Dynamically Provisioning Resources to Linux Virtual Servers

Friday, August 14, 2015: 11:15 AM - 12:15 PM,
Dolphin, Oceanic 4

Richard Young
Executive I.T. Specialist
IBM STG Lab Services

Agenda

1. The Value of Dynamically Provisioning and Deprovisioning Resources
2. Dynamically Adjusting Disk Storage Resources
3. Dynamically Adjusting Networking Resources
4. Dynamically Adjusting Memory Resources
5. Dynamically Adjusting CPU Resources
6. Automatically Adjusting Memory and CPU Resources

Agenda

1. **The Value of Dynamically Provisioning and Deprovisioning Resources**
2. Dynamically Adjusting Disk Storage Resources
3. Dynamically Adjusting Networking Resources
4. Dynamically Adjusting Memory Resources
5. Dynamically Adjusting CPU Resources
6. Automatically Adjusting Memory and CPU Resources

Dynamic Resource Configuration

- Helps to avoid Linux guest restarts and potential outage/downtime resource allocation changes
- Accommodate unplanned increases in application workload demands or application “enhancements” that consume more than expected resource
- It can allow for more efficient overall hypervisor operation (reduced operational overhead)
- Automated policy based reconfiguration is more responsive than manual adjustments.
- May provide assistance with upgrades by provisioning lower levels of resources both before a virtual server is in production and after it is removed from production.

Agenda

1. The Value of Dynamically Provisioning and Deprovisioning Resources
2. **Dynamically Adjusting Disk Storage Resources**
3. Dynamically Adjusting Networking Resources
4. Dynamically Adjusting Memory Resources
5. Dynamically Adjusting CPU Resources
6. Automatically Adjusting Memory and CPU Resources

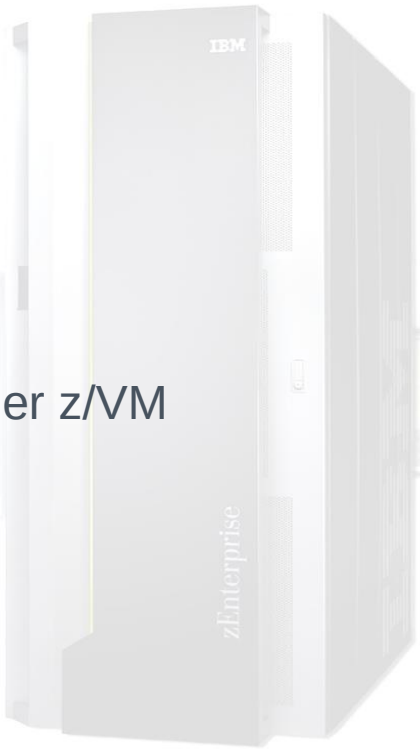
Dynamically Provisioning Resources

```
zlnx1:~ # lscss -t 3390
Device   Subchan.  DevType CU Type Use  PIM PAM POM  CHPIDs
----------------------------------------------------------------------
0.0.632a 0.0.04b6  3390/0c 3990/e9 yes  f0  f0  ff   16011700 00000000
0.0.632b 0.0.04b7  3390/0c 3990/e9      f0  f0  1f   16011700 00000000
0.0.632c 0.0.04b8  3390/0c 3990/e9      f0  f0  ff   16011700 00000000
zlnx1:~ # lschp | head -6
CHPID  Vary  Cfg.  Type  Cmg  Shared
==================================
0.00   1     1     1a    2    1
0.01   1     1     1d    2    1
0.02   1     1     0a    1    1
0.03   1     1     04    1    1
zlnx1:~ # chchp -v 0 0.01
Vary offline 0.01... done.
zlnx1:~ # lscss -t 3390
Device   Subchan.  DevType CU Type Use  PIM PAM POM  CHPIDs
----------------------------------------------------------------------
0.0.632a 0.0.04b6  3390/0c 3990/e9 yes  f0  f0  ff   16011700 00000000
0.0.632b 0.0.04b7  3390/0c 3990/e9      f0  f0  1f   16011700 00000000
0.0.632c 0.0.04b8  3390/0c 3990/e9      f0  f0  ff   16011700 00000000
zlnx1:~ # lschp | head -6
CHPID  Vary  Cfg.  Type  Cmg  Shared
==================================
0.00   1     1     1a    2    1
0.01   0     1     1d    2    1
0.02   1     1     0a    1    1
0.03   1     1     04    1    1
zlnx1:~ # chchp -v 1 0.01
Vary online 0.01... done.
zlnx1:~ #
```

- All (non-PCI) IO devices are attached via a defined channel
- In a native LPAR implementation you may need to change the channel (CHPID) state from Linux
- Be aware that lscss does not display the CHPID state
- Use chchp and lschp

Dynamically Adding Disk Resources

- Disk Storage Resource Types
  - ECKD
    - Full Volume
    - z/VM Minidisk
  - SCSI Luns
    - Via z/VM Emulated Device
    - Via Dedicated FCP Device
- All types can be dynamically added
- Can be performed whether in a native LPAR or under z/VM
- General Process
  - Add resource from hypervisor
  - Make new resource available
  - Bring virtual device online
  - Provision as usual



Dynamically Adding Disk Resources

```
zlnx1:~ # lscss -t 3390
Device   Subchan.  DevType CU Type Use  PIM PAM POM  CHPIDs
----------------------------------------------------------------------
0.0.632a 0.0.04b6  3390/0c 3990/e9 yes  f0  f0  ff   16011700 00000000
0.0.632b 0.0.04b7  3390/0c 3990/e9      f0  f0  1f   16011700 00000000
zlnx1:~ # cat /proc/cio_ignore
0.0.6000-0.0.6329
0.0.632c-0.0.63ff
zlnx1:~ # echo free 632c > /proc/cio_ignore
zlnx1:~ # lscss -t 3390
Device   Subchan.  DevType CU Type Use  PIM PAM POM  CHPIDs
----------------------------------------------------------------------
0.0.632a 0.0.04b6  3390/0c 3990/e9 yes  f0  f0  ff   16011700 00000000
0.0.632b 0.0.04b7  3390/0c 3990/e9      f0  f0  1f   16011700 00000000
0.0.632c 0.0.04b8  3390/0c 3990/e9      f0  f0  ff   16011700 00000000
zlnx1:~ # cat /proc/cio_ignore
0.0.6000-0.0.6329
0.0.632d-0.0.63ff
zlnx1:~ #
```

Dynamically Adding Disk Resources

```
zlnx1:~ # cat /etc/zipl.conf
# Modified by YaST2. Last modification on Tue Nov 20 21:00:21 CST 2012
[defaultboot]
defaultmenu = menu

###Don't change this comment - YaST2 identifier: Original name: linux###
[SLES11_SP2]
    image = /boot/image-3.0.13-0.27-default
    target = /boot/zipl
    ramdisk = /boot/initrd-3.0.13-0.27-default,0x2000000
    parameters = "root=/dev/disk/by-path/ccw-0.0.632a-part1  TERM=dumb cio_ignore=0.0.6000-0.0.6329,0.0.632c-63ff"
```

- The cio_ignore list is shown on the kernel parameters line of the zipl.conf
- Be sure to update it with newly (de)provisioned devices as you change the configuration of your system

Dynamically Adding Disk Resources

```
dirm for rgylxws8 amdisk 201 3390 autog 3338 LINUX  MR
DVHXMT1191I Your AMDISK request has been sent for processing to DIRMAINT
DVHXMT1191I at POKLBS1.
Ready; T=0.01/0.02 19:16:54
 DVHREQ2288I Your AMDISK request for RGYLXW   at * has been accepted.
 DVHSCU3541I Work unit 15191655 has been bu   and queued for processing.
 DVHSHN3541I Processing work unit 15191655 a  OUNG1 from POKLBS1,
 DVHSHN3541I notifying RYOUNG1 at POKLBS1, re  614 for RGYLXWS8 SSI
 DVHSHN3541I node *; to: AMDISK 0201 3390 AUTO  8 LINUX MR
 DVHBIU3450I The source for directory entry RG  has been updated.
 DVHBIU3424I The next ONLINE will take place im  ly.
 DVHDRC3451I The next ONLINE will take place via  bject directory.
 DVHRLA3891I Your DSATCTL request has been
 DVHBIU3428I Changes made to directory entr
 DVHBIU3428I online.
```

- DIRM add minidisk disk shown
- Could be full volume or partial volume
- Disk could be added via a dedicate as well
- If not using dirmaint, edit user direct and DIRECTXA

Dynamically Adding Disk Resources

```
RGYLXWS8:/ # lscss
Device   Subchan.  DevType CU Type Use   PIM PAM POM  CHPIDs
----------------------------------------------------------------------
0.0.1000 0.0.0000  1732/03 1731/03       80  80  ff   c4000000 00000000
0.0.1001 0.0.0001  1732/03 1731/03       80  80  ff   d1000000 00000000
0.0.1002 0.0.0002  1732/03 1731/03 yes   80  80  ff   c9000000 00000000
0.0.1003 0.0.0003  1732/03 1731/03 yes   80  80  ff   dd000000 00000000
0.0.0191 0.0.0004  3390/0c 3990/e9       80  80  ff   ff000000 00000000
0.0.0200 0.0.0006  3390/0c 3990/e9 yes   80  80  ff   ff000000 00000000
0.0.0192 0.0.0007  3390/0c 3990/e9       80  80  ff   ff000000 00000000
0.0.0009 0.0.0008  0000/00 3215/00 yes   80  80  ff   ff000000 00000000
0.0.0600 0.0.0009  1732/01 1731/01 yes   80  80  ff   00000000 00000000
0.0.0601 0.0.000a  1732/01 1731/01 yes   80  80  ff   00000000 00000000
0.0.0602 0.0.000b  1732/01 1731/01 yes   80  80  ff   00000000 00000000
0.0.000c 0.0.000c  0000/00 2540/00       80  80  ff   ff000000 00000000
0.0.000d 0.0.000d  0000/00 2540/00       80  80  ff   ff000000 00000000
0.0.000e 0.0.000e  0000/00 1403/00       80  80  ff   ff000000 00000000
0.0.0190 0.0.000f  3390/0c 3990/e9       80  80  ff   ff000000 00000000
0.0.019d 0.0.0010  3390/0c 3990/e9       80  80  ff   ff000000 00000000
0.0.019e 0.0.0011  3390/0c 3990/e9       80  80  ff   ff000000 00000000
RGYLXWS8:/ # vmcp q v dasd
DASD 0190 3390 P01RES R/O        214 CYL ON DASD  3F27 SUBCHANNEL = 000F
DASD 0191 3390 VM1US1 R/O        500 CYL ON DASD  3F10 SUBCHANNEL = 0004
DASD 0192 3390 LS3F18 R/W         50 CYL ON DASD  3F18 SUBCHANNEL = 0007
DASD 019D 3390 P01RES R/O        292 CYL ON DASD  3F27 SUBCHANNEL = 0010
DASD 019E 3390 P01RES R/O        500 CYL ON DASD  3F27 SUBCHANNEL = 0011
DASD 0200 3390 LS3F52 R/W      10015 CYL ON DASD  3F52 SUBCHANNEL = 0006
RGYLXWS8:/ #
```

- 201 minidisk still not available to Linux and not shown from a z/VM query virtual
- New storage must be `attached` or `linked` before it can be brought online

Dynamically Adding Disk Resources

```
RGYLXWS8:/ # vmcp link RGYLXWS8 201 201 MR
RGYLXWS8:/ # vmcp q v dasd
DASD 0190 3390 P01RES R/O       214 CYL ON DASD  3F27 SUBCHANNEL = 000F
DASD 0191 3390 VM1US1 R/O       500 CYL ON DASD  3F10 SUBCHANNEL = 0004
DASD 0192 3390 LS3F18 R/W        50 CYL ON DASD  3F18 SUBCHANNEL = 0007
DASD 019D 3390 P01RES R/O       292 CYL ON DASD  3F27 SUBCHANNEL = 0010
DASD 019E 3390 P01RES R/O       500 CYL ON DASD  3F27 SUBCHANNEL = 0011
DASD 0200 3390 LS3F52 R/W     10015 CYL ON DASD  3F52 SUBCHANNEL = 0006
DASD 0201 3390 LS3F18 R/W      3338 CYL ON DASD  3F18 SUBCHANNEL = 0005
RGYLXWS8:/ # lscss
Device   Subchan.  DevType CU Type Use  PIM PAM POM  CHPIDs
----------------------------------------------------------------------
0.0.1000 0.0.0000  1732/03 1731/03      80  80  ff   c4000000 00000000
0.0.1001 0.0.0001  1732/03 1731/03      80  80  ff   d1000000 00000000
0.0.1002 0.0.0002  1732/03 1731/03 yes  80  80  ff   c9000000 00000000
0.0.1003 0.0.0003  1732/03 1731/03 yes  80  80  ff   dd000000 00000000
0.0.0191 0.0.0004  3390/0c 3990/e9      80  80  ff   ff000000 00000000
0.0.0201 0.0.0005  3390/0c 3990/e9      80  80  ff   ff000000 00000000
0.0.0200 0.0.0006  3390/0c 3990/e9 yes  80  80  ff   ff000000 00000000
0.0.0192 0.0.0007  3390/0c 3990/e9      80  80  ff   ff000000 00000000
0.0.0009 0.0.0008  0000/00 3215/00 yes  80  80  ff   ff000000 00000000
0.0.0600 0.0.0009  1732/01 1731/01 yes  80  80  ff   00000000 00000000
0.0.0601 0.0.000a  1732/01 1731/01 yes  80  80  ff   00000000 00000000
0.0.0602 0.0.000b  1732/01 1731/01 yes  80  80  ff   00000000 00000000
0.0.000c 0.0.000c  0000/00 2540/00      80  80  ff   ff000000 00000000
0.0.000d 0.0.000d  0000/00 2540/00      80  80  ff   ff000000 00000000
0.0.000e 0.0.000e  0000/00 1403/00      80  80  ff   ff000000 00000000
0.0.0190 0.0.000f  3390/0c 3990/e9      80  80  ff   ff000000 00000000
0.0.019d 0.0.0010  3390/0c 3990/e9      80  80  ff   ff000000 00000000
0.0.019e 0.0.0011  3390/0c 3990/e9      80  80  ff   ff000000 00000000
RGYLXWS8:/ # chccwdev -e 201
Setting device 0.0.0201 online
Done
```

A z/VM "link" makes device available.

