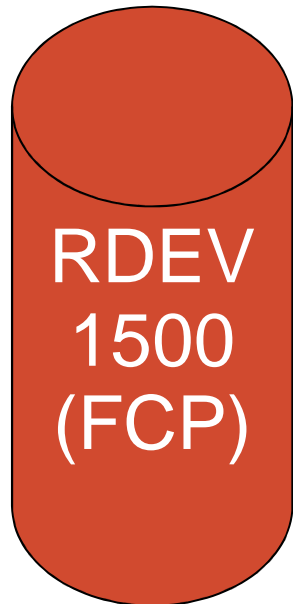
# z/VM, IOCDs, and 3390



CHPID, CNTLUNIT, and IODEVICE defined in IOCDs. Path selection chosen by I/O subsystem.

```
CHPID PATH=(CSS(0,1),B0),SHARED,NOTPART=((CSS(0),(CF2),(=))), *  
      SWITCH=03,PCHID=518,TYPE=FC  
:  
CNTLUNIT CUNUMBR=8000, *  
          PATH=((CSS(0),B0,B1,B8,B9,B4,B5,B6,B7),(CSS(1),B0,B1,B8, *  
          B9,B4,B5,B6,B7)),UNITADD=((00,256)), *  
          LINK=((CSS(0),0318,0319,0418,0419,0322,0323,0422,0423),(*  
          CSS(1),0318,0319,0418,0419,0322,0323,0422,0423)), *  
          CUADD=30,UNIT=2107  
IODEVICE ADDRESS=(8000,224),CUNUMBR=(8000),STADET=Y,UNIT=3390B
```

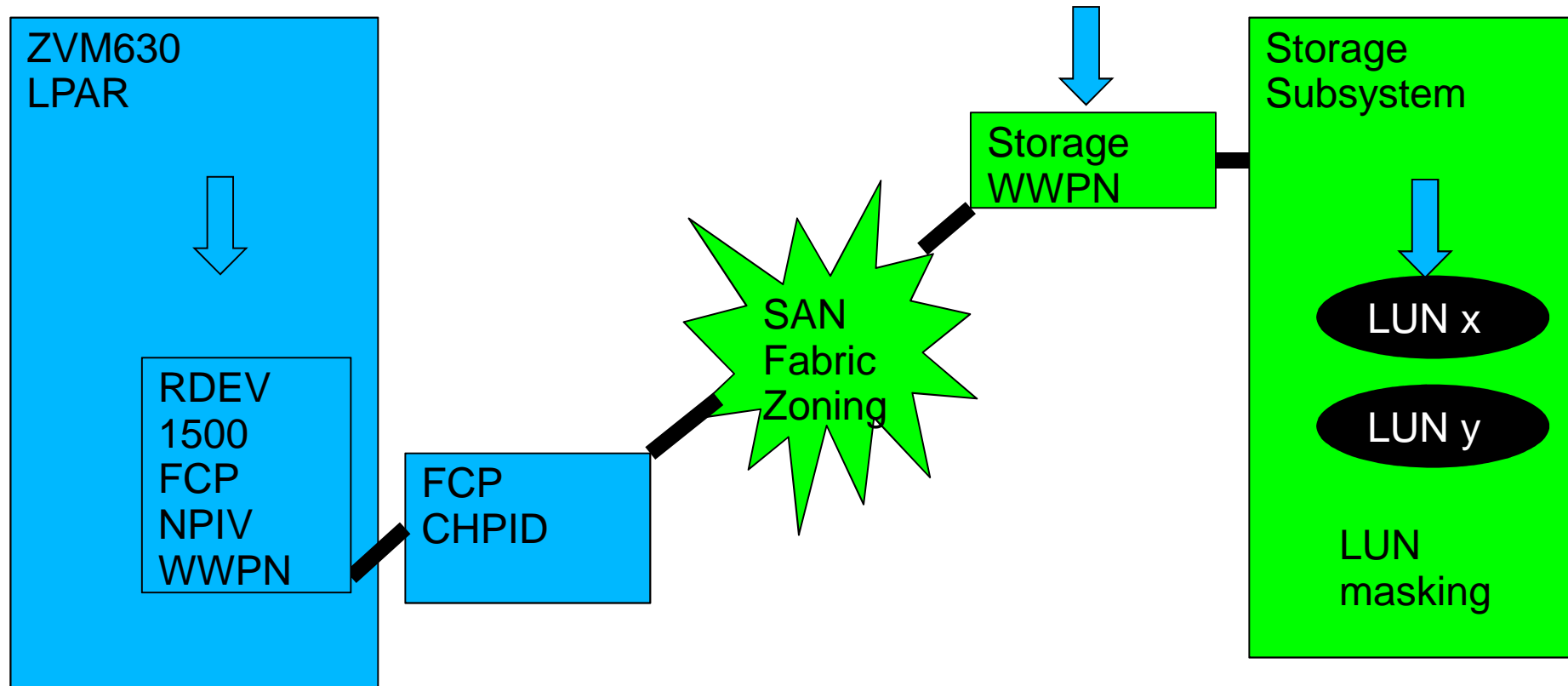
# z/VM, IOCDs, and FCP



CHPID, CNTLUNIT, and IODEVICE continue to be defined in the IOCDs. Single path.

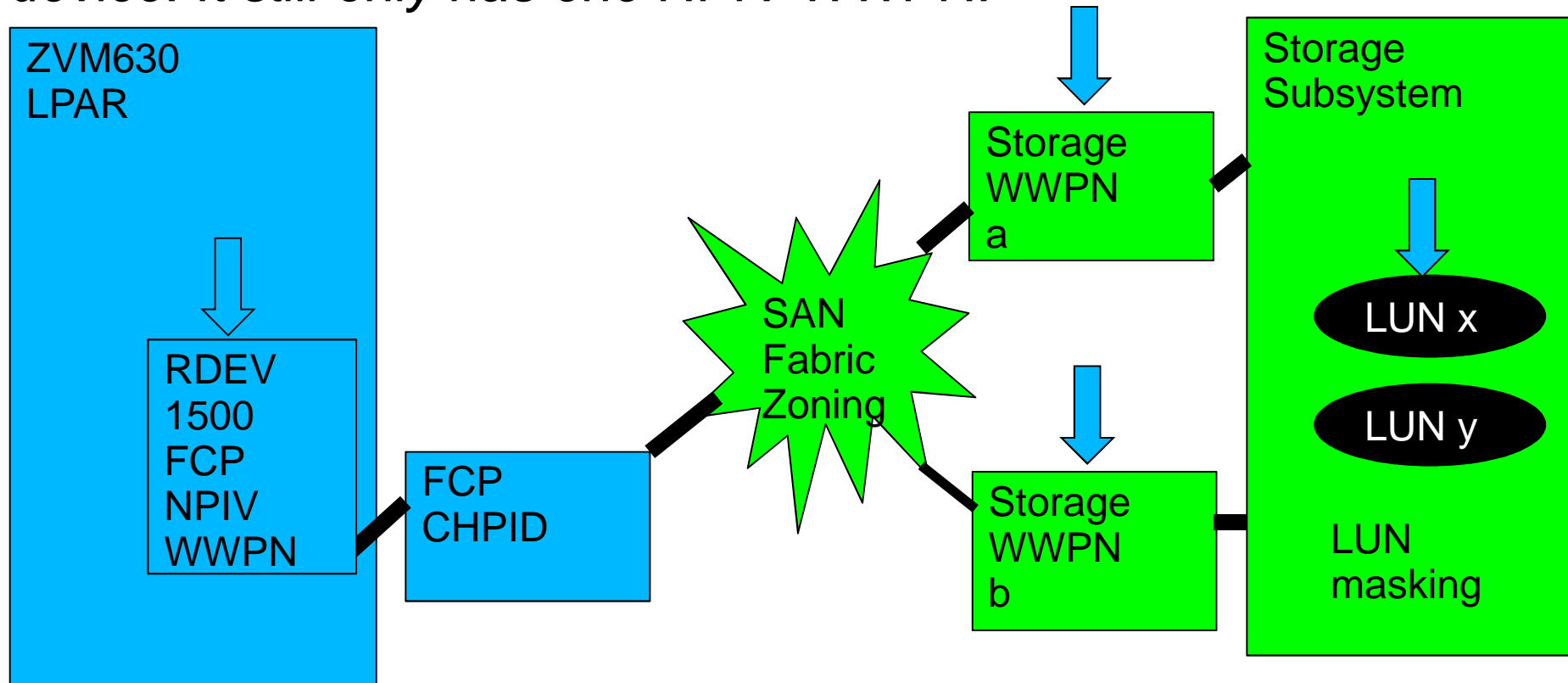
```
CHPID PATH=(CSS(0),00),SHARED, *  
      PARTITION=((PROD_ONE,PROD_TWO,QADEV,SERVICE),(=)),PCHID=5E8, *  
      TYPE=FCP  
CNTLUNIT CUNUMBR=1500,PATH=((CSS(0),00)),UNIT=FCP  
IODEVICE ADDRESS=(1500,016),CUNUMBR=(1500),UNIT=FCP
```

