Best Practices for RHEL on System z

Brad Hinson <bhinson@redhat.com>
Worldwide System z Sales, Strategy, Marketing
Agenda

- Initial Layout
- Memory Configuration
- Disks
- Networking
- Resource Sharing
Initial Layout
LPAR vs. z/VM

- “Should I install RHEL directly in LPAR, or on top of z/VM?”
- LPAR mode
  - Good for small number of systems, and saves software cost of z/VM, but..
  - Limits scalability, utilization, some dynamic changes
  - Access to thousands of I/O resources system-wide creates its own scalability issues
    - udev allocates resources for each device
    - Don't forget possibility of human error
LPAR vs. z/VM

• z/VM
  • Some estimate 5% virtualization overhead, probably lower, but benefits outweigh overhead
  • Better utilization with VM scheduler
  • Advanced virtualization features
    - Virtual networking
    - Memory management
    - Hypervisor network services
  • Staging environment
    - Run z/VM inside z/VM
Installing Additional Guests

- **RHN Satellite**
  - A system management platform designed to provide complete lifecycle management of the operating system and applications.
Installing and Managing Guests

- Replicated Environments
Installing and Managing Guests

- RHN Satellite: Manage multiple platforms from single interface
Memory Configuration
Resource Allocations

● “What are the minimum/maximum system requirements for a RHEL guest?”

● Memory
  - Minimum
    - 512 MB for NFS install, 768 MB for HTTP/FTP
    - After install, minimum 256 MB, except special cases
      - For example, router or firewall can be 64 MB
  - Maximum
    - z/VM: 256 GB real, 1 TB virtual (real + swap)
  - Recommended
    - Rule of thumb: Divide memory on Unix/x86 by 4
      - In one case, Oracle was moved from 32GB system to 4GB z/VM guest
Resource Allocations

• “What is the ideal swap configuration?”

• Linux Swap
  - VDISK: Virtual disk in memory, allocated on use, backed by Expanded memory allocated to LPAR
  - Increase limits in z/VM SYSTEM CONFIG Vdisk,
    Userlim infinite , /* user */
    Syslim  infinite          /* system */
  - SWAPGEN.EXEC: Define VDISK swap
    - http://www.sinenomine.net/products/vm/swapgen
    - Put on shared disk (preferred) or copy to each guest
Resource Allocations

• Swap configuration
  • Linux Swap
    - Usage in PROFILE EXEC for each user
      • SWAPGEN <virtual_disk> <size in 512 byte blocks>
      • Example: SWAPGEN 300 524288
        - Define disk 300, size 256 MB
    - Small VDISK for high priority swap space: 256 MB
      • /etc/fstab: /dev/dasdX1 swap swap pri=-1 0 0
        - To find X, use 'lsdasd' and look for disk 300
    - Next VDISK is twice the size of the first: 512 MB
      • /etc/fstab: /dev/dasdX1 swap swap pri=-2 0 0
        - Finally, add some ECKD DASD for emergency
          • /etc/fstab: /dev/dasdX1 swap swap pri=-3 0 0
Resource Allocations

“How should I size the guest based on the expected workload?”

Linux memory usage

- Check /proc/meminfo:
  - Buffers: xxx kB
  - Temporary storage for raw disk blocks. Sometimes gets filled with directory entries (dentries) when apps scan many directories.
  - Solution: /proc/sys/vm/vfs_cache_pressure
  - More info
    - http://www.linuxinsight.com/proc_sys_vm_vfs_cache_pressure.html
    - Recommended: set to 10,000
  - More dangerous but immediate
Resource Allocations

• Memory

  • Linux memory usage
    - Check /proc/meminfo:
      • Cached: xxx kB
      • Pagecache. Think of this as memory cache for applications/processes. Linux should swap these out over time, but sometimes this grows to a large number.
      • Solution: /proc/sys/vm/swappiness
      • How aggressively the kernel swaps pagecache to disk. Default value 60 is reasonable for most workloads. Increase value (i.e. 70) to make Linux more inclined to swap, leaving more memory free for caches. Decrease value (i.e. 50) to make Linux less inclined to swap, and improve application responsiveness.
    • More info
Resource Allocations

- Memory
  - Linux memory usage
    - Check /proc/meminfo:
      - Committed_AS: xxx kB
      - An estimate of how much RAM/swap you would need to make a 99.99% guarantee that there never is OOM (out of memory) for this workload. Use this as a guideline for how much memory/swap to define.
  - “How much memory can I safely overcommit in z/VM?”