Can be performed from Linux via 'vmcp"

Must still be brought online via "chccwdev"

Dynamically Adding Disk Resources

```
RGYLXWS8:/ # lsdasd
Bus-ID     Status      Name      Device  Type  BlkSz  Size      Blocks
==============================================================================
0.0.0200   active      dasda     94:0    ECKD  4096   7041MB    1802700
0.0.0201   active      dasdb     94:4    ECKD  4096   2347MB    600840

RGYLXWS8:/ # dasdfmt -b 4096  -f /dev/dasdb
Drive Geometry: 3338 Cylinders * 15 Heads =  50070 Tracks
I am going to format the device /dev/dasdb in the following way:
   Device number of device : 0x201
   Labelling device        : yes
   Disk label              : VOL1
   Disk identifier         : 0X0201
   Extent start (trk no)   : 0
   Extent end (trk no)     : 50069
   Compatible Disk Layout  : yes
   Blocksize               : 4096
--->> ATTENTION! <<---
All data of that device will be lost.
Type "yes" to continue, no will leave the disk untouched:
```

Dynamically Adding Disk Resources

```
RGYLXWS8:/ # fdasd -a /dev/dasdb
reading volume label ..: VOL1
reading vtoc ..........: ok
auto-creating one partition for the whole disk...
writing volume label...
writing VTOC...
rereading partition table...
RGYLXWS8:/ #
```

- Disk storage has been dynamically brought online, formatted, and partitioned
- Put file system on new device
  - `mkfs -t ext3 -c /dev/dasdb1`
- You could now add to a volume group and LVM to dynamically expand a filesystem without bring the Linux system down
  - `pvcreate /dev/dasdb1`
  - `vgextend VG00 /dev/dasdb1`
  - `lvextend -L+1G /dev/VG00/LV01 ; add one more GB to LV`
  - `ext2online /dev/VG00/LV01`
  - `resize2fs /dev/VG00/LV01`

Dynamically adding a SCSI LUN

```
attach 8a2a to rgylxsp2 as 1000
FCP 8A2A ATTACHED TO RGYLXSP2 1000
Ready; T=0.01/0.01 11:23:47
attach 8b2a to rgylxsp2 as 1001
FCP 8B2A ATTACHED TO RGYLXSP2 1001
Ready; T=0.01/0.01 11:23:55
attach 8c2a to rgylxsp2 as 1002
FCP 8C2A ATTACHED TO RGYLXSP2 1002
Ready; T=0.01/0.01 11:23:59
attach 8d2a to rgylxsp2 as 1003
FCP 8D2A ATTACHED TO RGYLXSP2 1003
Ready; T=0.01/0.01 11:24:04
```

- Dynamically making the FCP devices available to the guest virtual server
- In an NPIV configuration each device will represent a unique WWPN
- Each WWPN must be zoned to the correct storage resource

Dynamically adding a SCSI LUN

```
rgylxsp2:~ # lszfcp
Error: No fcp devices found.
rgylxsp2:~ # lscss
Device   Subchan.  DevType CU Type Use  PIM PAM POM  CHPIDs
----------------------------------------------------------------------
0.0.0600 0.0.0000  1732/01 1731/01 yes  80  80  ff   00000000 00000000
0.0.0601 0.0.0001  1732/01 1731/01 yes  80  80  ff   00000000 00000000
0.0.0602 0.0.0002  1732/01 1731/01 yes  80  80  ff   00000000 00000000
0.0.0700 0.0.0003  1732/01 1731/01 yes  80  80  ff   01000000 00000000
0.0.0701 0.0.0004  1732/01 1731/01 yes  80  80  ff   01000000 00000000
0.0.0702 0.0.0005  1732/01 1731/01 yes  80  80  ff   01000000 00000000
0.0.0800 0.0.0006  1732/01 1731/01      80  80  ff   02000000 00000000
0.0.0801 0.0.0007  1732/01 1731/01      80  80  ff   02000000 00000000
0.0.0802 0.0.0008  1732/01 1731/01      80  80  ff   02000000 00000000
0.0.0191 0.0.0009  3390/0c 3990/e9      80  80  ff   ff000000 00000000
0.0.0399 0.0.000a  3390/0c 3990/e9 yes  80  80  ff   ff000000 00000000
0.0.0200 0.0.000b  3390/0c 3990/e9 yes  80  80  ff   ff000000 00000000
0.0.0192 0.0.000c  3390/0c 3990/e9      80  80  ff   ff000000 00000000
0.0.0009 0.0.000d  0000/00 3215/00 yes  80  80  ff   ff000000 00000000
0.0.000c 0.0.000e  0000/00 2540/00      80  80  ff   ff000000 00000000
0.0.000d 0.0.000f  0000/00 2540/00      80  80  ff   ff000000 00000000
0.0.000e 0.0.0010  0000/00 1403/00      80  80  ff   ff000000 00000000
0.0.0190 0.0.0011  3390/0c 3990/e9      80  80  ff   ff000000 00000000
0.0.019d 0.0.0012  3390/0c 3990/e9      80  80  ff   ff000000 00000000
0.0.019e 0.0.0013  3390/0c 3990/e9      80  80  ff   ff000000 00000000
0.0.1000 0.0.0014  1732/03 1731/03      80  80  ff   c4000000 00000000
0.0.1001 0.0.0015  1732/03 1731/03      80  80  ff   d1000000 00000000
0.0.1002 0.0.0016  1732/03 1731/03      80  80  ff   c9000000 00000000
0.0.1003 0.0.0017  1732/03 1731/03      80  80  ff   dd000000 00000000
rgylxsp2:~ #
```

The new FCP devices are available but must be brought online to Linux

Dynamically adding a SCSI LUN

```
0.0.1000 0.0.0014  1732/03 1731/03       80   80   ff   c4000000 00000000
0.0.1001 0.0.0015  1732/03 1731/03       80   80   ff   d1000000 00000000
0.0.1002 0.0.0016  1732/03 1731/03       80   80   ff   c9000000 00000000
0.0.1003 0.0.0017  1732/03 1731/03       80   80   ff   dd000000 00000000
rgylxsp2:~ # chccwdev -e 1000-1003
Setting device 0.0.1000 online
Done
Setting device 0.0.1001 online
Done
Setting device 0.0.1002 online
Done
Setting device 0.0.1003 online
Done
rgylxsp2:~ # zfcp_disk_configure 0.0.1002  0x500507630908856b  0x4003402A00000000  1
No configuration file for adapter 0.0.1002
Configuring FCP disk 500507630908856b:4003402a00000000
rgylxsp2:~ # lsluns -a
adapter = 0.0.1002
        port = 0x500507630908856b
                lun = 0x4003402a00000000        /dev/sg0        Disk    IBM:2107900
rgylxsp2:~ #
```

Dynamically adding a SCSI LUN

- zfcp_ping & zfcp_show are new diagnostic tools when can be helpful when need to configure FCP attached storage
- SLES 11 SP3/RHEL 6.4
- libzfcphbaapi0 package

```
rgylxsp3:~ # zfcp_show 0.0.1000 | more

Interconnect Element Name       0x100000051e364632
Interconnect Element Domain ID  036
Interconnect Element Type       Switch
Interconnect Element Ports      256
        ICE Port 000  Online
                Attached Port [WWPN/ID] 0x50050763060005b2 / 0x240000 [N_Port]
        ICE Port 001  Online
                Attached Port [WWPN/ID] 0x50050763060045b2 / 0x240100 [N_Port]
        ICE Port 002  Online
                Attached Port [WWPN/ID] 0x500507630310007f / 0x240200 [N_Port]
```

```
rgylxsp3:~ # zfcp_ping 0x500507630903856b
Sending PNG from BUS_ID=0.0.1000 speed=4 GBit/s
        echo received from WWPN (0x500507630903856b) tok=0 time=27.210 ms
        echo received from WWPN (0x500507630903856b) tok=1 time=4.485 ms
        echo received from WWPN (0x500507630903856b) tok=2 time=5.468 ms

---------- ping statistics -----------
min/avg/max = 4.485/12.388/27.210 ms
--------------------------------------
```

Dynamically adding a SCSI LUN

```
rgylxsp2:/etc/udev # cd rules.d/
rgylxsp2:/etc/udev/rules.d # ls
40-alsa.rules                 52-xpram.rules          79-yast2-drivers.rules
51-dasd-0.0.0200.rules        57-osasnmpd.rules       81-mount.rules
51-packagekit-firmware.rules  59-dasd.rules           81-mptctl.rules
51-qeth-0.0.0600.rules        60-readahead.rules      85-usb_autosuspend_devices.rules
51-qeth-0.0.0700.rules        70-kpartx.rules         85-usb_elotouch_wakeup.rules
51-zfcp-0.0.1002.rules        70-persistent-net.rules 99-iwlwifi-led.rules
51-zfcp-0.0.1003.rules        71-multipath.rules      99-pcsc_lite.rules
52-hw_random.rules            77-network.rules
rgylxsp2:/etc/udev/rules.d #
```

- Confirm the udev entries were made so the definitions are persistent.
- Also make sure your z/VM dedicates exist in the user directory so the devices are available after a restart of the guest virtual server

Dynamically adding a SCSI LUN

```
rgylxsp2:~ # fdisk /dev/sda

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p1
Partition number (1-4, default 1): 1
First sector (2048-41943039, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-41943039, default 41943039):
Using default value 41943039

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
rgylxsp2:~ # pvcreate /dev/sda1
  Physical volume "/dev/sda1" successfully created
rgylxsp2:~ #
```