# z/VM EDEV Needs to Know: RDEV, target WWPN(s) and target LUN(s).



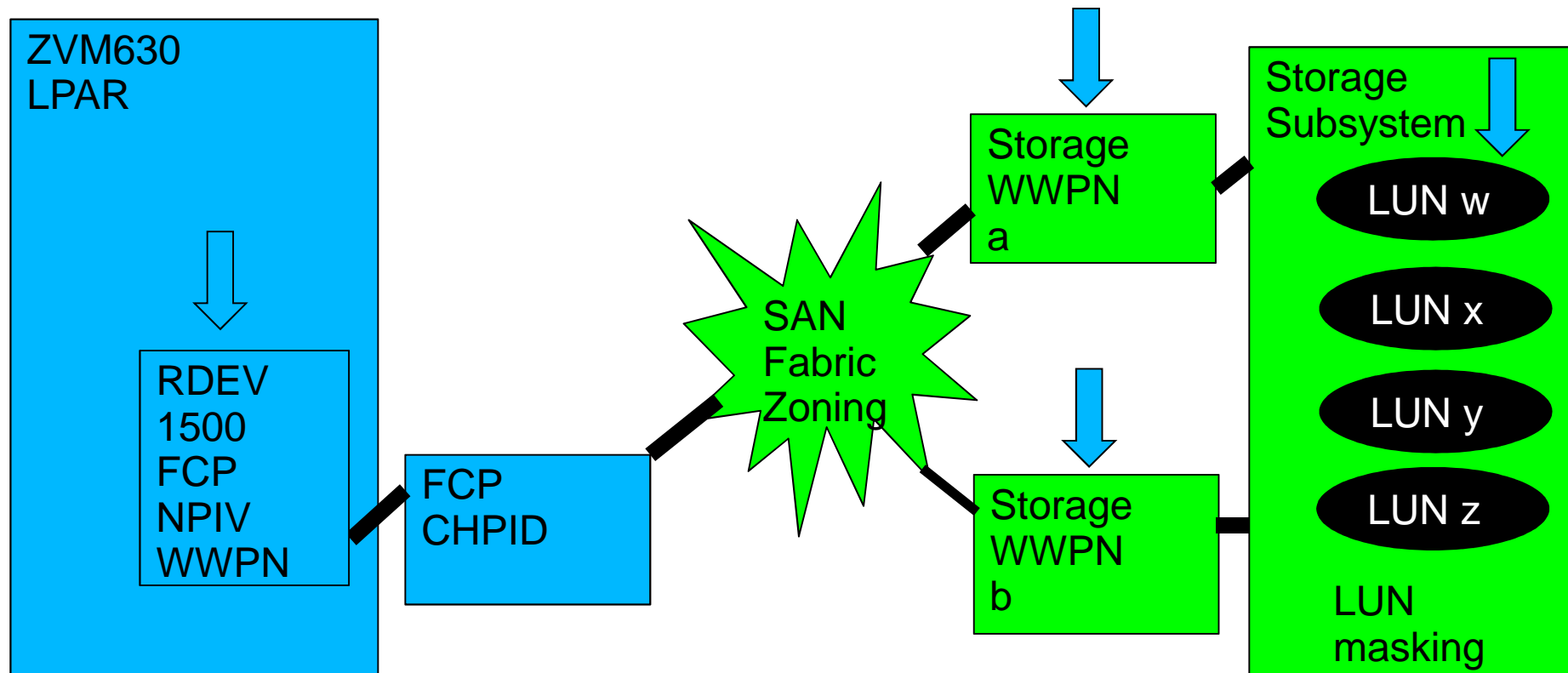
# z/VM EDEV Needs to Know: RDEV, target WWPN(s) and target LUN(s).

*A single RDEV can leap multiple WWPNs in a single bound, i.e., a single RDEV can login to multiple WWPNs in the storage device. It still only has one NPIV WWPN.*



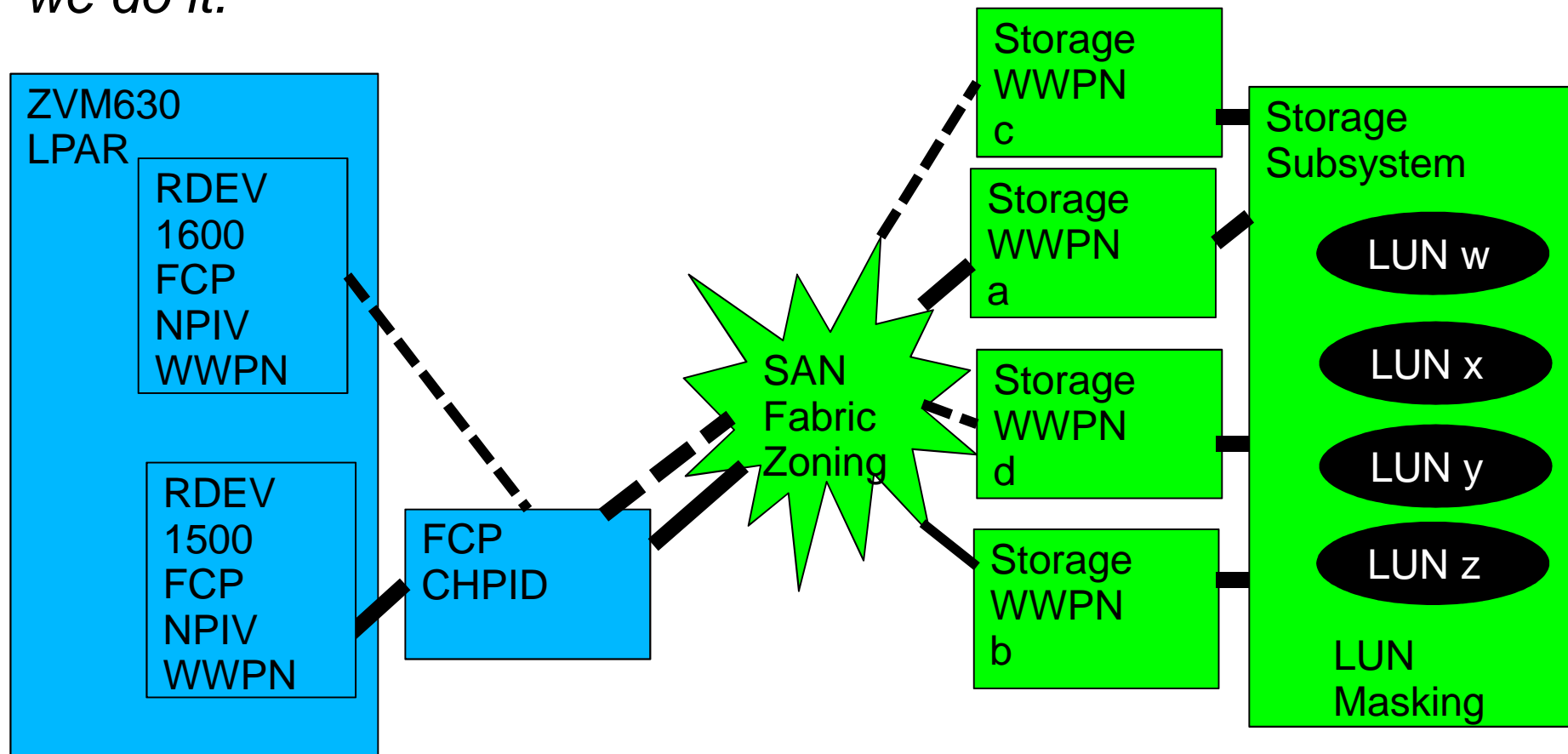
# z/VM EDEV Needs to Know: RDEV, target WWPN(s) and target LUN(s).