- Overcommit in z/VM
  - 1.5:1 to 2:1 for production, 4:1 for development
  - Don't forget to add z/VM swap, “q alloc page” for current usage
Problem: Double Paging

- Issue for virtual memory Hypervisors
  - Hypervisor determines that it is short on central storage and pages out a guest page
  - The guest determines that it wants to page out the same frame so the Hypervisor must page the frame back into central storage
  - The guest pages out the frame
  - This has been addressed by z/VM with Collaborative Memory Management
Cooperative Memory Management (CMM)

- Problem scenario:
  - Virtual memory utilization exceeds real memory availability
  - z/VM paging operations become excessive
  - Overall performance suffers
- Solution:
  - Real memory constraint detected by z/VM Virtual Machine Resource Manager
  - Linux images signaled to reduce virtual memory consumption
  - Demand on real memory is reduced, improving performance and throughput
Cooperative Memory Management (CMM)

- First enable in z/VM
  - Login as user VMRMSVM
  - Create VMRM CONFIG, add 1 line
    - NOTIFY MEMORY LNX* RH*
      - (assuming VM names begin with “LNX” and “RH”)
  - Add XAUTOLOG statement to AUTOLOG1's PROFILE EXEC
    - 'CP XAUTOLOG VMRMSVM'
Cooperative Memory Management (CMM)

- Now enable in RHEL
  - Insert CMM module (driver)
    - `# modprobe cmm`
  - Monitor/set values with `/proc/sys/vm/cmm_*`
  - Even better, let cpuplugd service set it dynamically
  - Load cmm at every boot
    - `/etc/rc.local`, last line: `modprobe cmm`
Disks
z/VM disks

- Shared vs. Dedicated volume for 191 (home)
  - Individual 191-A disk good for development environment, prepare multiple install environments
  - Shared 191-A disk good for production
    - Share installation kernel, initial RAMdisk, easily copy PARM and CONF files and/or use template
    - Dis(?)advantage: read-only 191-A disk for each user
z/VM disks

- Minidisk vs. Dedicated volumes for Linux use
  - MDISK uses minidisk caching (expanded memory), Dedicated does not
    - Debated topic, was true on older hardware, need to confirm
    - Explicitly disable minidisk caching in z/VM user directory (i.e. for Oracle)
      - MINIOPT NOMDC
  - MDISK lets you use part of a disk, more flexibility
  - MDISK can share/link disks between VMs
  - MDISK can (and should) define start at 2\textsuperscript{nd} cylinder (cyl 1) to avoid changing label
Disks for Linux Use

• 2 Kinds of Disk
  • DASD or ECKD (Extended Count Key Data)
    - Fixed sizes
      • Mod-3: 3339 cyl, approximately 2.2 GB formatted
      • Mod-9: 10017 cyl, approximately 6 GB formatted
      • Mod-27: 30051 cyl, approximately 22 GB formatted
      • Mod-54 (rare)
      • Mod-108 (now possible with RHEL 5.4 and later)
    - Some z/VM system admins more familiar with ECKD
    - May fit into some existing backup solutions
    - Need LVM to create larger file systems suitable for databases
DASD Tools