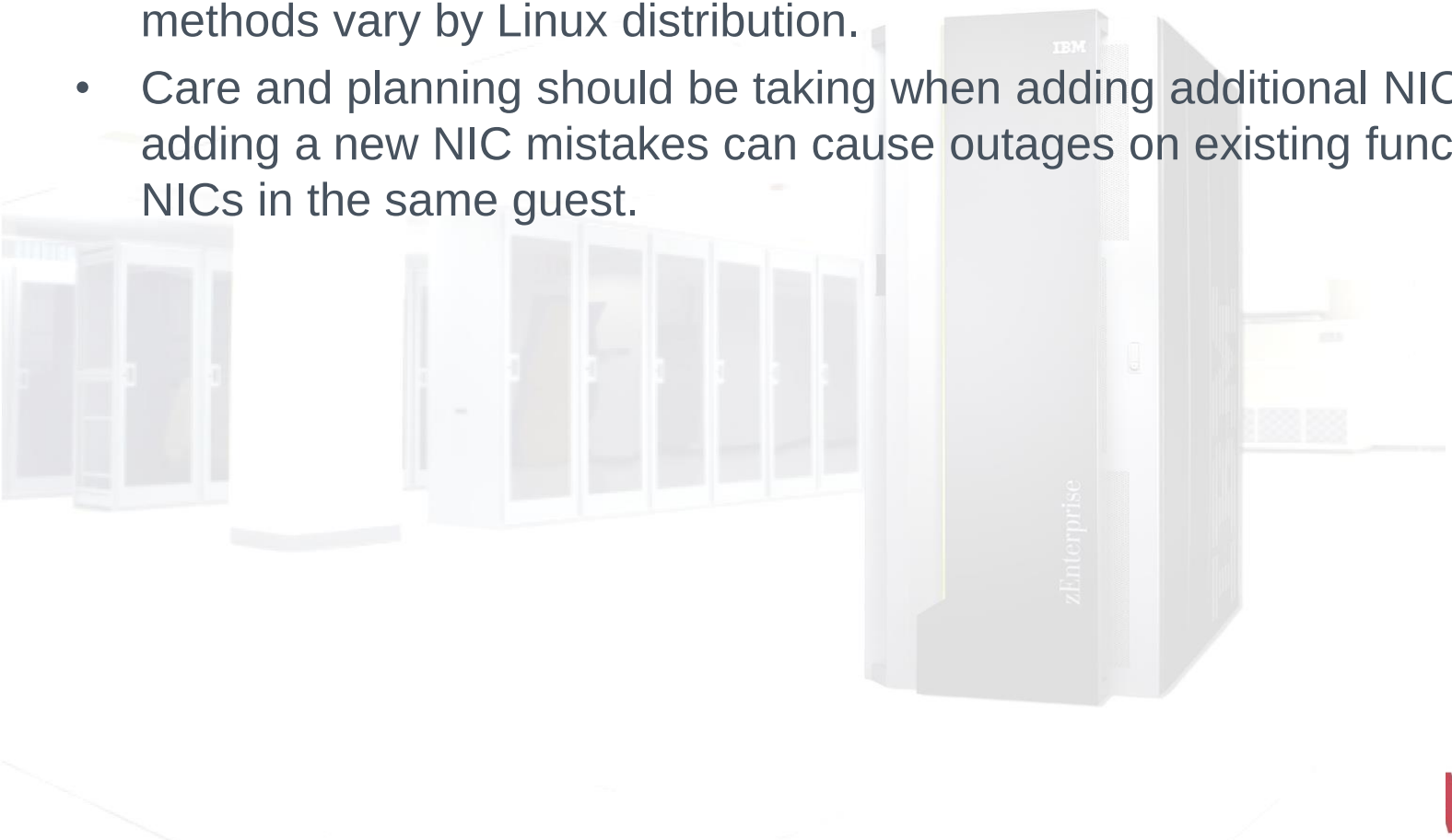
- At this point you can add the device as you normally would
- Define to the multipather, partition, and place a file system on the device or add to a logical volume

Agenda

1. The Value of Dynamically Provisioning and Deprovisioning Resources
2. Dynamically Adjusting Disk Storage Resources
3. **Dynamically Adjusting Networking Resources**
4. Dynamically Adjusting Memory Resources
5. Dynamically Adjusting CPU Resources
6. Automatically Adjusting Memory and CPU Resources

Dynamically Adding Network Interfaces

- Much like dynamically adding disk resources a directory alone does not make the NIC available to Linux.
- Once the NIC is defined there are multiple ways to configure it and some methods vary by Linux distribution.
- Care and planning should be taking when adding additional NIC. When adding a new NIC mistakes can cause outages on existing functioning NICs in the same guest.



Dynamically Adding Network Interfaces

```
RGYLXWS8:~ # lsqeth
Device name                   : eth0
--------------------------------------------
        card_type             : GuestLAN QDIO
        cdev0                 : 0.0.0600
        cdev1                 : 0.0.0601
        cdev2                 : 0.0.0602
        chpid                 : 00
        online                : 1
        portname              : NET172A
        portno                : 0
        state                 : UP (LAN ONLINE)
        priority_queueing     : always queue 2
        buffer_count          : 64
        layer2                : 1
        isolation             : none

RGYLXWS8:~ # znetconf -c
Device IDs                  Type    Card Type      CHPID Drv. Name         State
------------------------------------------------------------------------------
0.0.0600,0.0.0601,0.0.0602 1731/01 GuestLAN QDIO     00 qeth eth0         online
RGYLXWS8:~ #
```

- This system that has only one NIC and a second NIC will be added

Dynamically Adding Network Interfaces

- New NIC added to the zVM user directory
  - Virtual device 700
  - Type QDIO
  - VSWITCH NET172B

```
dirm for rgylxws8 NICDEF 0700 TYPE QDIO DEV 3 LAN SYSTEM NET172B
DVHXMT1191I Your NICDEF request has been sent for processing to DIRMAINT
DVHXMT1191I at POKLBS1.
Ready; T=0.01/0.02 01:43:35
 DVHREQ2288I Your NICDEF request for RGYLXWS8 at * has been accepted.
 DVHBIU3450I The source for directory entry RGYLXWS8 has been updated.
 DVHBIU3424I The next ONLINE will take place immediately.
 DVHDRC3451I The next ONLINE will take place via delta object directory.
 DVHRLA3891I Your DSATCTL request has been relayed for processing.
 DVHBIU3428I Changes made to directory entry RGYLXWS8 have been placed
 DVHBIU3428I online.
 DVHREQ2289I Your NICDEF request for RGYLXWS8 at * has completed; with RC
 DVHREQ2289I = 0.
 DVHREQ2288I Your DSATCTL request for DIRMAINT at
 DVHREQ2288I * has been accepted.
 DVHREQ2289I Your DSATCTL request for DIRMAINT at
 DVHREQ2289I * has completed; with RC = 0.
```

Dynamically Adding Network Interfaces

- “DEFINE NIC” issued to make the new virtual NIC available to the guest
- Since it was already defined in the user directory it automatically coupled to its virtual switch
- znetconf now shows the new virtual NIC
- Since the NIC is yet unconfigured, it is still offline

```
RGYLXWS8:~ # vmcp define nic  0700 TYPE QDIO DEV 3
NIC 0700 is created; devices 0700-0702 defined
RGYLXWS8:~ # vmcp couple 700 to system net172b
HCPCPL2788E NIC 0700 not connected; already connected to VSWITCH SYSTEM NET172B
Error: non-zero CP response for command 'COUPLE 700 TO SYSTEM NET172B': #2788
RGYLXWS8:~ # znetconf -u
Scanning for network devices...
Device IDs                Type    Card Type      CHPID Drv.
------------------------------------------------------------
0.0.0700,0.0.0701,0.0.0702 1731/01 OSA (QDIO)        01 qeth
RGYLXWS8:~ # znetconf -c
Device IDs                Type    Card Type      CHPID Drv. Name        State
------------------------------------------------------------------------------
0.0.0600,0.0.0601,0.0.0602 1731/01 GuestLAN QDIO     00 qeth eth0        online
RGYLXWS8:~ #
```

Dynamically Adding Network Interfaces

- We could use tools such as Yast, netconfig, or redhat-config-network to configure the interface, but we will use znetconf from s390-tools
- znetconf allows you to configure many different possible attributes of the QDIO device
- Note: znetconf does not create a udev entry
- After executing znetconf the device (not the interface) will be online

```
RGYLXWS8:~ # znetconf -a 0700 -o layer2=1
Scanning for network devices...
Successfully configured device 0.0.0700 (eth1)
```

Dynamically Adding Network Interfaces

- To bring the network interface online you need an ifcfg-eth*x* script
- If you copy an existing file (such as ifcfg-eth0) you should have only a few changes to make
  - IPADDR, NETMASK, NETWORK
  - _nm_name, BROADCAST
- It is highly recommended to put a udev entry in place (/etc/udev/rules.d) so you have a persistent configuration across reboots

```
BOOTPROTO="static"
UNIQUE=""
STARTMODE="onboot"
IPADDR="172.110.100.38“
NETMASK="255.255.255.0"
NETWORK="172.110.100.0"
BROADCAST="172.110.100.255"
_nm_name='qeth-bus-ccw-0.0.0700‘
```

Dynamically Adding Network Interfaces

- You can activate your new configuration with `rcnetwork restart`
- If your new interface configuration breaks your existing network, logon to the 3270 console for the guest and move the ifcfg-eth*x* script to another directory and reissue your `rcnetwork restart` command.

Modifying Attributes of an OSA

- Can be performed without restarting the server
- Network interface must be taken offline in many cases
- Don’t take offline with chccwdev
- Utilize /sys filesystem interface to take offline/online
- Details documented in the Linux Device Drivers, Features, and Commands manual on DeveloperWorks (See link at end of presentation)

Modifying Attributes

```
rgylxsp2:~ # lsqeth
Device name                    : eth0
-------------------------------------------
        card_type              : GuestLAN QDIO
        cdev0                  : 0.0.0600
        cdev1                  : 0.0.0601
        cdev2                  : 0.0.0602
        chpid                  : 00
        online                 : 1
        portname               : dontcare
        portno                 : 0
        state                  : UP (LAN ONLINE)
        priority_queueing      : always queue 2
        buffer_count           : 64
        layer2                 : 1
        isolation              : none

Device name                    : eth1
-------------------------------------------
        card_type              : GuestLAN QDIO
        cdev0                  : 0.0.0700
        cdev1                  : 0.0.0701
        cdev2                  : 0.0.0702
        chpid                  : 01
        online                 : 1
        portname               : 0
        portno                 : 0
        state                  : UP (LAN ONLINE)
        priority_queueing      : always queue 2
        buffer_count           : 64
        layer2                 : 1
        isolation              : none
```

- This system has two network interfaces
- The buffer count on one of them will be increased
- The system will not be brought down
- Only the interface being changed will be stopped

Modifying Attributes

- The specific device is found under /sys/bus./ccwgroup/drivers/qeth
- The eth1 interface is stopped
- The attempt to take device 700 offline fails because it must be done via the /sys filesystem

```
rgylxsp2:~ # cd /sys/bus/ccwgroup/drivers/qeth/
rgylxsp2:/qeth # cd 0.0.0700/
rgylxsp2:/0.0.0700 # cat buffer_count
64
rgylxsp2:/0.0.0700 # ifconfig eth1 down
rgylxsp2:/0.0.0700 # chccwdev -d 700
Setting device 0.0.0700 offline
Failed (Invalid argument)
```

Modifying Attributes

- At the top you can see the buffer_count can not be changed while the device is online
- “echo” a 1 or 0 in the “online” file to control the device state
- This same process can be used to change other attribute, but some such as layer2, may need changes to the udev configuration to be made permanent

```
rgylxsp2:/0.0.0700 # echo 128 > buffer_count
-bash: echo: write error: Operation not permitted
rgylxsp2:/0.0.0700 # ls
blkt          cdev0  chpid    if_name     layer2  performance_stats  power
state      ungroup
buffer_count  cdev1  driver   inbuf_size  net     portname
priority_queueing  subsystem
card_type     cdev2  hw_trap  isolation   online  portno             recover
uevent
rgylxsp2:/0.0.0700 # echo 0 > online
rgylxsp2:/0.0.0700 # echo 128 > buffer_count
rgylxsp2:/0.0.0700 # echo 1 > online
```

Modifying Attributes

```
rgylxsp2:/0.0.0700 # cat buffer_count
128

rgylxsp2:/0.0.0700 # lsqeth
Device name                    : eth1
-------------------------------------------------
        card_type              : GuestLAN QDIO
        cdev0                  : 0.0.0700
        cdev1                  : 0.0.0701
        cdev2                  : 0.0.0702
        chpid                  : 01
        online                 : 1
        portname               : 0
        portno                 : 0
        state                  : SOFTSETUP
        priority_queueing      : always queue 2
        buffer_count           : 128
        layer2                 : 1
        isolation              : none
```

- Only device 0700 shown, device 0600 omitted for readability
- buffer_count has been changed to the maximum
- At this point the “interface” eth1 just needs to be brought up

Modifying Attributes - ethtool

- “ethtool” can dynamically set or query ethernet interface attributes and statistics
- It can be used to set generic receive offload (GRO), TCP segmentation offload (TSO), checksum operations and other options...
- ethtool –k queries “offload” settings

```
zlnx1:/etc/sysconfig/network # ethtool -k eth0
Offload parameters for eth0:
rx-checksumming: on
tx-checksumming: off
scatter-gather: off
tcp-segmentation-offload: off
udp-fragmentation-offload: off
generic-segmentation-offload: off
generic-receive-offload: on
large-receive-offload: off
rx-vlan-offload: on
tx-vlan-offload: on
ntuple-filters: off
receive-hashing: off
```