*A single RDEV can leap multiple WWPNs in a single bound, i.e., a single RDEV can login to multiple WWPNs in the storage device and see multiple LUNs. It still only has one NPIV WWPN.*



# z/VM EDEV Needs to Know: RDEV, target WWPN(s) and target LUN(s).

*Multiple RDEVs with NPIV attached to the SAME LUNs. 2 RDEVs each with two WWPNs to the same LUNs! This is how we do it.*



# z/VM and Emulated Devices

- EDEVs use a CP imbedded FBA emulator which in turn drives the I/O to the FCP devices.
- When running FCP only you must code the EDEVs in the SYSTEM CONFIG.
- EDEVs are treated as FBA devices.
  - Attached to the system
  - MDISKS defined on EDEV's volser as if it were really real.
    - As such can be handled by DIRMAINT
  - Supported as CP devices for paging, spooling, and directory.



# FCP SCSI EDEV Architecture

- Four LPARs running zvm630 non-SSI.
- Each LPAR using multiple LUNs per FCP subchannel.
  - Two subchannels each login to two WWPNs.
    - One NPIV WWPN per subchannel
- EDEV defined in SYSTEM CONFIG or dynamically by CP SET EDEV command.
  - Path defined to target WWPN (storage device) and the LUN.
    - One LUN per EDEV
- Each LPAR uses ED00-ED0F for VM and Linux opsys data and EE00-EE0F for paging and spooling.
  - These are emulated devices so I can do what I want!

# FCP EDEV IPL and Usage

- Will show IPL from FCP and EDEV
- SYSTEM CONFIG entries
- Paging to EDEV
- Dynamic usage of EDEV
  - Tooling
  - CP Commands
- Linux opsys data is on MDISKS on EDEV.
- EDEVs are device type 9336 or FB-512
- Mapping EDEV from other LPARs.
  - SERVICE LPAR is mapped to all FCP devices used for EDEV on all LPARs
  - Most system MDISKS are on same locations on different VOLSERs.

# SCSI IPL is fun!

HMC1: Load - Mozilla Firefox

https://192.168.101.21/hmc/content?taskId=136&refresh=521

Load - [REDACTED]

CPC: [REDACTED]

Image: [REDACTED]

Load type  Normal  Clear  SCSI  SCSI dump

Store status

Load address \* 01500 ← FCP RDEV

Load parameter SYSG

Time-out value 60 ← 60 to 600 seconds

Worldwide port name 500507680C2212F ← Target WWPN

Logical unit number 0

Boot program selector 0

Boot record logical block address C8 ← Must code

Operating system specific load parameters

# SCSI IPL is fun!

The SAPL screen points at the FCP device for IPL. The PDVOL parameter points to the EDEV that contains the IPL settings. The EDEV "ED00" must be in the online list coded in SYSTEM CONFIG.

```
STAND ALONE PROGRAM LOADER: z/VM VERSION 6 RELEASE 3.0

DEVICE NUMBER: 01500      MINIDISK OFFSET: 57200      EXTENT: 1
MODULE NAME:  CPLOAD      LOAD ORIGIN: 1000

-----IPL PARAMETERS-----
pdvol=ed00 pdnum=4

-----COMMENTS-----
zvm630 RSU 1501

-----
9= FILELIST  10= LOAD  11= TOGGLE EXTENT/OFFSET
```

# EDEVICE Statements in the SYSTEM CONFIG

*These two EDEVICE statement create EDEV's at ED00 and ED01. Each has two RDEVs with two WWPNs each. Built-in redundancy creates high availability.*

```
edevice ED00  type fba attr SCSI fcp_dev 1500,  
              wwpn 500507680C2212FC lun 0000000000000000,  
              fcp_dev 1500,  
              wwpn 500507680C531317 lun 0000000000000000,  
              fcp_dev 1600,  
              wwpn 500507680C2312FC lun 0000000000000000,  
              fcp_dev 1600,  
              wwpn 500507680C521317 lun 0000000000000000  
edevice ED01  type fba attr SCSI fcp_dev 1500,  
              wwpn 500507680C2212FC lun 0001000000000000,  
              fcp_dev 1500,  
              wwpn 500507680C531317 lun 0001000000000000,  
              fcp_dev 1600,  
              wwpn 500507680C2312FC lun 0001000000000000,  
              fcp_dev 1600,  
              wwpn 500507680C521317 lun 0001000000000000
```

# EDEVICES must be ONLINE at IPL time

```
Devices ,
  Offline_at_IPL 0000-FFFF,
  Online_at_IPL  ,
  330-34F,
  430-44F,
  530-54F,
  630-64F,
  1704,
  1804,
  1900-1910,
  1A00-1A10,
  1B00-1B10,
  1C00-1C10,
  C000-C00F,
  ED00-ED0F,
  EE00-EE0F,
  F200-F20F,
  F300-F30F,
  Sensed          0000-FFFF
```

All LPARs except the SERVICE LPAR use the “offline all online what’s needed” approach. Even though EDEVs aren’t real devices they must be in the online list.

The FCP devices must be online.

Otherwise you will not be able to have a good IPL!

# Paging to EE00-EE0F

EXTENT VOLID	EXTENT RDEV	TOTAL START	PAGES END	HIGH PAGES	% IN USE	PAGE USED	USED
RFPG00	EE00	4	2097151	2048K	7	7	1%
RFPG01	EE01	4	2097151	2048K	53	63	1%
RFPG02	EE02	4	2097151	2048K	0	0	0%
RFPG03	EE03	4	2097151	2048K	62	63	1%
RFPG04	EE04	4	2097151	2048K	0	0	0%
RFPG05	EE05	4	2097151	2048K	0	0	0%
RFPG06	EE06	4	2097151	2048K	0	0	0%
RFPG07	EE07	4	2097151	2048K	0	0	0%
RFPG08	EE08	4	2097151	2048K	0	0	0%
RFPG09	EE09	4	2097151	2048K	0	0	0%
RFPG0A	EE0A	4	2097151	2048K	0	0	0%
RFPG0B	EE0B	4	2097151	2048K	0	0	0%
RFPG0C	EE0C	4	2097151	2048K	43	63	1%
SUMMARY				26624K	165		1%
USABLE				26624K	165		1%
Ready; T=0.01/0.01 19:35:58							

# Great CP Commands for FCP

```
q edev ed00 details
```

```
EDEV ED00 TYPE FBA ATTRIBUTES SCSI
```

```
VENDOR: IBM PRODUCT: 2145 REVISION: 0000
```

```
BLOCKSIZE: 512 NUMBER OF BLOCKS: 16777216
```

```
PATHS:
```

```
FCP_DEV: 1500 WWPN: 500507680C2212FC LUN: 0000000000000000
```

```
CONNECTION TYPE: SWITCHED STATUS: ONLINE
```

```
FCP_DEV: 1500 WWPN: 500507680C531317 LUN: 0000000000000000
```

```
CONNECTION TYPE: SWITCHED STATUS: ONLINE
```

```
FCP_DEV: 1600 WWPN: 500507680C2312FC LUN: 0000000000000000
```

```
CONNECTION TYPE: SWITCHED STATUS: ONLINE
```

```
FCP_DEV: 1600 WWPN: 500507680C521317 LUN: 0000000000000000
```

```
CONNECTION TYPE: SWITCHED STATUS: ONLINE
```

```
EQID: 600507680C808097E0000000000000F0F0000000000000FFFFFF
```

```
Ready; T=0.01/0.01 19:57:29
```

```
q fcp wwpn 1500
```