- lsdasd: Display z/VM address to Linux block device mapping

<table>
<thead>
<tr>
<th>#</th>
<th>Bus-ID</th>
<th>Status</th>
<th>Name</th>
<th>Device</th>
<th>Type</th>
<th>BlkSz</th>
<th>Size</th>
<th>Blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0.0100</td>
<td>active</td>
<td>dasda</td>
<td>94:0</td>
<td>ECKD</td>
<td>4096</td>
<td>7042MB</td>
<td>1802880</td>
<td></td>
</tr>
<tr>
<td>0.0.0101</td>
<td>active</td>
<td>dasdb</td>
<td>94:4</td>
<td>ECKD</td>
<td>4096</td>
<td>7042MB</td>
<td>1802880</td>
<td></td>
</tr>
<tr>
<td>0.0.0102</td>
<td>active</td>
<td>dasdc</td>
<td>94:8</td>
<td>ECKD</td>
<td>4096</td>
<td>7042MB</td>
<td>1802880</td>
<td></td>
</tr>
<tr>
<td>0.0.0103</td>
<td>active</td>
<td>dasdd</td>
<td>94:12</td>
<td>ECKD</td>
<td>4096</td>
<td>7042MB</td>
<td>1802880</td>
<td></td>
</tr>
<tr>
<td>0.0.0300</td>
<td>active</td>
<td>dasde</td>
<td>94:16</td>
<td>FBA</td>
<td>512</td>
<td>256MB</td>
<td>524288</td>
<td></td>
</tr>
</tbody>
</table>
DASD Tools

- **dasdfmt**: Low level format
  - `-d cdl`: Specify compatible disk layout
    - Alternative is “-d ldl” for Linux layout
  - `-b 4096`: 4k block size
  - `-p`: Display progress
    - Omit if running in background with “&”
  - `-y`: Answer “yes” to all questions, use with care
  - `-F`: Force if in use, use with care
  - `-f /dev/dasdX`: Specify DASD device

```
# dasdfmt -d cdl -b 4096 -p -y -F -f /dev/dasdd

cyl 10016 of 10016 |##########################################################################| 100%

Finished formatting the device.
Rereading the partition table... ok
```
DASD Tools

- **fdasd**: Partition tool
  - `fdasd -a /dev/dasdX`: Create one partition and exit, use with care
  - `fdasd /dev/dasdX`: Enter menu similar to “fdisk”

```bash
# fdasd -a /dev/dasdd
reading volume label ..: VOL1
reading vtoc ...........: ok

auto-creating one partition for the whole disk...
writing volume label...
writing VTOC...
rereading partition table...
```

- After partitioning, use “mke2fs -j -b 4096 /dev/dasdX1” to create file system or “pvcreate” to make LVM
DASD

• Add a new disk
  • Update dasd= in /etc/modprobe.conf
    - Use comma or dash to create list, no spaces
      • Example: 100-103,200,300-302
  • Rebuild initial RAMdisk

```
# mkinitrd -v -f /boot/initrd-$(uname -r).img $(uname -r)
```

  - Check output for “dasd=” parameter
  • Run zipl to update MBR

```
# zipl -V
```

• Bring online immediately

```
# chccwdev -e 300
```

• Preallocate dasd= to save these steps
Disks for Linux Use

• 2 Kinds of Disk
  • SCSI or zFCP (Fiber Channel Protocol)
    - Use LUNs of any size, best for database partitions
    - Point-to-Point supported, but Fibre switch recommended
    - Not managed by z/VM, managed from Linux
    - NPIV: Virtualization of WWPN and LUN, recommended for DR site
    - EDEV: FBA DASD emulation using FCP LUNs
      • Multipathed
Why FCP?