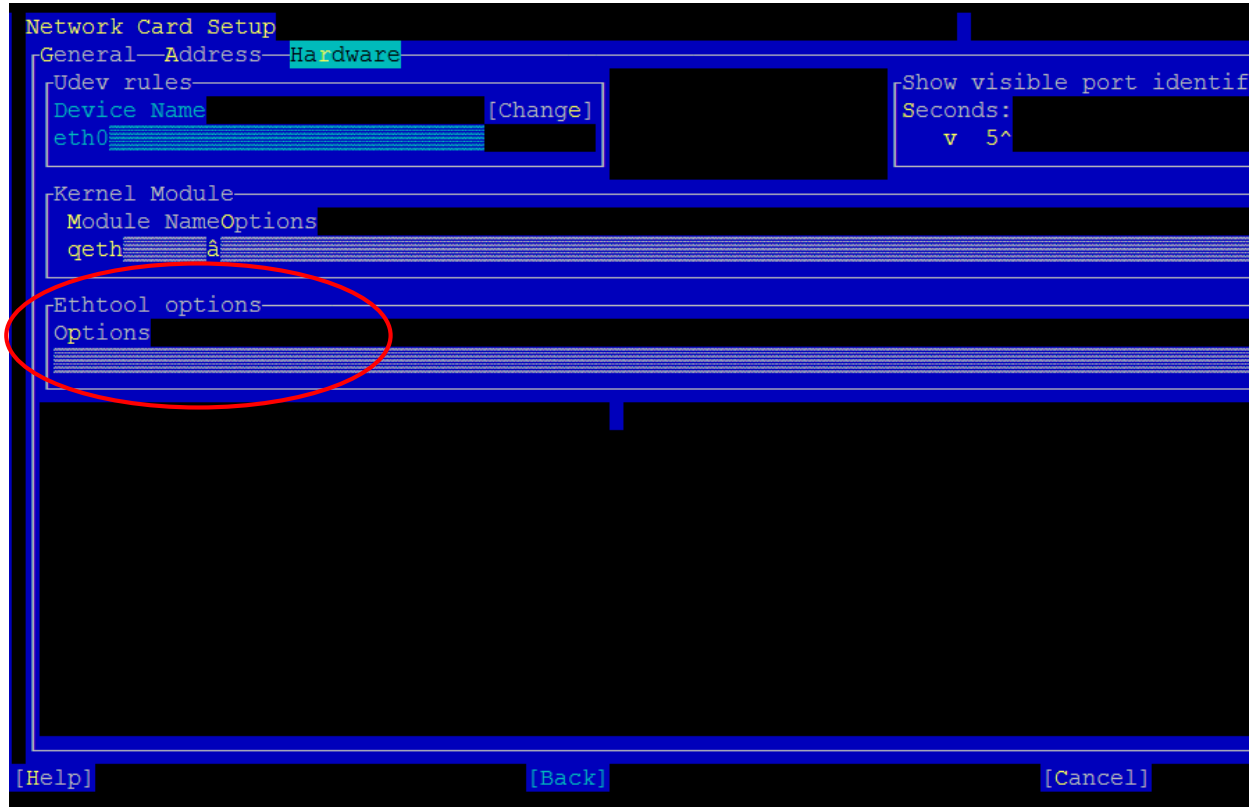
Modifying Attributes - ethtool

- ethtool –K can set offload options
- TCP segmentation offload (TSO) applies only to Layer 3 interfaces
- It can be set dynamically as shown
- GRO can also be controlled but as of kernel 2.6.39 GRO and rx-checksumming are on by default

```
zlnx1:/etc/sysconfig/network # ethtool -K eth0 tx on sg on tso on
zlnx1:/etc/sysconfig/network # ethtool -k eth0
Offload parameters for eth0:
rx-checksumming: on
tx-checksumming: on
scatter-gather: on
tcp-segmentation-offload: on
udp-fragmentation-offload: off
generic-segmentation-offload: on
generic-receive-offload: on
large-receive-offload: off
rx-vlan-offload: on
tx-vlan-offload: on
ntuple-filters: off
receive-hashing: off
```

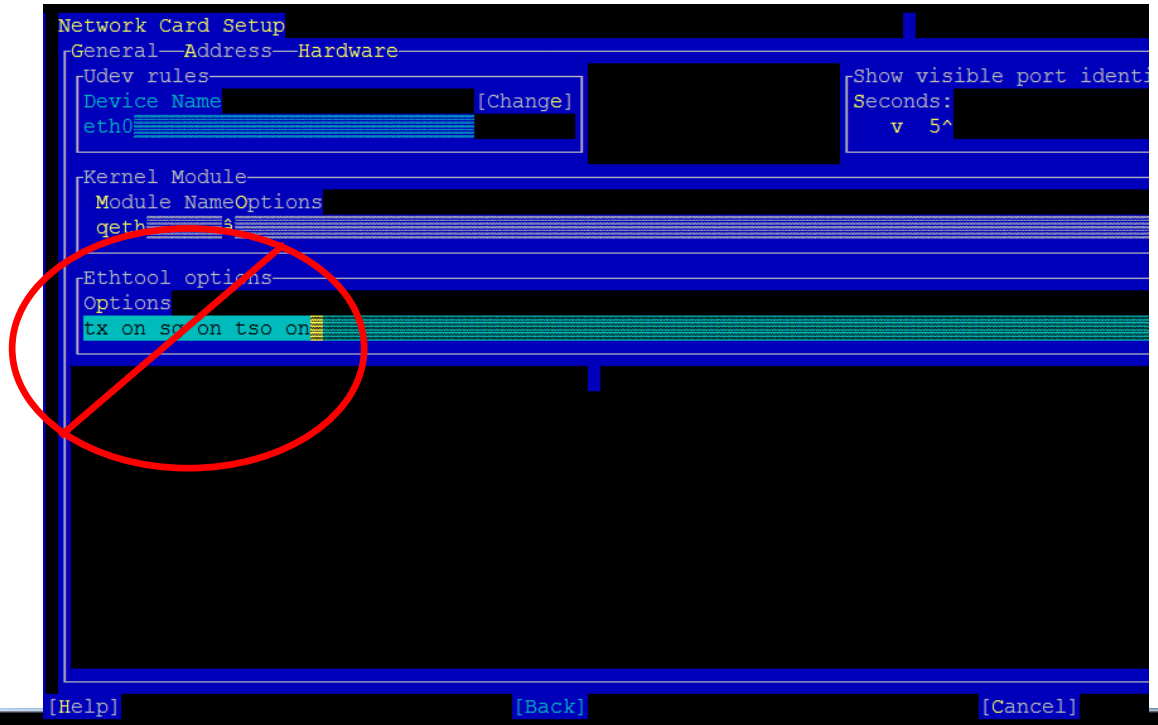
Modifying Attributes - ethtool

- Methods by which ethtool changes are made permanent vary by distribution and code levels
- For example, Yast has a place to code ethtool options, but only for ethtool –s and NOT the “-K” offload commands
- ethtool “offload” commands could be made permanent in other ways..
  - /etc/init.d/ scripts



Modifying Attributes - ethtool

- Attempting to code the ethtool offload commands results in an error during interface activation
- Only ethtool –s options should be coded here



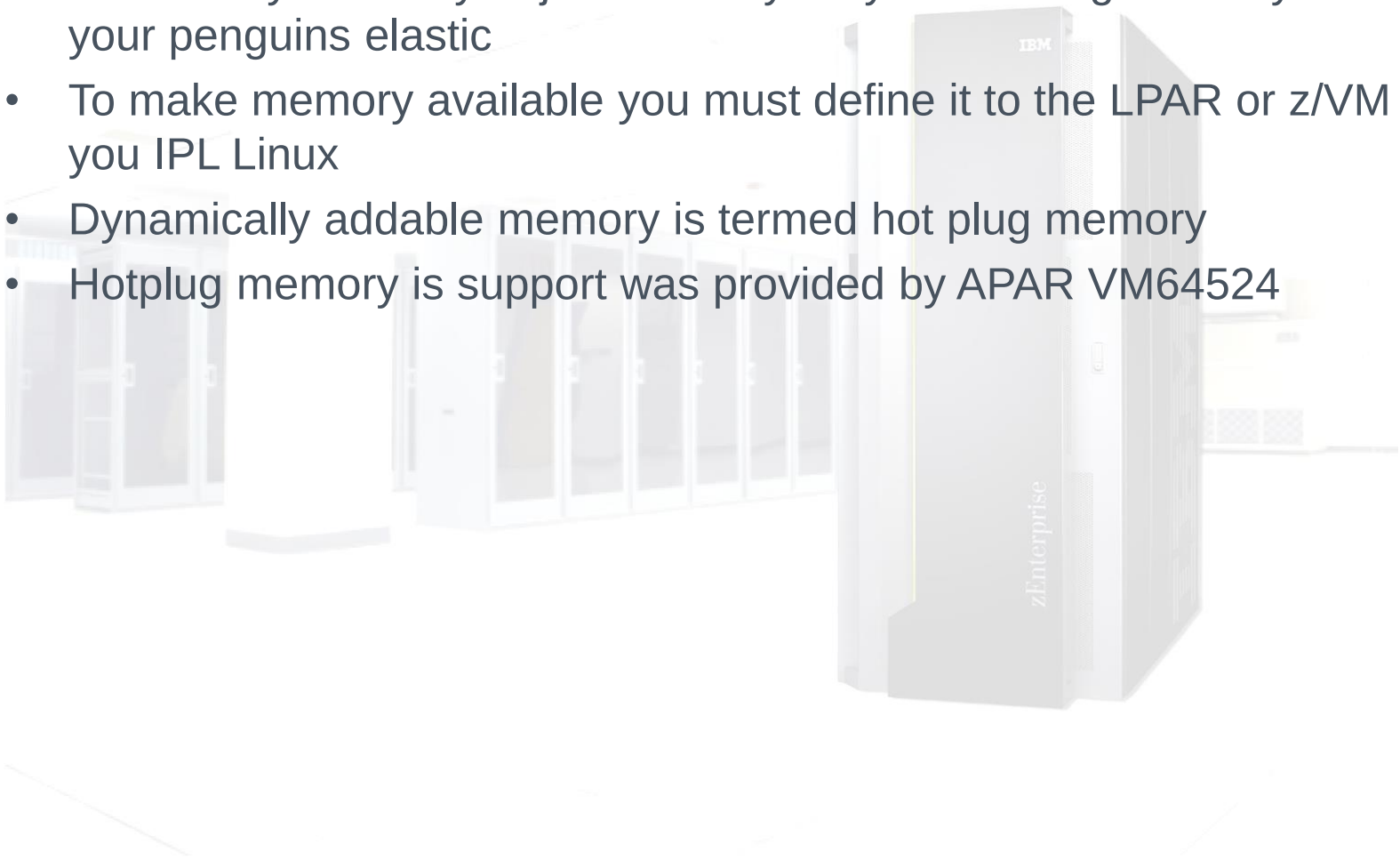

```
zlnx1 ifup:
zlnx1 ifup:
zlnx1 ifup: IP address: 127.0.0.2/8
zlnx1 ifup:
zlnx1 ifup:     eth0      name: OSA Express Network card (0.0.0842)
zlnx1 ifup: Error while executing '/sbin/ethtool -s eth0 tx on sg on tso on': [1] ethtool: bad command line argument(s)
zlnx1 ifup: For more information run ethtool -h
zlnx1 ifup:     eth0
zlnx1 ifup: IP address: 10.104.100.111/24
zlnx1 ifup:
zlnx1 kernel: klogd 1.4.1, log source = /proc/kmsg started.
zlnx1 kernel: type=1400 audit(1381946894.998:2): apparmor="STATUS" operation="profile_load" name="/bin/ping" pid=955 comm
```