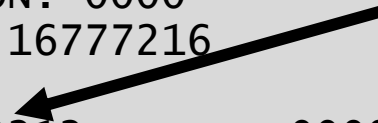
```
FCP 1500 NPIV WWPN C05076DCBD000780
```

```
CHPID 00 PERM WWPN C05076DCBD005E81
```

```
ATTACHED TO SYSTEM
```

```
Ready; T=0.01/0.01 19:58:17
```

WWPN of the storage



Us in VM and  
mainframe side





# DOEDEV Mapping Tool

```
/**/  
arg edev lun .  
  
'CP SET EDEV' edev ,  
'TYPE FBA ATTR SCSI',  
'FCP_DEV 1500 ',  
'WWPN 500507680C2212FC',  
'LUN ' lun'000000000000'  
  
'CP SET EDEV' edev ,  
'TYPE FBA ATTR SCSI',  
'ADD PATH',  
'FCP_DEV 1500 ',  
'WWPN 500507680C531317',  
'LUN ' lun'000000000000'  
  
'CP VARY ON' edev
```

*This tool will dynamically create an EDEV using the CP SET EDEV command. I use it to map any FCP used for EDEVs on any LPARs. Handily enough the storage guy graciously has double mapped all EDEVs on RDEV 1500 in the service zone! I take great care not to inadvertently write to another LPARs EDEV. I accept the risk. It is great for checking allocations, mapping MDISKS, backups, and creating new volumes.*

# Using the DOEDEV Mapping Tool

```
doedev 1000 000a
EDEV 1000 was created.
EDEV 1000 was modified.
1000 varied online
1 device(s) specified; 1 device(s) successfully varied online
Ready; T=0.01/0.01 20:13:13
q 1000
DASD 1000 RQRES1
Ready; T=0.01/0.01 20:13:15
q mdisk userid pmaint cf0 loc drct
TargetID Tdev OwnerID Odev Dtype Vol-ID Rdev StartLoc Size
PMAINT 0CF0 PMAINT 0CF0 9336 RSRES1 ED00 575600 172800
Ready; T=0.01/0.01 20:13:48
attach 1000 system
DASD 1000 ATTACHED TO SYSTEM RQRES1
Ready; T=0.01/0.01 20:14:04
def mdisk fcf0 575600 172800 rqres1
DASD FCF0 DEFINED
Ready; T=0.01/0.01 20:18:35
< do some editing via CMS, whatever >
```

*From the service zone map to  
PMAINT CF0 of a different LPAR!*

# Dynamic EDEV Commands

```
det fcf0
```

```
DASD FCF0 DETACHED
```

```
Ready; T=0.01/0.01 20:42:29
```

```
q edev 1000 details
```

```
EDEV 1000 TYPE FBA ATTRIBUTES SCSI
```

```
VENDOR: IBM PRODUCT: 2145 REVISION: 0000
```

```
BLOCKSIZE: 512 NUMBER OF BLOCKS: 16777216
```

```
PATHS:
```

```
FCP_DEV: 1500 WWPN: 500507680C2212FC LUN: 000A000000000000
```

```
CONNECTION TYPE: SWITCHED STATUS: ONLINE
```

```
FCP_DEV: 1500 WWPN: 500507680C531317 LUN: 000A000000000000
```

```
CONNECTION TYPE: SWITCHED STATUS: ONLINE
```

```
EQID: 600507680C808097E0000000000000F0C50000000000FFFFFF
```

```
Ready; T=0.01/0.01 20:42:37
```

```
det 1000 system
```

```
DASD 1000 DETACHED SYSTEM
```

```
Ready; T=0.01/0.01 20:43:19
```

```
vary off 1000
```

```
1000 varied offline
```

```
1 device(s) specified; 1 device(s) successfully varied offline
```

```
Ready; T=0.01/0.01 20:43:23
```

```
set edev 1000 clear
```

```
EDEV 1000 was cleared.
```

```
Ready; T=0.01/0.01
```

*Wind it all down.*

*Be aware that you can  
clobber data when disk  
mapping!*

# FCP EDEV IPL and Usage

- Will show IPL from FCP and EDEV
- SYSTEM CONFIG entries
- Paging to EDEV
- Dynamic usage of EDEV
  - Tooling
  - CP Commands
- Mapping EDEV from other LPARs.
  - SERVICE LPAR is mapped to all FCP devices used for EDEV on all LPARs
  - Most system MDISKs are on same locations on different VOLSERs.

# Linux FCP Usage

- Will show IPL from FCP and EDEV
- FCP devices dedicated to the Linux machine
  - Sole use of entire subchannel
  - Advantageous?
- Linux will learn the WWPNs and LUNs.
- Linux tailored to use multipathing.
- Linux payload data (databases) live on the FCP.
- SCSI LUNs are available from 4 subchannels each with 2 WWPN's giving 8 paths to a LUN!
- There are 10 LUNs so there are 80 (8 paths X 10) devices for Linux to manage.
  - Manages in multipathing
  - Aggregates devices so that a unique LUN is mapped to WWID.
- Linux distribution is SLES11SP3.

# Adding FCP Devices in Linux

- Comprehensive command set available to add and manage FCP devices.
  - vmcp, lszfcp, lsluns, lsscsi, zfcplib\_configure,
- Also yast has complete interface.
- Must use multipathing for redundancy and I/O balancing.
- Will show the command interface to manage two dedicated FCP devices each with two WWPN interfaces to two LUNs.

# Linux Directory Entry

```
USER LINUXA2 password 32G 1024G G
```

```
DEDICATE 1000 1704
```

```
DEDICATE 1001 1804
```

```
DEDICATE 1002 1904
```

```
DEDICATE 1003 1A04
```

```
MDISK 0150 FB-512 69601200 600000 RFLXLO MR
```

```
MDISK 0151 FB-512 1200 69600000 RFLXLO MR
```

```
MDISK 0303 FB-512 70201200 16800000 RFLXLO MR
```

*Real addresses always DEDICATED  
in the virtual 1000-100x range.*

*MDISKS on EDEV*

Linux has an I/O driver to the SCSI FCP. When dedicating the subchannels to the Linux machine the I/O is managed by the driver. CP is still involved but as a pass through.

# Linux: sample of IPL time messages concerning SCSI

```
SCSI subsystem initialized
rdac: device handler registered
```

```
alua: device handler registered
emc: device handler registered
hp_sw: device handler registered
Creating device nodes with udev
udev: starting version 147
```

```
scsi0 : zfc
```

```
qdio: 0.0.1000 ZFCP on SC 0 using AI:1 QEBSM:1 PCI:1 TDD:1 SIGA: W A
```

```
scsi1 : zfc
```

```
qdio: 0.0.1001 ZFCP on SC 1 using AI:1 QEBSM:1 PCI:1 TDD:1 SIGA: W A
```

```
scsi 0:0:0:0: Direct-Access      IBM          2145          0000 PQ: 0 ANSI: 6
```

```
scsi2 : zfc
```

```
qdio: 0.0.1002 ZFCP on SC 2 using AI:1 QEBSM:1 PCI:1 TDD:1 SIGA: W A
```

```
sd 1:0:0:9: [sda1] Attached SCSI disk
```

```
scsi3 : zfc
```

```
qdio: 0.0.1003 ZFCP on SC 3 using AI:1 QEBSM:1 PCI:1 TDD:1 SIGA: W A
```

*Some but by no means all of the  
IPL time messages.*



# Linux: sample of IPL time messages concerning SCSI

```
sd 0:0:1:0: [sdb] Attached SCSI disk
sd 0:0:0:0: [sda] Attached SCSI disk
sd 0:0:1:1: [sdd] Attached SCSI disk
sd 0:0:0:1: [sdc] Attached SCSI disk
sd 0:0:1:2: [sde] Attached SCSI disk
sd 0:0:1:3: [sdf] Attached SCSI disk
sd 0:0:1:4: [sdg] Attached SCSI disk
sd 0:0:0:2: [sdi] Attached SCSI disk
sd 0:0:0:3: [sdj] Attached SCSI disk
sd 0:0:0:4: [sdk] Attached SCSI disk
sd 0:0:0:5: [sd] Attached SCSI disk
:
```