* Performance advantages
  - FCP is much faster than FICON
  - Reason 1: asynchronous I/O
  - Reason 2: no ECKD emulation overhead
* No disk size restrictions
* Up to 15 partitions (16 minor numbers per device)
* SCSI disks do not waste disk space (no low-level formatting)

* System z integration in existing FC SANs
* Use of existing FICON infrastructure
  - FICON Express adapter cards
  - FC switches / Cabling
  - Storage subsystems
* Dynamic configuration
  - Adding of new storage subsystems possible without IOCDS change
* Does NOT require more CPU than FICON
FCP

• Requires 3 Fields in /etc/zfcp.conf
  • Create this file if it doesn't exist
  • `<FCP_device> <WWPN> <LUN>`
    - Example: `0.0.010a 0x5005...073d 0x4020...0000`
  • Run `/sbin/zfcpconf.sh`, okay to run repeatedly
  • Save time, use this instead of sysfs
FCP

• Multipath
  • First add additional entries to /etc/zfcp.conf
    - Copy/paste, change WWPN to multipath channel (recommended)
    - Copy/paste, change FCP_device to multipath device (less popular, but still useful)
  • /etc/multipath.conf, comment “blacklist” statement with “#”
  • Create multipath

  # multipath -v2

  • Configure multipath to start on boot

  # chkconfig multipathd on
  # service multipathd start
  Starting multipathd daemon: [ OK ]
FCP

- Multipath
  - Give the LUN a recognizable name
    - `/etc/multipath.conf`:

```
multipaths {
  multipath {
    wwid        36005076305ffc73d0000000000000201a
    alias       test_lun
    no_path_retry 5
  }
}
```

- Result
  - Instead of `/dev/mapper/mpathX`, you have
    `/dev/mapper/test_lun`

Get this number from output of “multipath -v2 -ll”

Make up this name
FCP

- NPIV
  - Virtualize WWPN and LUN
  - Clone or kickstart a guest, use the same /etc/zfcp.conf
  - NPIV for disaster recovery
    - z/VM will virtualize FCP device, same at both sites
    - At DR site, WWPN may change
      - Solution: Use both primary and offsite entries in zfcp.conf
      - Offsite entries will silently fail
      - In DR, offsite entries will work, primary entries will silently fail
Backup

- Never backup a running Linux system. Data in memory cache won't be written to disk, backup will be inconsistent
  - Either shutdown and backup, and use external imaging/cloning software...
    - FDR/Upstream z/OS product
  - Either shutdown and backup, or use a backup product at the file system level
    - FDR/Upstream Linux product
    - Tivoli Storage Manager (TSM)
    - Veritas NetBackup (resource intensive)
    - Open Source
      - Amanda, Bacula, Linux rsync
Networking
Networking with VSWITCH

- Define 2 OSA devices, primary and backup
  - Use different CHPIDs going to separate physical switches
- Define layer 3 (IP mode, default) or layer 2 (Ethernet mode)
  - Layer 3 fine for IP traffic, where there is no need to alter Ethernet header (HTTP, FTP, NFS, most application traffic)
  - Layer 2 required for changes to Ethernet header (DHCP, VLAN, VIPA)
- Layer 2 configuration
  - /etc/sysconfig/network-scripts/ifcfg-eth0:
    - OPTIONS="layer2=1"
- Layer 3 configuration when network capture required
  - OPTIONS="fake_ll=1" for fake Ethernet header
  - z/VM VSWITCH config for AUTOLOG1
    - 'CP SET VSWITCH vswnname GRANT username PRO'
IP Address Takeover / STONITH

Server 2 takes over role of Server 1 in case of connectivity problems.
Setting Up IP Address Takeover

- Activate IP address takeover
  
  ```
  # echo 1 > /sys/bus/ccwgroup/driver/qeth/0.0.b000/ipa_takeover/enable
  ```

- Add the IP range that can be taken over
  
  ```
  # qethconf ipa add 10.1.1.0/24 eth0
  ```
  
  This enables the IP addresses 10.1.1.0 to 10.1.1.255 to be taken over

- Verify the IP address takeover configuration
  
  ```
  # qethconf ipa list
  ```

- To take over the IP address, issue
  
  ```
  # ifconfig eth0 10.1.1.1 up
  ```
Virtual IP Addresses (VIPA) Support

- Minimize outage due to adapter or network failure
- Bind server applications to system-wide virtual IP addresses (instead of adapter specific addresses)
- Server can be reached via different routes