Agenda

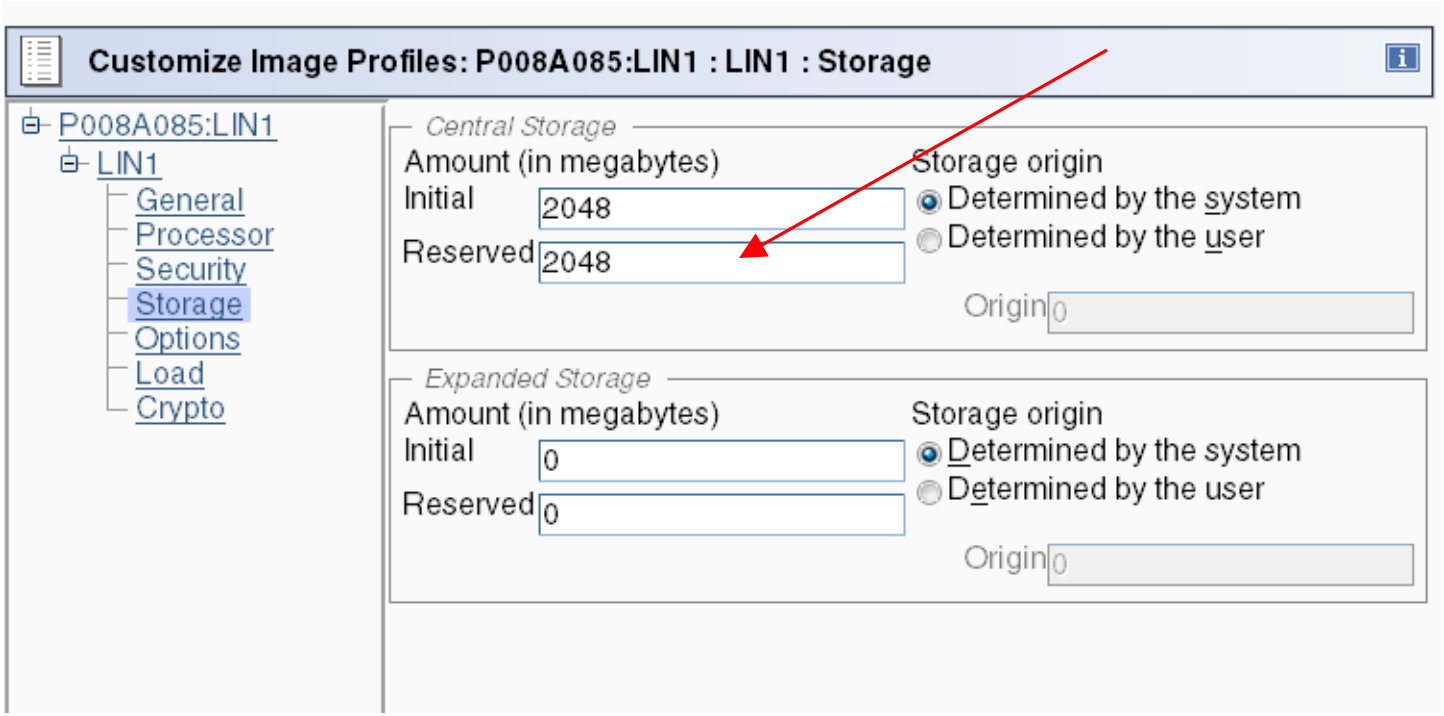
1. The Value of Dynamically Provisioning and Deprovisioning Resources
2. Dynamically Adjusting Disk Storage Resources
3. Dynamically Adjusting Networking Resources
4. **Dynamically Adjusting Memory Resources**
5. Dynamically Adjusting CPU Resources
6. Automatically Adjusting Memory and CPU Resources

Dynamically Provisioning Memory Resources

- You can dynamically adjust memory for your running Linux system making your penguins elastic
- To make memory available you must define it to the LPAR or z/VM before you IPL Linux
- Dynamically addable memory is termed hot plug memory
- Hotplug memory is support was provided by APAR VM64524



Dynamically Provisioning Memory Resources



- Defining “Reserved” storage to the LPAR will allow you to dynamically add memory to a running Linux server running natively in a partition

Dynamically Provisioning Memory Resources

```
 RGYLX0E4 DIRECT   A0  F 80  Trunc=72 Size=20 Line=0 Col=1 Alt=0

===== * * * Top of File * * *
===== USER RGYLX0E4 1GYLX0E4 1G 2G G
=====    INCLUDE LINDFLT
=====    CPU 00
=====    CPU 01
=====    CRYPTO    APVIRTUAL
=====    IUCV ANY
=====    LOADDEV PORTNAME 5005076306138411
=====    LOADDEV LUN 4011402E00000000
=====    MACHINE ESA 4
=====    OPTION APPLMON MAXCONN 128
=====    DEDICATE 1000 3B46
=====    DEDICATE 2000 3B66
=====    DEDICATE 4000 1FF6
=====    NICDEF 0700 TYPE QDIO DEV 3 LAN SYSTEM NET172A
```

- This z/VM guest has a user directory entry with 1GB of initial memory and 2 GB of maximum memory
- In z/VM, changing the memory size or configuration of a guest causes a storage reset (all storage is cleared)
- If you are running Linux natively in an LPAR without z/VM, you would use reserved storage in the LPAR definition to set aside potential additional memory
- In z/VM, define the memory to be dynamically enabled as “standby” storage

Dynamically Provisioning Memory Resources

```
21:15:04 Ready; T=0.01/0.02 21:15:04
21:15:14 define storage 1G standby 1G
21:15:14 00: STORAGE = 1G MAX = 2G INC = 2M STANDBY = 1G RESERVED = 0
21:15:14 00: Storage cleared - system reset.
```

- "**`DEFINE STORAGE 1G STANDBY 1G`**" issued for this guest
- Issuing a DEFINE STORAGE command causes storage to be cleared
- Anything running at the time of the reset will be immediately terminated without running any shutdown procedures
- This means if you issued this command from a CMS EXEC, CMS is no longer running because storage has been cleared.

Dynamically Provisioning Memory Resources

- Example COMMAND statement in User Directory

```
USER RGYLX0E1 RGYLX0E1 3G 8G G
   INCLUDE LINDFLT
   COMMAND DEFINE STORAGE 2G STANDBY 2G
   CPU 00
   CRYPTO    APVIRTUAL
   IUCV ANY
   OPTION MAXCONN 128
   LINK RGYLXMNT 0191 0191 RR
   MDISK 0200 3390 1 END LS20C8 MR READ WRITE MULTIPLE
```

Dynamically Provisioning Memory Resources

```
ICH70001I RGYLX0E1 LAST ACCESS AT 20:23:51 ON THURSDAY, SEPTEMBER 22, 2011
00: NIC 0600 is created; devices 0600-0602 defined
00: z/VM Version 6 Release 1.0, Service Level 1002 (64-bit),
00: built on IBM Virtualization Technology
00: There is no logmsg data
00: FILES: 0001 RDR,   NO PRT,   NO PUN
00: LOGON AT 20:26:20 EDT THURSDAY 09/22/11
00: STORAGE = 2G MAX = 8G INC = 4M STANDBY = 2G RESERVED = 0
00: Storage cleared - system reset.
z/VM V6.1.0    2010-10-15 11:49
DMSACP723I A (191) R/O
20:26:20 DIAG swap disk defined at virtual address 101 (64989 4K pages of swap
   space)
20:26:20  Detected interactive logon
20:26:20  MUST BE LOGGING ON FROM TERMINAL
```

Dynamically Provisioning Memory Resources

```
rgylx0e4:~ # cat /proc/meminfo
MemTotal:        2051920 kB
MemFree:         1877596 kB
Buffers:           10304 kB
Cached:            51160 kB
SwapCached:            0 kB
Active:            29788 kB
Inactive:          54872 kB
Active(anon):      23212 kB
Inactive(anon):      120 kB
Active(file):       6576 kB
Inactive(file):    54752 kB
Unevictable:           0 kB
Mlocked:               0 kB
SwapTotal:             0 kB
```

- After IPLing Linux in this guest, observe via /proc/meminfo that approximately 2GB of memory is available
- The “standby” memory is not reported by /proc/meminfo
- The /sys file system however has an awareness of this “standby” or “hot plug” memory
- With s390-tools, **lsmem** can be used to report this information and **chmem** to bring storage elements online or offline

Dynamically Provisioning Memory Resources…

```
rgylx0e4:~ # lsmem
Address Range                              Size (MB)   State    Removable  Device
==========================================================================================
```

Core Memory

```
0x0000000000000000-0x000000000fffffff        256   online   no         0-63
0x0000000010000000-0x000000006fffffff       1536   online   yes        64-447
0x0000000070000000-0x000000007fffffff        256   online   no         448-511
```

```
0x0000000080000000-0x00000000ffffffff       2048   offline  -          512-1023
```

Hotplug Memory

```
Memory device size  : 4 MB
Memory block size   : 256 MB
Total online memory : 2048 MB
Total offline memory: 2048 MB
```

- The lsmem command is an easy way to view core and hotplug memory status
- The display looks and works the same whether running under z/VM or running natively

Dynamically Provisioning Memory Resources

```
rgylx0e4:~ # chmem -e 2g
rgylx0e4:~ # lsmem
Address Range                              Size (MB)   State    Removable  Device
==========================================================================================
0x0000000000000000-0x000000000fffffff         256   online   no         0-63
0x0000000010000000-0x000000006fffffff        1536   online   yes        64-447
0x0000000070000000-0x000000007fffffff         256   online   no         448-511
0x0000000080000000-0x00000000ffffffff        2048   online   yes        512-1023

Memory device size  : 4 MB
Memory block size   : 256 MB
Total online memory : 4096 MB
Total offline memory: 0 MB
```

- An additional 2GB of memory now available for use
- The change is temporary, when Linux is restarted, hotplug memory will be offline.
- Remember to make permanent changes for the dynamic resource changes.

Dynamically Provisioning Memory Resources

```
rgylx0e4:~ # chmem -d 2g
rgylx0e4:~ # lsmem
Address Range                               Size (MB)   State    Removable  Device
==========================================================================================
0x0000000000000000-0x000000000fffffff         256  online   no         0-63
0x0000000010000000-0x000000006fffffff        1536  online   yes        64-447
0x0000000070000000-0x000000007fffffff         256  online   no         448-511
0x0000000080000000-0x00000000ffffffff        2048  offline  -          512-1023

Memory device size  : 4 MB
Memory block size   : 256 MB
Total online memory : 2048 MB
Total offline memory: 2048 MB
```

- Storage no longer needed can also be removed to ensure efficient operation

Dynamically Provisioning Memory Resources

```
zlnx1:~ # lsmem
Address Range                           Size (MB)  State    Removable  Device
==========================================================================
==
0x0000000000000000-0x000000000fffffff        256  online   no         0-1
0x0000000010000000-0x000000004fffffff       1024  online   yes        2-9
0x0000000050000000-0x000000007fffffff        768  online   no         10-15
0x0000000080000000-0x00000000ffffffff       2048  offline  -          16-31

Memory device size  : 128 MB
Memory block size   : 256 MB
Total online memory : 2048 MB
Total offline memory: 2048 MB
zlnx1:~ # chmem -e 1024
zlnx1:~ # lsmem
Address Range                           Size (MB)  State    Removable  Device
==========================================================================
==
0x0000000000000000-0x000000000fffffff        256  online   no         0-1
0x0000000010000000-0x000000004fffffff       1024  online   yes        2-9
0x0000000050000000-0x000000007fffffff        768  online   no         10-15
0x0000000080000000-0x00000000bfffffff       1024  online   yes        16-23
0x00000000c0000000-0x00000000ffffffff       1024  offline  -          24-31

Memory device size  : 128 MB
Memory block size   : 256 MB
Total online memory : 3072 MB
Total offline memory: 1024 MB
zlnx1:~ # chmem -d 3072
chmem: Could only set 2048 MB of memory offline.
zlnx1:~ #
```

- The process and results are the same when running in a native LPAR as shown
- Attempts to take more memory offline than possible will result in only the removable memory being taken offline

Dynamically Provisioning – Large Pages

- Large pages can be added permanently via hugepages=<npages> in the kernel parameter line of zipl.conf
- Huge page information can be queried via /proc/meminfo

```
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       1024 kB
```

- Also queried via `/proc/sys/vm/nr_hugepages`
- Can be set dynamically via `echo xxx > /proc/sys/vm/nr_hugepages`
- Hotplug memory allocated as moveable and can only be used by movable resources.
- By default Large Pages are not allocated as movable resource but can be made to allocate from movable hotplug memory with:

```
# echo 1 > /proc/sys/vm/hugepages_treat_as_movable
```

- Hotplug memory allocated to large pages can not be set offline until all large pages are released
- For more information see Documentation/vm/hugetlbpage.txt
- Middleware exploiters may require configuration also

Dynamically Provisioning Large Pages

```
VmallocTotal:   134217728 kB
VmallocUsed:       39196 kB
VmallocChunk:   134150024 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       1024 kB
zlnx1:~ # cat   /proc/sys/vm/nr_hugepages
0
zlnx1:~ # echo 1500 >   /proc/sys/vm/nr_hugepages
zlnx1:~ # cat   /proc/sys/vm/nr_hugepages
1500
zlnx1:~ #
```

```
VmallocChunk:   134150024 kB
HugePages_Total:    1500
HugePages_Free:     1500
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       1024 kB
zlnx1:~ #
```

- Don’t forget to make dynamic changes permanent in zipl.conf kernel parameter
- Allocate your large pages as soon as possible to avoid fragmentation issues

Dynamic Memory - Considerations

- To add and remove memory takes some small advanced planning. Develop a standard policy around how you will handle memory needs.
- Memory can be added or removed whether you are running under z/VM or in a native LPAR
- zVM User Directory `COMMAND` statement provides an effective way to issue the `DEFINE STORAGE` command in an non-disruptive manner.
- Remember not all memory sections will be removable, and the removable state can change over time
- Hot plugged memory is NOT currently managed by cpuplugd memory management (cmm)