These devices correspond to a LUN  
are visible in the /dev directory:

```
ls /dev/sd* -l|wc -l
80
```

# Fun commands: lszfcp

```
linuxa2:/etc/multipath # lszfcp -H
0.0.1000 host0
0.0.1001 host1
0.0.1002 host2
0.0.1003 host3
ddpixa2:/etc/multipath # lszfcp -P
0.0.1000/0x500507680c531317 rport-0:0-0
0.0.1000/0x500507680c2212fc rport-0:0-1
0.0.1001/0x500507680c521317 rport-1:0-0
0.0.1001/0x500507680c2312fc rport-1:0-1
0.0.1002/0x500507680c231317 rport-2:0-0
0.0.1002/0x500507680c5212fc rport-2:0-1
0.0.1003/0x500507680c221317 rport-3:0-0
0.0.1003/0x500507680c5312fc rport-3:0-1
ddpixa2:/etc/multipath #
```

*-H list the virtual device and the host it is logged onto.*

*-P list the virtual device, the storage wwpn, and the remote port.*

# Fun commands: lszfcp

```
linuxa2:/etc/multipath # lszfcp -l 0x0009000000000000
0.0.1000/0x500507680c531317/0x0009000000000000 0:0:0:9
0.0.1000/0x500507680c2212fc/0x0009000000000000 0:0:1:9
0.0.1001/0x500507680c521317/0x0009000000000000 1:0:0:9
0.0.1001/0x500507680c2312fc/0x0009000000000000 1:0:1:9
0.0.1002/0x500507680c231317/0x0009000000000000 2:0:0:9
0.0.1002/0x500507680c5212fc/0x0009000000000000 2:0:1:9
0.0.1003/0x500507680c221317/0x0009000000000000 3:0:0:9
0.0.1003/0x500507680c5312fc/0x0009000000000000 3:0:1:9
```

*List the virtual devices, storage WWPNS, and remote ports that are used by LUN 9.*

# Fun commands: lsscsi

```
linuxa2:/etc/multipath # lsscsi 3H: --device
[3:0:0:0]    disk    IBM      2145      0000    /dev/sdbi [67:192]
[3:0:0:1]    disk    IBM      2145      0000    /dev/sdb1 [67:240]
[3:0:0:2]    disk    IBM      2145      0000    /dev/sdbm [68:0]
[3:0:0:3]    disk    IBM      2145      0000    /dev/sdbp [68:48]
[3:0:0:4]    disk    IBM      2145      0000    /dev/sdbq [68:64]
[3:0:0:5]    disk    IBM      2145      0000    /dev/sdb1 [68:80]
[3:0:0:6]    disk    IBM      2145      0000    /dev/sdb5 [68:96]
[3:0:0:7]    disk    IBM      2145      0000    /dev/sdbv [68:144]
[3:0:0:8]    disk    IBM      2145      0000    /dev/sdbw [68:160]
[3:0:0:9]    disk    IBM      2145      0000    /dev/sdcb [68:240]
[3:0:1:0]    disk    IBM      2145      0000    /dev/sdbj [67:208]
[3:0:1:1]    disk    IBM      2145      0000    /dev/sdbk [67:224]
[3:0:1:2]    disk    IBM      2145      0000    /dev/sdbn [68:16]
[3:0:1:3]    disk    IBM      2145      0000    /dev/sdbo [68:32]
[3:0:1:4]    disk    IBM      2145      0000    /dev/sdbt [68:112]
[3:0:1:5]    disk    IBM      2145      0000    /dev/sdbu [68:128]
[3:0:1:6]    disk    IBM      2145      0000    /dev/sdbx [68:176]
[3:0:1:7]    disk    IBM      2145      0000    /dev/sdb1 [68:192]
[3:0:1:8]    disk    IBM      2145      0000    /dev/sdbz [68:208]
[3:0:1:9]    disk    IBM      2145      0000    /dev/sdca [68:224]
```

*Display devices belong to host 3 and output the vendor, device type, device node entry and the major and minor node.*

# Linux FCP Usage

- Will show IPL from FCP and EDEV
- FCP devices dedicated to the Linux machine
  - Sole use of entire subchannel
  - Advantageous?
- Linux will learn the WWPNs and LUNs.
- Linux tailored to use multipathing.
- Linux payload data (databases) live on the FCP.
- SCSI LUNs are available from 4 subchannels each with 2 WWPN signon giving 8 paths to a LUN!
- There are 10 LUNs so there are 80 (8 paths X 10) devices for Linux to manage.
  - Manages in multipathing
  - Aggregates devices so that a unique LUN is mapped to WWID.

# Adding FCP Devices in Linux

- Comprehensive command set available to add and manage FCP devices.
  - vmcp, lszfcp, lsluns, lsscsi, zfcplib\_configure,
- Also yast has complete interface.
- Must use multipathing for redundancy and I/O balancing.
- Will show the command interface to manage two dedicated FCP devices each with two WWPN interfaces to two LUNs.

# Use vmcp to query the devices

```
linux1:/sbin # vmcp q fcp
FCP 1000 ON FCP 1503 CHPID 00 SUBCHANNEL = 0000
1000 TOKEN = 0000000037F3A400
1000 DEVTYPE FCP VIRTUAL CHPID 00 FCP REAL CHPID 00
1000 QDIO ACTIVE QIOASSIST ACTIVE QEBSM
1000
1000 INP + 01 IOCNT = 00002895 ADP = 128 PROG = 000 UNAVAIL = 000
1000 BYTES = 0000000000000000
1000 OUT + 01 IOCNT = 00002939 ADP = 000 PROG = 128 UNAVAIL = 000
1000 BYTES = 0000000000EE7616
1000 DATA ROUTER ELIGIBLE
WWPN C05076DCBD00078C
FCP 1001 ON FCP 1603 CHPID 01 SUBCHANNEL = 0001
1001 TOKEN = 0000000037F3A300
1001 DEVTYPE FCP VIRTUAL CHPID 01 FCP REAL CHPID 01
1001 QDIO ACTIVE QIOASSIST ACTIVE QEBSM
1001
1001 INP + 01 IOCNT = 00001201 ADP = 128 PROG = 000 UNAVAIL = 000
1001 BYTES = 0000000000000000
1001 OUT + 01 IOCNT = 00001242 ADP = 000 PROG = 128 UNAVAIL = 000
1001 BYTES = 00000000004B6359
1001 DATA ROUTER ELIGIBLE
WWPN C05076DCBD00018C
```

# The zfcplib\* family: zfcplib\_san\_disc will discover what's what

```
linux1:/sbin # zfcplib_san_disc -b 0.0.1000 -w
0x500507680c231317
0x500507680c5212fc
linux1:/sbin # zfcplib_san_disc -b 0.0.1000 -p 0x500507680c231317 -L
0x0000000000000000
0x0001000000000000
linux1:/sbin # zfcplib_san_disc -b 0.0.1000 -p 0x500507680c5212fc -L
0x0000000000000000
0x0001000000000000
linux1:/sbin # zfcplib_san_disc -b 0.0.1001 -w
0x500507680c221317
0x500507680c5312fc
linux1:/sbin # zfcplib_san_disc -b 0.0.1001 -p 0x500507680c5312fc -L
0x0000000000000000
0x0001000000000000
linux1:/sbin # zfcplib_san_disc -b 0.0.1001 -p 0x500507680c221317 -L
0x0000000000000000
0x0001000000000000
```



# The zfcplib\* family: zfcplib\_san\_disc will discover what's what

```
linux1:/sbin # zfcplib_san_disc -b 0.0.1000 -w  
0x500507680c231317  
0x500507680c5212fc  
linux1:/sbin # zfcplib_san_disc -b 0.0.1000 -p 0x500507680c231317 -L  
0x0000000000000000  
0x0001000000000000  
linux1:/sbin # zfcplib_san_disc -b 0.0.1000 -p 0x500507680c5212fc -L  
0x0000000000000000  
0x0001000000000000
```

Ask for the target WWPN then  
ask the WWPN for the LUNs.