<table>
<thead>
<tr>
<th>OSA ADDR. TABLE - layer3</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Addr</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>10.1.1.1</td>
</tr>
<tr>
<td>10.2.1.1</td>
</tr>
</tbody>
</table>

System z

Linux on System z

- Application / Server
  - dummy0
    - VIPA = 10.1.1.1
- OSA A000-A002
- OSA B000-B002
- eth0: 10.2.1.1
- eth1: 10.3.1.1

LAN1: 10.2.1.0
LAN2: 10.3.1.0

OSA Express
OSA Express
Setting Up Virtual IP Addresses (VIPA)

- Create a virtual interface and adding the VIPA using a dummy interface
  
  ```
  # modprobe dummy
  # ifconfig dummy0 10.1.1.1 netmask 255.255.255.0
  ```

  or use an interface alias
  
  ```
  # ifconfig eth0:1 10.1.1.1 netmask 255.255.255.0
  ```

- Layer3 only: register virtual IP address with physical devices
  
  ```
  # echo 10.1.1.1 > /sys/class/net/eth0/device/vipa/add4
  # echo 10.1.1.1 > /sys/class/net/eth1/device/vipa/add4
  ```

- Configure the routes with a dynamic routing daemon such as Quagga
  
  ```
  # route add -host 10.1.1.1 gw 10.2.1.1  //if LAN1 works
  # route add -host 10.1.1.1 gw 10.3.1.1  //if LAN2 works
  ```
QETH Device sysfs Attribute **large_send**

- Offload TCP segmentation from Linux stack to OSA-card:
  ```bash
  OPTIONS='large_send=TSO'
  ⇒ move workload from Linux to OSA-Express adapter
  # echo TSO > /sys/devices/qeth/0.0.b004/large_send
  ```

- Offload TCP segmentation from Linux stack to device driver:
  ```bash
  OPTIONS='large_send=EDDP'
  # echo EDDP > /sys/devices/qeth/0.0.b004/large_send
  ```

- **TSO**
  - TCP frame up to 64 kbytes
- **EDDP**
  - TCP frame up to 64 kbytes

→ performance advantage with large outgoing packets
QETH Device sysfs Attribute **checksumming**

- Offload checksumming for incoming IP packages from Linux stack to OSA-card
  
  OPTIONS='checksumming=hw_checksumming'

  or

  ```
  # echo hw_checksumming > \
  /sys/devices/qeth/0.0.b004/checksumming
  ```

- Available for OSA-devices in layer3 mode only

QETH Device sysfs Attribute **recover**

- enforce recovery of a qeth device

  ```
  # echo 1 > /sys/devices/qeth/0.0.b004/recover
  ```
QETH Device sysfs Attribute **buffer_count**

- The number of allocated buffers for inbound QDIO traffic → Memory usage.

---

**Per QETH device memory usage:**

- Control data structures: ~200 KB
- Memory for one buffer: 64 KB

<table>
<thead>
<tr>
<th>buffer_count</th>
<th>Memory usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>~712 KB</td>
</tr>
<tr>
<td>128</td>
<td>~8.4 MB</td>
</tr>
</tbody>
</table>

---

Save memory

Boost performance
Resource Sharing
Shared Kernel

- Named Saved Segments (NSS)
  - Using NSS, the z/VM hypervisor makes operating system code in shared real memory pages available to z/VM guest virtual machines.
  - Linux guest operating systems using z/VM can boot from the NSS and be run from a single copy of the Linux kernel in memory.
Shared Read-only Root
Security
Hardening a System

- SELinux
  - Policy-based security based on roles. Policies are statements that determine whether certain actions are allowed.
Hardening a System

- Other Tools
  - AIDE: Advanced Intrusion Detection Environment
    - Open source equivalent to TripWire.
  - Linux Audit Subsystem

```
Kernel

Audit Daemon

/etc/audit/audit.rules

Record Access Attempt

SELinux + File Permissions

Task Performed

Event Dispatcher

Audit Daemon

Syslog File(s)
```
Questions?