Summary of Memory Hotplug

- Basic memory hotplug requirements:
  - ✓ VM64524 support
  - ✓ DEFINE STORAGE STANDBY issued before Linux is IPLed
  - ✓ For native LPAR, RESERVED STORAGE must be defined before the LPAR is activated
  - ✓ SLES 11 / RHEL 6 provide support in Linux
- Suspend/Resume restriction: The Linux instance must not have used any hotplug memory since it was last booted. (Has worked if freed in advance)
- You may not be able to disable hotplug memory that has been enabled
- Can be very helpful when exact future memory need is unknown, without over allocating online memory from the start.
- After a Linux reboot core memory is made available again and hotplug memory is freed

Agenda

1. The Value of Dynamically Provisioning and Deprovisioning Resources
2. Dynamically Adjusting Disk Storage Resources
3. Dynamically Adjusting Networking Resources
4. Dynamically Adjusting Memory Resources
5. **Dynamically Adjusting CPU Resources**
6. Automatically Adjusting Memory and CPU Resources

Dynamically Provisioning CPU Resources

- Multithreaded application or multiple applications in a single virtual server could potentially benefit from additional virtual CPs
- Conversely too many virtual CPs could be counter productive from a performance perspective. If you are over provisioned you can dynamically remove processor resource.
- Adding or removing virtual CPs could impact monitoring applications or middleware that might query the number of processors on startup (ie the Java Virtual Machine)
- z/VM “DEFINE CPU” is a Class G command
- (R.O.T.) Don’t add unnecessary virtual CPs and never more virtual CPs than logical processors available.
- Remember adding virtual CPs does not add physical capacity to the machine
- Middleware (such as Java) is adapting to being run in a virtualized world with dynamically changing resources. Example -Xtune:elastic

```
This option turns on JVM function that accommodates changes in the machine configuration
   dynamically at run time.
-Xtune:elastic
     • Such changes might include the number of processors, or the amount of installed RAM.
```

Dynamically Managing Virtual CPs

```
===== USER RGYLX0E4 1GYLX0E4 1G 2G G
=====    INCLUDE LINDFLT
=====    CPU 00
=====    CPU 01
=====    CRYPTO    APVIRTUAL
=====    IUCV ANY
=====    LOADDEV PORTNAME 5005076306138411
=====    LOADDEV LUN 4011402E00000000
=====    MACHINE ESA 4
=====    OPTION APPLMON MAXCONN 128
```

- The directory entry shows two initial virtual CPs
- The maximum potential virtual CPs shown is four
- z/VM does not make the additional potential virtual CPs available for Linux to enable on its own
- The additional potential virtual CPs must first be **defined** in the z/VM guest before dynamically enabling on Linux

Dynamically Managing Virtual CPs

```
rgylx0e4:~ # vmcp q v
STORAGE = 1G
XSTORE = none
CPU 00   ID  FF12EBBE20978000 (BASE) CP   CPUAFF ON
CPU 01   ID  FF12EBBE20978000 CP   CPUAFF ON
AP 51 CEX2A Queue 08 shared
CONS 0009 DISCONNECTED    TERM START
     0009 CL T NOCONT NOHOLD COPY 001    READY FORM STANDARD
     0009 TO RGYLX0E4 RDR DIST RGYLX0E4  FLASHC 000 DEST OFF
     0009 FLASH       CHAR       MDFY       0 FCB       LPP OFF
     0009 3215   NOEOF OPEN 0013 NOKEEP NOMSG NONAME
     0009 SUBCHANNEL = 000A
```

- The current z/VM guests virtual resources are displayed from within Linux
- The two initial and active virtual CPs are shown
- Notice there is no information displayed about the potential additional virtual CPs

Dynamically Managing Virtual CPs

```
rgylx0e4:~ # mpstat -A
Linux 2.6.32.29-0.3-default (rgylx0e4)  04/01/11        _s390x_

13:19:24     CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest   %idle
13:19:24     all    1.43    0.00    0.65    0.30    0.00    0.02    0.06    0.00   97.53
13:19:24       0    1.62    0.00    0.67    0.29    0.00    0.02    0.03    0.00   97.37
13:19:24       1    1.25    0.00    0.64    0.30    0.00    0.02    0.08    0.00   97.70

13:19:24     CPU    intr/s
13:19:24     all      0.00
13:19:24       0      0.00
13:19:24       1      0.00

13:19:24     CPU
13:19:24       0
13:19:24       1
```

- Note the mpstat output from before defining the additional virtual CPs
- Observe the even distribution of idle time and usage

Dynamically Managing Virtual CPs

```
rgylx0e4:/sys/devices/system/cpu # ls
cpu0  cpu1  dispatching  kernel_max  offline  online  perf_events  possible  present  rescan
rgylx0e4:/sys/devices/system/cpu # cat kernel_max
63
rgylx0e4:/sys/devices/system/cpu # cat online
0-1
rgylx0e4:/sys/devices/system/cpu # cat offline
2-63
rgylx0e4:/sys/devices/system/cpu # cat possible
0-63
rgylx0e4:/sys/devices/system/cpu # cat present
0-1
rgylx0e4:/sys/devices/system/cpu # cat sched_mc_power_savings
0
rgylx0e4:/sys/devices/system/cpu #
```

- The Linux sysfs file system can access information about the two active virtual CPs
- No information about the two potential additional virtual CPs is shown yet

Dynamically Managing Virtual CPs

```
rgylx0e4:/sys/devices/system/cpu # modprobe vmcp
rgylx0e4:/sys/devices/system/cpu # vmcp define CPU 03 type cp
CPU 03 defined
rgylx0e4:/sys/devices/system/cpu # vmcp define CPU 02 type cp
CPU 02 defined
rgylx0e4:/sys/devices/system/cpu # ls
cpu0  cpu1  dispatching  kernel_max  offline  online  perf_events  possible  present
rgylx0e4:/sys/devices/system/cpu #
```

- Using the **`vmcp`** command we pass the zVM **`CP DEFINE CPU`** commands on to our z/VM guest.
- Remember this is a class G guest enabling the additional resources previously defined in the user directory
- After defining the additional virtual CPs in z/VM we still do not see them in the Linux /sys filesystem.

Dynamically Managing Virtual CPs

```
rgylx0e4:/sys/devices/system/cpu # vmcp q v
STORAGE = 1G
XSTORE = none
CPU 00  ID  FF12EBBE20978000 (BASE) CP   CPUAFF ON
CPU 01  ID  FF12EBBE20978000 CP   CPUAFF ON
CPU 03  ID  FF12EBBE20978000 STOPPED CP   CPUAFF ON
CPU 02  ID  FF12EBBE20978000 STOPPED CP   CPUAFF ON
AP 51 CEX2A Queue 08 shared
CONS 0009 DISCONNECTED    TERM START
     0009 CL T NOCONT NOHOLD COPY 001    READY FORM STANDARD
     0009 TO RGYLX0E4 RDR DIST RGYLX0E4  FLASHC 000 DEST OFF
     0009 FLASH       CHAR       MDFY       0 FCB       LPP OFF
     0009 3215   NOEOF OPEN 0013 NOKEEP NOMSG NONAME
     0009 SUBCHANNEL = 000A
RDR  000C CL * NOCONT NOHOLD   EOF       READY
     000C 2540         CLOSED    NOKEEP NORESCAN  SUBCHANNEL = 000E
PUN  000D CL A NOCONT NOHOLD COPY 001    READY FORM STANDARD
     000D TO RGYLX0E4 PUN DIST RGYLX0E4  DEST OFF
```

- By using the z/VM QUERY VIRTUAL command we can see the additional virtual CPs have been defined to the guest
- The new virtual CPs are in a “stopped” state

Dynamically Managing Virtual CPs

```
rgylx0e4:/sys/devices/system/cpu # mpstat -A
Linux 2.6.32.29-0.3-default (rgylx0e4)  04/01/11        _s390x_

13:23:58     CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest   %idle
13:23:58     all    0.47    0.00    0.23    0.10    0.00    0.01    0.02    0.00   99.16
13:23:58       0    0.54    0.00    0.24    0.10    0.00    0.01    0.01    0.00   99.10
13:23:58       1    0.41    0.00    0.23    0.10    0.00    0.01    0.03    0.00   99.23
rgylx0e4:/sys/devices/system/cpu # ls
cpu0  cpu1  dispatching  kernel_max  offline  online  perf_events  possible  present  rescan  sched_mc_p
rgylx0e4:/sys/devices/system/cpu #  echo 1 > rescan
rgylx0e4:/sys/devices/system/cpu # ls
cpu0  cpu1  cpu2  cpu3  dispatching  kernel_max  offline  online  perf_events  possible  present  rescan
rgylx0e4:/sys/devices/system/cpu #
```

- **`mpstat`** is only reporting two CPUs
- The `rescan` operation is used to search for new CPUs in the guest.
- After rescan, additional /sysfs entries exist

Dynamically Managing Virtual CPs

```
rgylx0e4:/sys/devices/system/cpu # mpstat -A
Linux 2.6.32.29-0.3-default (rgylx0e4)  04/01/11        _s390x_

13:24:41     CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest   %idle
13:24:41     all    0.43    0.00    0.21    0.09    0.00    0.01    0.02    0.00   99.23
13:24:41       0    0.49    0.00    0.22    0.09    0.00    0.01    0.01    0.00   99.18
13:24:41       1    0.37    0.00    0.21    0.09    0.00    0.01    0.02    0.00   99.29
13:24:41       2    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
13:24:41       3    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
```

- mpstat reports 0% use and 0% idle for the new CPUs. This is because they are stopped and offline
- The new CPUs must still be brought online to Linux

Dynamically Managing Virtual CPs

```
rgylx0e4:/sys/devices/system/cpu/cpu2 # echo 1 > online
rgylx0e4:/sys/devices/system/cpu/cpu2 # ls
address  capability  configure  crash_notes  idle_count  idle_time_us  online  polarization  topology
rgylx0e4:/sys/devices/system/cpu/cpu2 # cat online
1
rgylx0e4:/sys/devices/system/cpu/cpu2 # echo 1 > ../cpu3/online
rgylx0e4:/sys/devices/system/cpu/cpu2 #
```

- Bring the new CPUs online to Linux by echoing 1 in to the "online" file for the given CPU

Dynamically Managing Virtual CPs

```
rgylx0e4:/sys/devices/system/cpu # mpstat -A
Linux 2.6.32.29-0.3-default (rgylx0e4)  04/01/11        _s390x_

13:26:36     CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest   %idle
13:26:36     all    0.33    0.00    0.17    0.07    0.00    0.01    0.02    0.00   99.41
13:26:36       0    0.39    0.00    0.18    0.07    0.00    0.01    0.01    0.00   99.33
13:26:36       1    0.30    0.00    0.17    0.07    0.00    0.01    0.02    0.00   99.43
13:26:36       2    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
13:26:36       3    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
```

- On a idle system, the new CPUs momentarily show 100% idle after being brought online
- Once a little bit of workload hits the system, this quickly changes

Dynamically Managing Virtual CPs

```
rgylx0e4:/sys/devices/system/cpu # ls
cpu0  cpu1  cpu2  cpu3  dispatching  kernel_max  offline  online  perf_events  possible
rgylx0e4:/sys/devices/system/cpu # echo 0 > cpu1/online
rgylx0e4:/sys/devices/system/cpu # echo 0 > cpu3/online
rgylx0e4:/sys/devices/system/cpu #

rgylx0e4:/sys/devices/system/cpu # mpstat -A
Linux 2.6.32.29-0.3-default (rgylx0e4)  04/01/11        _s390x_

13:27:53     CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest   %idle
13:27:53     all    0.27    0.00    0.14    0.06    0.00    0.01    0.01    0.00   99.52
13:27:53       0    0.35    0.00    0.16    0.06    0.00    0.01    0.01    0.00   99.40
13:27:53       1    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
13:27:53       2    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
13:27:53       3    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
```

- You can take dynamically added CPUs offline again
- You can take offline CPUs that were initially online as well

Dynamically Managing Virtual CPs

- The latest levels of s390-tools has two new commands
  - lscpu
  - chcpu

```
[root@rgylxr64 ~]# lscpu
Architecture:          s390x
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Big Endian
CPU(s):                2
On-line CPU(s) list:   0,1
Thread(s) per core:    1
Core(s) per socket:    1
Socket(s) per book:    1
Book(s):               2
Vendor ID:             IBM/S390
BogoMIPS:              9765.00
Hypervisor:            z/VM 6.2.0
Hypervisor vendor:     IBM
Virtualization type:   full
Dispatching mode:      horizontal
[root@rgylxr64 ~]# lscpu -ae
CPU BOOK SOCKET CORE ONLINE CONFIGURED POLARIZATION ADDRESS
0   0    0      0    yes    yes        horizontal   0
1   1    1      1    yes    yes        horizontal   1
```