# The zfcplib\* family: zfcplib\_disk\_configure bring it up and online

```
linux1:/sbin # zfcplib_disk_configure 0.0.1000 0x500507680c5212fc 0x0000000000000000 1
Configuring FCP disk 500507680c5212fc:0000000000000000
linux1:/sbin # zfcplib_disk_configure 0.0.1000 0x500507680c5212fc 0x0001000000000000 1
Configuring FCP disk 500507680c5212fc:0001000000000000
linux1:/sbin # zfcplib_disk_configure 0.0.1000 0x500507680c231317 0x0001000000000000 1
Configuring FCP disk 500507680c231317:0001000000000000
linux1:/sbin # zfcplib_disk_configure 0.0.1000 0x500507680c231317 0x0000000000000000 1
Configuring FCP disk 500507680c231317:0000000000000000
linux1:/sbin # zfcplib_disk_configure 0.0.1001 0x500507680c221317 0x0000000000000000 1
Configuring FCP disk 500507680c221317:0000000000000000
linux1:/sbin # zfcplib_disk_configure 0.0.1001 0x500507680c221317 0x0001000000000000 1
Configuring FCP disk 500507680c221317:0001000000000000
linux1:/sbin # zfcplib_disk_configure 0.0.1001 0x500507680c5312fc 0x0001000000000000 1
Configuring FCP disk 500507680c5312fc:0001000000000000
linux1:/sbin # zfcplib_disk_configure 0.0.1001 0x500507680c5312fc 0x0000000000000000 1
Configuring FCP disk 500507680c5312fc:0000000000000000
linux1:/sbin #
```

# The zfcplib\* family: zfcplib\_disk\_configure bring it up and online

```
linux1:/sbin # zfcplib_disk_configure 0.0.1000 0x500507680c5212fc 0x0000000000000000 1  
Configuring FCP disk 500507680c5212fc:0000000000000000  
linux1:/sbin # zfcplib_disk_configure 0.0.1000 0x500507680c5212fc 0x0001000000000000 1  
Configuring FCP disk 500507680c5212fc:0001000000000000
```

Specify the devices address,  
WWPN and LUN to bring it  
online.

# Multipathing i/o

- Technique that provides multiple physical paths to an I/O device.
  - Fault tolerant
  - Performance enhancement
  - Load balancing
  - Round robin
  - Best service time
- Takes defaults or the settings from `/etc/multipath.conf`
- Linux implements with the `multipathd` daemon:  
`chkconfig multipathd on`  
`chkcongig boot.multipath on`

# multipath.conf – set defaults and 2145 specific settings.

```
/etc/multipath.conf
defaults {
polling_interval 30
checker_timeout 10
path_selector "round-robin 0"
}

devices {
device {
vendor "IBM "
product "2145"
path_grouping_policy group_by_prio
path_checker tur
prio "alua"
rr_weight priorities
no_path_retry "5"
failback immediate
dev_loss_tmo 120
fast_io_fail_tmo 25
rr_min_io_rq 1
}
}
```

These settings were taken from the 2145 manual, IBM suggestions, and trial and error to ensure that there are 4 paths active and paths standby to each WWID. Copy/Paste is your friend.

If you change settings when multipath is running you can reload with “multipath –r”.

# multipath.conf: blacklist and simple names.

```
blacklist {  
  wwid "*"   
}  
blacklist_exceptions {  
  wwid "36005076*"   
}  
multipaths {  
  multipath {  
    wwid 3600507680c808097e000000000000072  
    alias mpatha  
  }  
  :  
  :  
  multipath {  
    wwid 3600507680c808097e00000000000007b  
    alias mpathj  
  }  
}
```

Exclude all WWIDs except for the ones we want (start with 36005076\*).

Assign a short name based on a match with the WWID. The short name can be used when referring to the device, such as mount, fdisk, etc.

# Fun commands: multipath --ll

```
linuxa2:/etc/multipath # multipath -ll mpatha
mpatha (3600507680c808097e00000000000072) dm-3 IBM      ,2145
size=200G features='1 queue_if_no_path' hwhandler='0' wp=rw
|+- policy='round-robin 0' prio=50 status=active
| |- 0:0:1:0 sdb  8:16   active ready running
| |- 1:0:1:0 sdv  65:80   active ready running
| |- 2:0:1:0 sdap 66:144  active ready running
| `-- 3:0:1:0 sdbj 67:208  active ready running
`+- policy='round-robin 0' prio=10 status=enabled
  |- 0:0:0:0 sda  8:0     active ready running
  |- 1:0:0:0 sdu  65:64   active ready running
  |- 2:0:0:0 sdao 66:128  active ready running
  `-- 3:0:0:0 sdbi 67:192  active ready running
```

List the short name “mpatha” and display its status including size, device type, policy, devices, etc. “active” state lists the current i/o paths that will be used while “enabled” are the standby paths in case of “the troubles”.

# Use Isluns to show the ports and the LUNs

```
linux1:/sbin # lsluns
Scanning for LUNS on adapter 0.0.1000
  at port 0x500507680c231317:
    0x0000000000000000
    0x0001000000000000
  at port 0x500507680c5212fc:
    0x0000000000000000
    0x0001000000000000
Scanning for LUNS on adapter 0.0.1001
  at port 0x500507680c221317:
    0x0000000000000000
    0x0001000000000000
  at port 0x500507680c5312fc:
    0x0000000000000000
    0x0001000000000000
```

The Isluns command will show the ports and the LUNs. Another great command.



# z/VM and FCP Conclusions

- EDEV IPL works quite well.
  - EDEVs contain FBA minidisks.
- Use EDEV for VM and Linux opsys data
- Use DEDICATED FCP for Linux payloads.
- Seems fast:
  - Elapsed time in DDR
  - Client running extensive I/O timing checks
- Many commands and tools in both VM and Linux.
  - Not shown but valuable is the CMS SCSIDISC tool for WWPN and LUN discovery on attached FCP devices.
  - Starting to work with tape devices too.
- Linux multipathing is powerful.
  - Spent a lot of time getting multipathing options set just right.
- Fault tolerance must be created for EDEVs.
- Make friends with your storage administrators.

# Shout Out to My Crew at Vicom

Len Santalucia



Alex Kim



Gregorz Powiedziuk



John Wolfgang



**For More Information please contact...**

**Len Santalucia**, CTO & Business Development Manager

Vicom Infinity, Inc.

One Penn Plaza – Suite 2010

New York, NY 10119

804-918-3728 office

917-856-4493 mobile

[lsantalucia@vicominfinity.com](mailto:lsantalucia@vicominfinity.com)