Dynamically Managing Virtual CPs

```
[root@rgylxr64 ~]# chcpu -r
Triggered rescan of CPUs
[root@rgylxr64 ~]# lscpu -ae
CPU BOOK SOCKET CORE ONLINE CONFIGURED POLARIZATION ADDRESS
0   0    0      0    yes    yes        horizontal   0
1   -    -      -    no     yes        horizontal   1
[root@rgylxr64 ~]# chcpu -e 1
CPU 1 enabled
[root@rgylxr64 ~]# lscpu -ae
CPU BOOK SOCKET CORE ONLINE CONFIGURED POLARIZATION ADDRESS
0   0    0      0    yes    yes        horizontal   0
1   1    1      1    yes    yes        horizontal   1
```

Rescan CPs

Enable CP 1

```
[root@rgylxr64 ~]# lscpu -ae
CPU BOOK SOCKET CORE ONLINE CONFIGURED POLARIZATION ADDRESS
0   0    0      0    yes    yes        horizontal   0
1   1    1      1    yes    yes        horizontal   1
[root@rgylxr64 ~]# chcpu -d 1
CPU 1 disabled
[root@rgylxr64 ~]# lscpu -ae
CPU BOOK SOCKET CORE ONLINE CONFIGURED POLARIZATION ADDRESS
0   0    0      0    yes    yes        horizontal   0
1   -    -      -    no     yes        horizontal   1
[root@rgylxr64 ~]#
```

Disable CP 1

Dynamically Managing Virtual CPs

```
[root@rgylxr64 ~]# lscpu -ae
CPU BOOK SOCKET CORE ONLINE CONFIGURED POLARIZATION ADDRESS
0   0    0      0    yes    yes        horizontal   0
1   1    1      1    yes    yes        horizontal   1
[root@rgylxr64 ~]# chcpu -p vertical
chcpu: Failed to set vertical dispatch mode: Operation not supported
[root@rgylxr64 ~]#
```

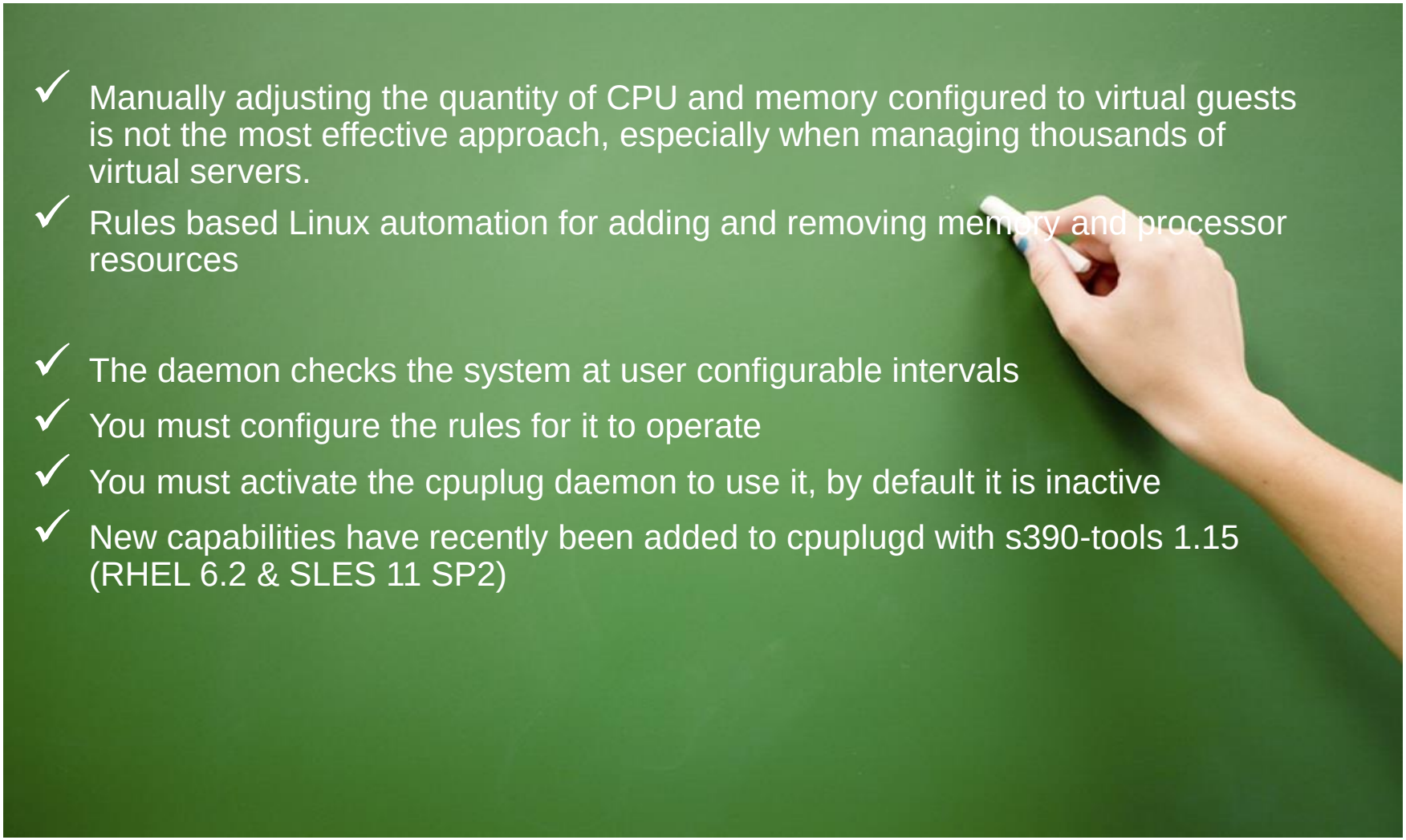
```
[root@lbskvm2 ~]# lscpu -ae
CPU BOOK SOCKET CORE L1d:L1i:L2d:L2i ONLINE CONFIGURED POLARIZATION ADDRESS
0   0    0      0    0:0:0:0         yes    yes        horizontal   0
1   0    0      1    1:1:1:1         yes    yes        horizontal   1
2   0    0      2    2:2:2:2         yes    yes        horizontal   2
3   0    0      3    3:3:3:3         yes    yes        horizontal   3
[root@lbskvm2 ~]# chcpu -p vertical
Successfully set vertical dispatching mode
[root@lbskvm2 ~]# lscpu -ae
CPU BOOK SOCKET CORE L1d:L1i:L2d:L2i ONLINE CONFIGURED POLARIZATION ADDRESS
0   0    0      0    0:0:0:0         yes    yes        vert-medium  0
1   0    0      1    1:1:1:1         yes    yes        vert-low     1
2   0    0      2    2:2:2:2         yes    yes        vert-low     2
3   0    0      3    3:3:3:3         yes    yes        vert-low     3
```

- Polarization – Configures dispatching in a Hiperdispatch like manner
- Function applies to native LPAR deployments
- Horizontal (default) – spread even across all logical process
- Vertical – dispatch across as few as possible

Agenda

1. Value of Dynamic Resource Configuration
2. Dynamically Adjusting Disk Storage Resources
3. Dynamically Adjusting Networking Resources
4. Dynamically Adjusting Memory Resources
5. Dynamically Adjusting CPU Resources
6. **Automatically Adjusting Memory and CPU Resources**

What is cpuplugd and why is it important



- ✓ Manually adjusting the quantity of CPU and memory configured to virtual guests is not the most effective approach, especially when managing thousands of virtual servers.
- ✓ Rules based Linux automation for adding and removing memory and processor resources

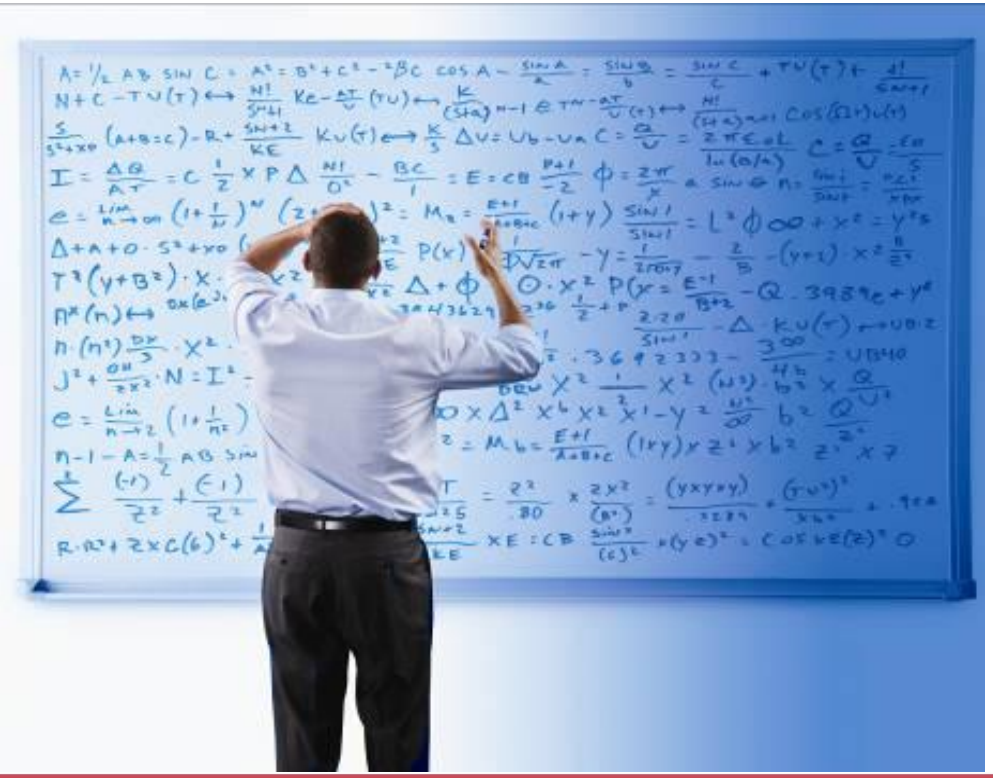
- ✓ The daemon checks the system at user configurable intervals
- ✓ You must configure the rules for it to operate
- ✓ You must activate the cpuplug daemon to use it, by default it is inactive
- ✓ New capabilities have recently been added to cpuplugd with s390-tools 1.15 (RHEL 6.2 & SLES 11 SP2)

cpuplugd – Planning

- The default rules are NOT recommendations, they are syntax examples.
- You should customize the configuration to fit your environment. Each virtual server may have different needs based on workload, middleware, and other factors.
- `cpuplugd -V -f -c /etc/sysconfig/cpuplugd` - This invokes cpuplugd in the foreground with verbose messaging to help you understand its operation. It is highly recommend you use this to understand how cpuplugd is functioning
  - Send to logfile: `cpuplugd -c <config file> -f -V>&<logname> &`
- When building rules for cpuplugd, it is important to understand what state you will be in after you execute a “plug” or “unplug” operation when writing the rules.
- Suggested Reading:
  - May 2012 *Paper ZSW03228* “**Using the Linux cpuplugd Daemon to manage CPU and memory resources from z/VM Linux guests**”

cpuplugd – Planning

- Remember some middleware such as DB2 and Oracle have memory managers and resource optimizing code of their own
- The purpose and operation of these are different than cpuplugd
- With that said any CPUs or memory brought online dynamically would immediately be available for use



cpuplugd – Rule Considerations

- Ensure you can grow CPU capacity to what the application requires to perform well (don't artificially limit). Use other mechanisms to throttle MIP usage based on shares/priorities.
- Rules based on the last couple of sample intervals are more responsive than ones based on averages over minutes. Slower responses to change can mean lower throughput for your applications
- Keep in mind you can only add/remove a full virtual CP of capacity.
- Avoid rules that plug and immediately unplug CPUs continuously
  - Plug = idle < 50
  - Unplug = idle > 50
- This means at times you might have > 1 virtual CPs of idle capacity as an acceptable state.

cpuplugd - What if I run with defaults

- `Defaults change by distro and verison/release`

- `CPU_MIN= 1` and `CPU_MAX= 0` (maximum available)
- `UPDATE= 5`
- `HOTPLUG="(loadavg > onumcpus + 0.75) & (idle < 10.0)"`
- `HOTUNPLUG="(loadavg < onumcpus - 0.25) | (idle > 50)"`
- Basic variables can be defined as:
  - `loadavg:` The load average over the past minute
  - `onumcpus:` The number of cpus which are online now
  - `runable_proc:` The current quantity of runable processes
  - `idle:` The current idle percentage
- Unplug at 51% idle? After unplug, what is my cpu busy?
- Plug only at 91% busy? What if my runable processes are growing high?

cpuplugd – What the variables represent

- **idle:** Current idle – Where 1 idle processor = 100 and 4 idle processors = 400 (/proc/stat 4th value). Idle does NOT stop at 100!
- **loadavg:** The current load average – The first /proc/loadavg value. The average number of runnable process. Not average CPU utilization! One looping process on a system would cause this to approach 1.0 Five looping processes on a single CPU system would cause this to approach 5.0
- **onumcpus:** The actual number of cpus which are online
  (Via: /sys/devices/system/cpu/cpu%d/online)
- **runable_proc**: The current quantity of runnable processes (The 4th /proc/loadavg value)

cpuplugd – Variables and rule capabilities

- New predefined keywords
  - `user` - the current CPU user percentage
  - `nice` - the current CPU nice percentage
  - `system` - the current CPU system percentage
  - `idle` - the current CPU idle percentage
  - `iowait` - the current CPU iowait percentage
  - `irq` - the current CPU irq percentage
  - `softirq` - the current CPU softirq percentage
  - `steal` - the current CPU steal percentage
  - `guest` - the current CPU guest percentage
  - `guest_nice` - the current CPU guest_nice percentage
  - `cpustat.<name>` - data from /proc/stat and /proc/loadavg
  - `time` - floating point timestamp in "seconds.microseconds" since Unix Epoch

- Historical function available and extremely useful
  - 0 is current interval
  - `cpustat.idle[0]  ….  cpustat.idle[99]`

- User Defined Variables Now Supported (See examples next slide)

User defined variables example for CPU

- user_0="(cpustat.user[0] - cpustat.user[1])"
- nice_0="(cpustat.nice[0] - cpustat.nice[1])"
- system_0="(cpustat.system[0] - cpustat.system[1])"
- user_2="(cpustat.user[2] - cpustat.user[3])"
- nice_2="(cpustat.nice[2] - cpustat.nice[3])"
- system_2="(cpustat.system[2] - cpustat.system[3])"
- CP_Active0="(user_0 + nice_0 + system_0)/ (cpustat.total_ticks[0] - cpustat.total_ticks[1])"
- CP_Active2="(user_2 + nice_2 + system_2)/ (cpustat.total_ticks[2] - cpustat.total_ticks[3])"
- **CP_ActiveAVG="(CP_Active0+CP_Active2) / 2“**

- idle_0="(cpustat.idle[0] - cpustat.idle[1])"
- iowait_0="(cpustat.iowait[0] - cpustat.iowait[1])"
- idle_2="(cpustat.idle[2] - cpustat.idle[3])"
- iowait_2="(cpustat.iowait[2] - cpustat.iowait[3])"
- CP_idle0="(idle_0 + iowait_0)/ (cpustat.total_ticks[0] - cpustat.total_ticks[1])"
- CP_idle2="(idle_2 + iowait_2)/ (cpustat.total_ticks[2] - cpustat.total_ticks[3])"
- **CP_idleAVG="(CP_idle0 + CP_idle2) / 2"**

cpuplugd memory management features

Why dynamic adjustments to memory?

- Too little free memory
  - Can't start new programs/processes
  - Swapping degrades performance
  - OOM killer kills your middleware server process to "save" the system

- Too much free memory
  - Excessive use of page cache, that may get paged out at hypervisor layer
  - Cause stealing of memory pages from other guest with legitimate need
  - More memory stress and paging at the hypervisor layer
  - Degrades overall performance

- Unlike physical only environments, over allocating memory in virtual environments can be very counter productive. This is not just true for z/VM.
- Manual adjustments don't happen fast enough

Automated adjustments of memory

- Problems stemming from over/undersized memory allocations of guests in virtualized environments are not unique to Linux on System z
- Even the most accurate sizing is irrelevant as soon as the requirements change
- The cpuplug daemon determines how much memory to add or remove based upon the rules you put in place
- It is based on the same configurable interval you set for CPU rules
- The memory increment added or removed is configurable (and you should)
- Separate plug and unplug rules are used for memory management
- There are NO default memory plug and unplug rules
- If you start cpuplugd without any configuration changes it will manage CPUs but NOT memory.
- Be sure to have the following z/VM PTFs on:
  - APAR VM65060 **REQUIRED**!
    - **540 UM33537**
    - **620 UM33539**

Automated adjustments of memory

- cpuplugd uses CMM (or cmm1) to return unused pages of memory to the hypervisor via a diagnose call. This memory must be actually free and not used as cache. The decision as to how many pages to return is controlled by the rules you write for cpuplugd
- The cmm module must be loaded
- Don't mix cpuplugd and VMRM management of CMM.
- CMMA (or cmm2) is an alterative mechanism to return pages of memory to the hypervisor by checking a bit on the page. It will only operate on free pages and z/VM has to perform a scan for memory before the pages are actually reclaimed from a guest.
- Linux pagecache can be the large consumer of free memory. If you guest idles while holding page cache z/VM could page this memory out causing long delays when activity resumes. (Example: slow ssh logins first thing in the morning)
- Understand that manually freeing pagecache alone, does not return the formerly used pages to z/VM. One of the two above CMM mechanisms must be used to return the pages.
- If the pages are not returned to z/VM, they will likely be paged out because they are idle as Linux is not using them. The next time Linux allocates something to that page, it would have to be paged in causing unnecessary delays

Linux memory management at a high level

- Understanding Linux memory management effects how you might write your plugd rules
- Application requests for memory are managed as follows:
  - With sufficient free pages, the request is fulfilled immediately
  - If that causes the amount of free memory to fall below a high water mark, an **asynchronous** page scan by kswapd is triggered in the background.
  - If serving the request would cause the amount of free memory to fall below a low water mark, a so called direct scan is triggered, and the application **waits** until this scan provides the required pages.
  - The system may decide to mark anonymous pages (pages that are not related with files on disks) for swapping and initiate that these pages be written to swap asynchronously.
- The async page scan is in an early indicator of a memory shortage
- Direct scans are more costly in terms of application performance
- Writing rules based on the scans can be more responsive than waiting until some paging activity occurs.

Automated adjustments of memory

- Basic variables for writing memory plug and unplug rules
  - **apcr:** the amount of page cache reads listed in vmstat bi/bo
  - **Freemem:** the amount of free memory (in megabyte)
  - **swaprate** the number of swapin & swapout operations
  - **cpustat.<name>** - from /proc/stat and /proc/loadavg
  - **meminfo.<name>** - any value from /proc/meminfo
  - **vmstat.<name>** - any value from /proc/vmstat
  - **time** - floating point timestamp in "seconds.microseconds"
- CMM pool size and increment
  - **CMM_MIN** min size of static page pool (default 0)
  - **CMM_MAX** max size of static page pool
    - default 512MB
  - **CMM_INC** amount for memunplug only (previously for plug and unplug)
    - 10% of free memory + cache, in pages
  - **CMM_DEC** amount for memplug operation
    - default 10% of total memory in pages
- With heavier IO rates you may want to allow the system to utilize more memory to help improve performance. This memory would get utilized by pagecache.
- Looking at “cache” for free memory might be skewed if you have a lot of shared memory (databases or java for example)

User defined variable example for memory

- The page scan rate can be calculated as the sum of:
  - `vmstat.pgscan_kswapd_dma`
  - `vmstat.pgscan_kswapd_normal`
  - `vmstat.pgscan_kswapd_movable`
    - pgscan_k="vmstat.pgscan_kswapd_dma[0] + vmstat.pgscan_kswapd_normal[0] + vmstat.pgscan_kswapd_movable[0]“
- The direct page scan rate can be calculated as the sum of:
  - `vmstat.pgscan_direct_dma`
  - `vmstat.pgscan_direct_normal`
  - `vmstat.pgscan_direct_movable`
    - pgscan_d="vmstat.pgscan_direct_dma[0] + vmstat.pgscan_direct_normal[0] + vmstat.pgscan_direct_movable[0]“
- The available part of the cache that could be freed can be calculated as the:
  - `meminfo.Cached -meminfo.Shmem`
    - avail_cache="meminfo.Cached -meminfo.Shmem"

cpuplugd summary

- CPU Hotplug memory management will NOT release page cache memory
- The CMM module needs to be loaded before starting cpuplugd
- Understand how much memory you want to allow CMM to reclaim and the rate at which you will return memory. The last thing you want is a failing memory allocation, or adverse performance impact.
- Under heavier IO load you may want more free memory available to Linux
- The goal is for Linux to dynamically return pages of memory to z/VM when not in use, and to allow the entire system to operate more efficiently
- The amount of memory required an application to run is a function of the application program code, the workload volume, and any other software added to monitor or manage the environment.
- cpuplugd does NOT plug and unplug memory (chmem), it only uses CMM
- cpuplugd does NOT add more CPUs than what you have active at boot time

References

- **Linux on System z Device Drivers, Features, and Commands**

  SC33-8411-22

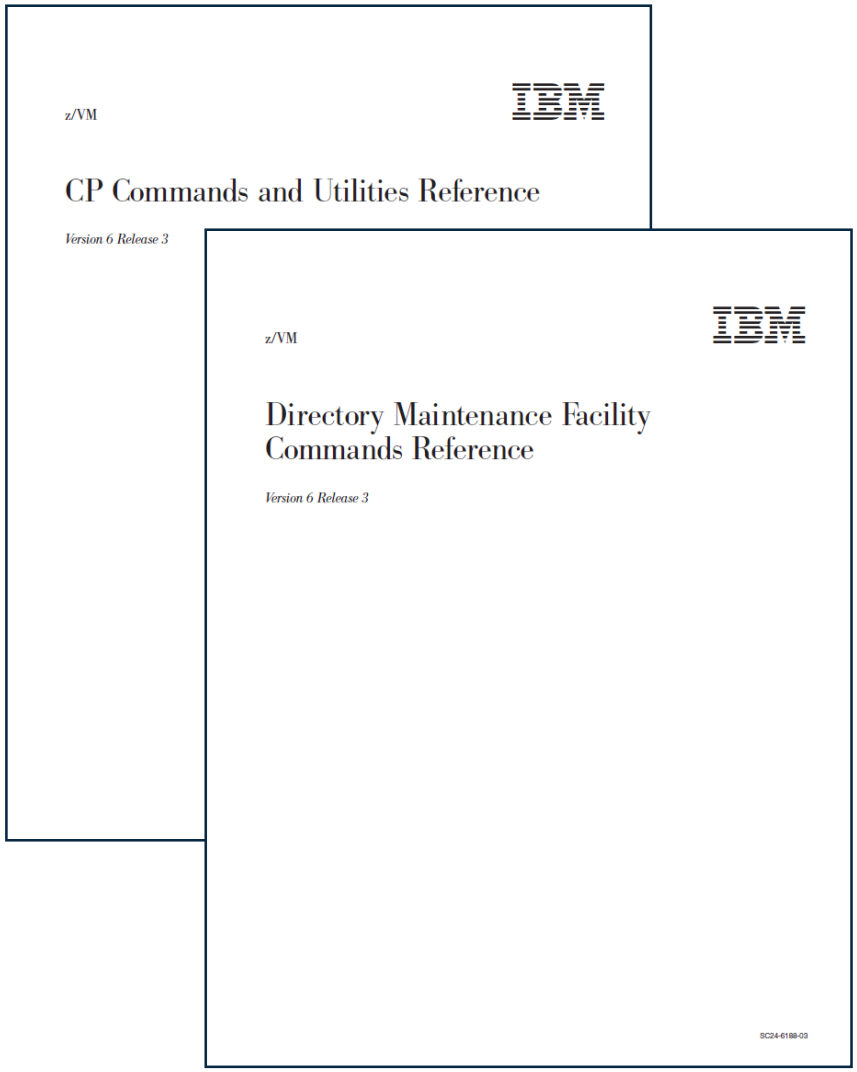
  http://www.ibm.com/developerworks/linux/linux390/documentation_dev.html

- **Using the Linux cpuplugd Daemon to manage CPU and memory resources from z/VM Linux guests**

  ZSW03228-USEN-00

  http://www.ibm.com/developerworks/linux/linux390/perf/tuning_cpuhotplug.html#cpuplugd

References



http://www.vm.ibm.com/library

- **z/VM CP Commands and Utilities Reference**
  SC24-6175-01

- **z/VM Directory Maintenance Facility Commands Reference**
  SC24-6188-03

Thank you for attending

Richard G. Young

Executive I.T. Specialist

IBM STG Lab Services

Virtualization & Linux on System z

777 East Wisconsin Ave

Milwaukee, WI 53202

Tel 262 893 8662

Email: ryoung1@us.ibm.com

Complete your session evaluations online at www.SHARE.org/Orlando-Eval