## **About Vicom Infinity**

Account Presence Since Late 1990's

IBM Premier Business Partner

Reseller of IBM Hardware, Software, and Maintenance

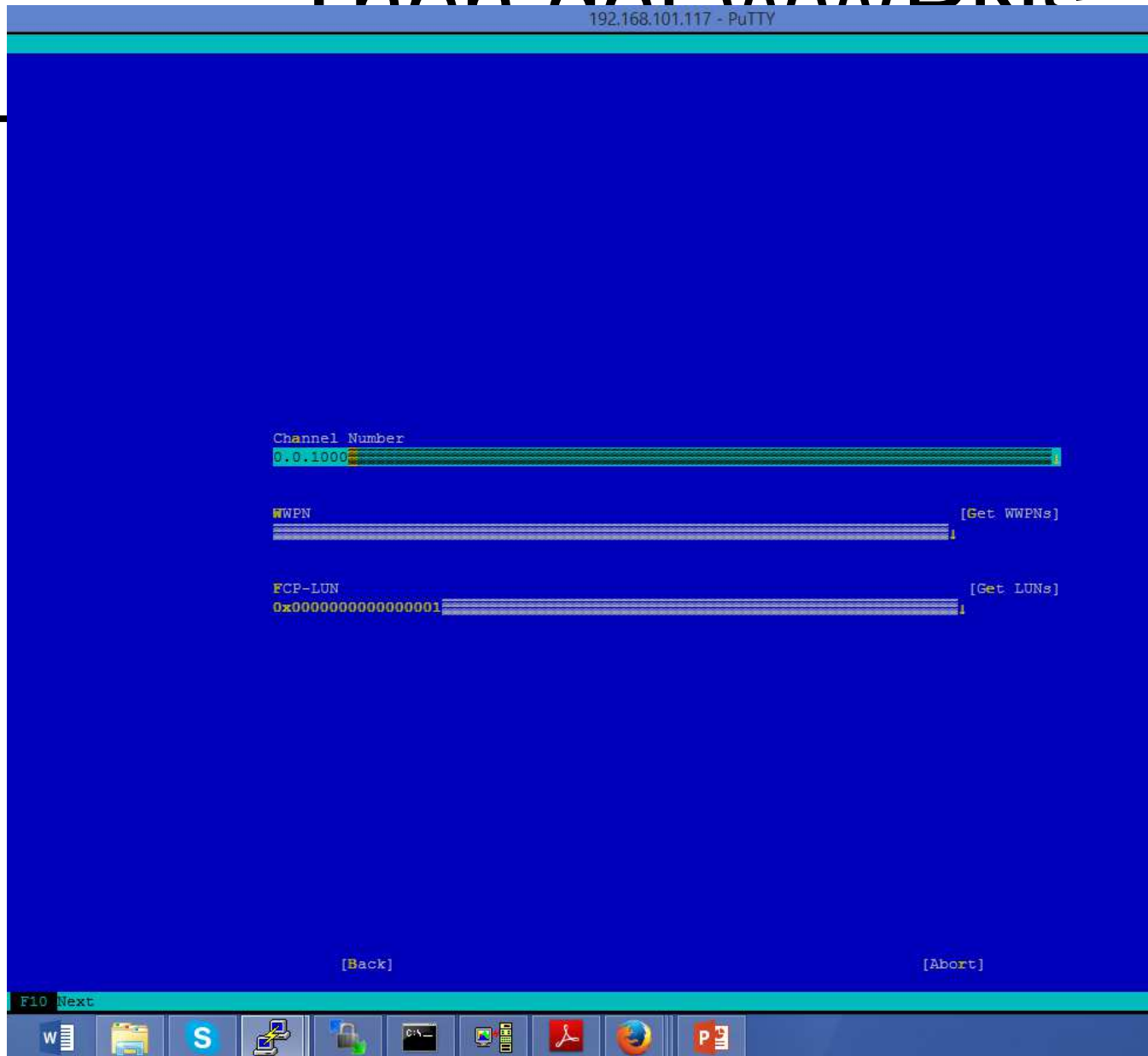
Vendor Source for the Last 10 Generations of Mainframes/IBM Storage

Professional and IT Architectural Services

Vicom Family of Companies Also Offer Leasing & Financing, Computer Services, and IT Staffing & IT Project Management



# Then get WWP/DNs



# Then get WWPNs

```
YaST2 - zfcpl @ linux1
Add New ZFCP Device

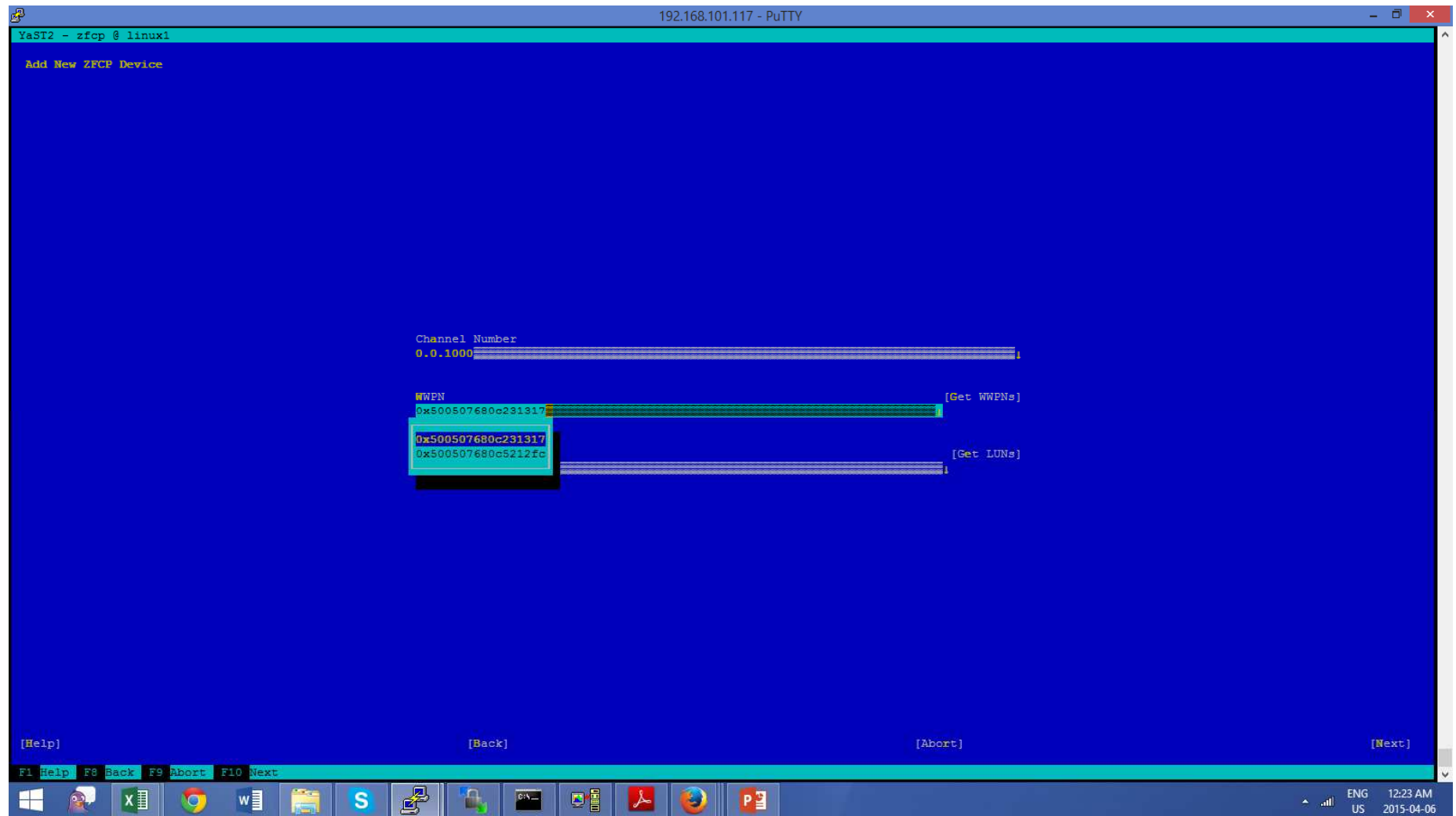
Channel Number
0-0.1000

WWPN                                [Get WWPNs]
0x500507680c231317

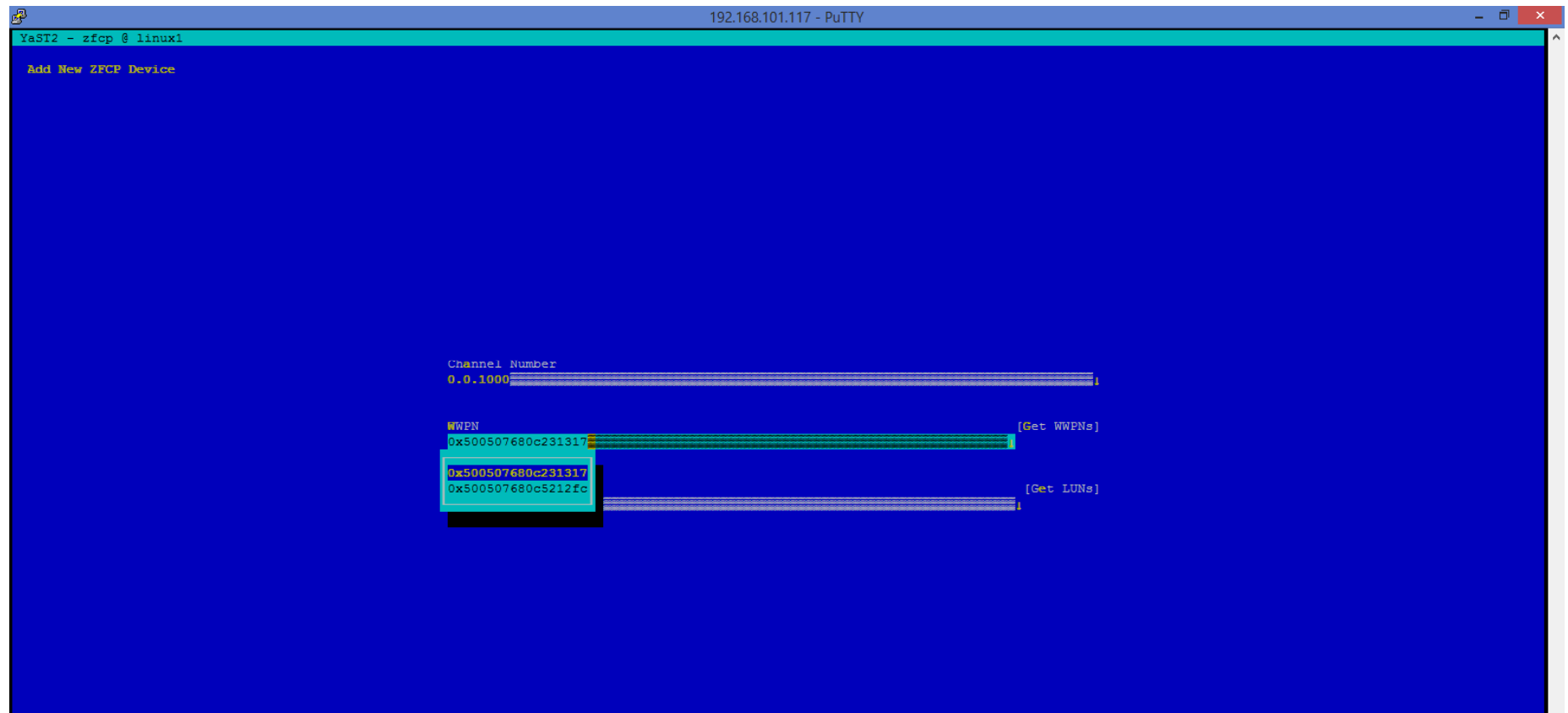
FCP-LUN                               [Get LUNs]
0x0000000000000001

[Help]                                [Back]                                [Abort]                                [Next]
```

# Then list WWPNs



# Then list WWPNs



The screenshot shows a terminal window titled "YaST2 - zfcpl @ linux1" with a blue background. The prompt is "Add New ZFCP Device". The user has entered "Channel Number" as "0.0.1000". Below this, the "WWPN" field is highlighted in green and contains "0x500507680c231317". To the right of this field is the label "[Get WWPNs]". Below the WWPN field, the "LUN" field is highlighted in cyan and contains "0x500507680c5212fc". To the right of this field is the label "[Get LUNs]".

```
YaST2 - zfcpl @ linux1
Add New ZFCP Device

Channel Number
0.0.1000

WWPN [Get WWPNs]
0x500507680c231317
0x500507680c5212fc [Get LUNs]
```



# Then list WWPNs

```
192.168.101.117 - PuTTY
YaST2 - zfcop @ linux1
Add New ZFCP Device

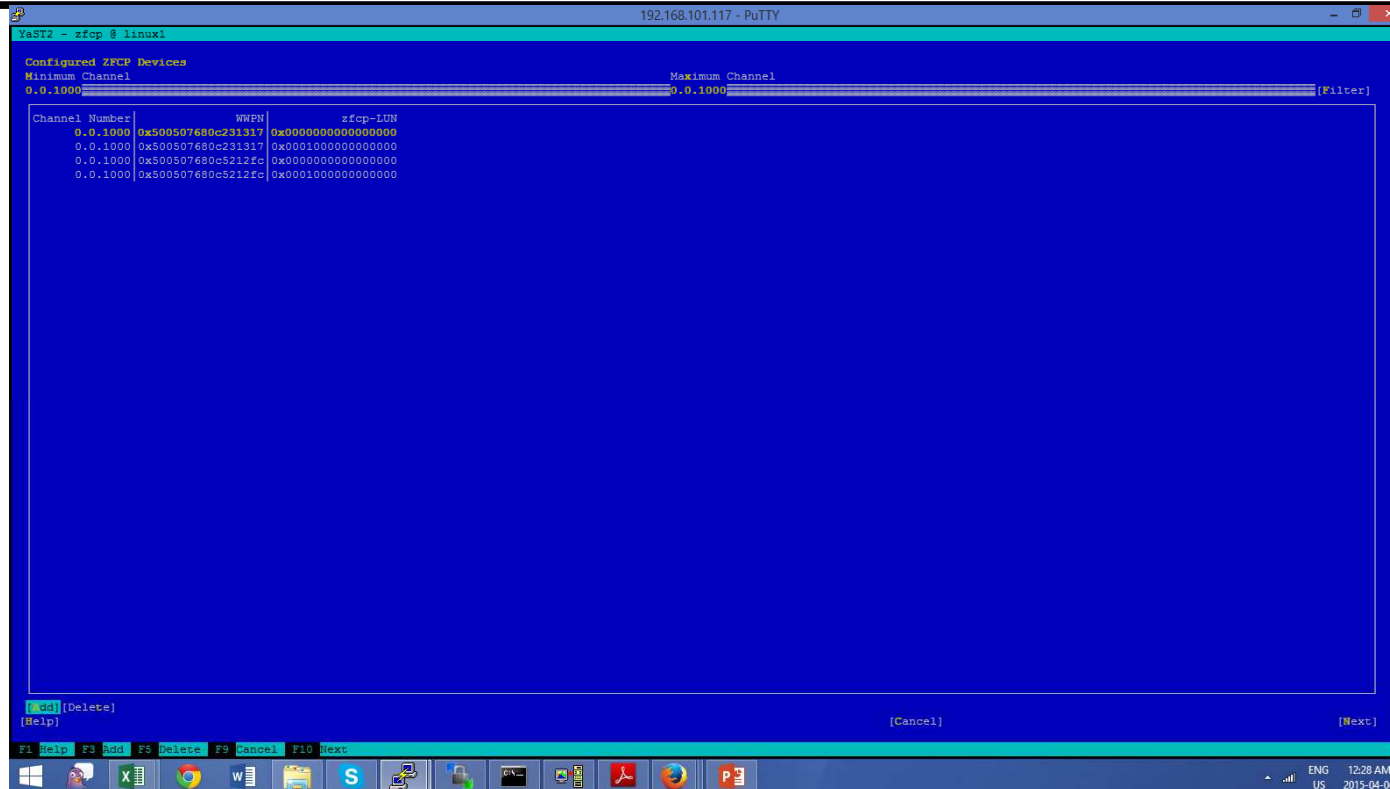
Channel Number
0.0.1000

WWPN [Get WWPNs]
0x500507680c231317

FCP-LUN [Get LUNs]
0x0000000000000000
0x0000000000000000
0x0001000000000000

[Help] [Back] [Abort]
F1 Help F8 Back F9 Abort F10 Next
```

# Then list WWPNs



The screenshot shows a terminal window titled '192.168.101.117 - PuTTY'. The terminal content is as follows:

```
VaST2 - zfcpl @ linux1
Configured ZFCP Devices
Minimum Channel: 0.0.1000
Maximum Channel: 0.0.1000 (Filter)

Channel Number | WWPN | zfcpl-LUN
0.0.1000 | 0x500507680c231317 | 0x0000000000000000
0.0.1000 | 0x500507680c231317 | 0x0001000000000000
0.0.1000 | 0x500507680c5212fc | 0x0000000000000000
0.0.1000 | 0x500507680c5212fc | 0x0001000000000000
```

At the bottom of the terminal, there are keyboard shortcuts: [Add] [Delete], [Help], [Cancel], and [Next]. The Windows taskbar is visible at the bottom of the image.