Read-only Root File System & Other Resource Sharing Techniques

Brad Hinson
World Wide Lead, Linux on System z

E : bhinson@redhat.com
P : +1 919 360 0443 (US EST)

Agenda

- Read-only Root
- Shared Home Directories
- Shared Kernel (NSS)
- Discontinuous Saved Segment (DCSS)
- Cooperative Memory Management (CMM)

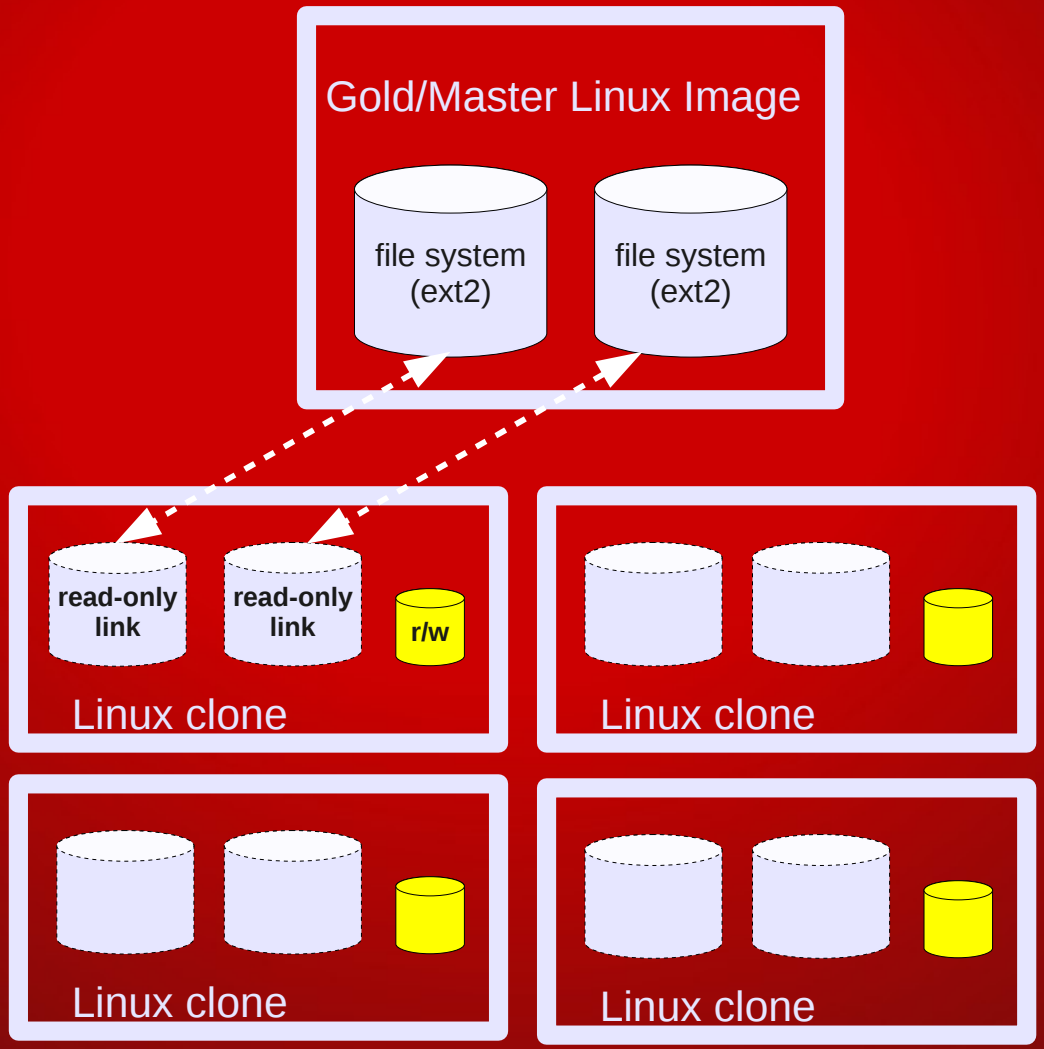
Read-only Root

- Sharing and Maintaining RHEL 5.3 Linux Under z/VM

- Overview
- Background of Read-only Root Linux
- Summary of Virtual Machines
- Building a Read-Write Maintenance System
- Building a Read-only Root System
- Maintaining Systems
- Other Considerations
- Appendices/scripts

Read-only Root

- Overview and Background



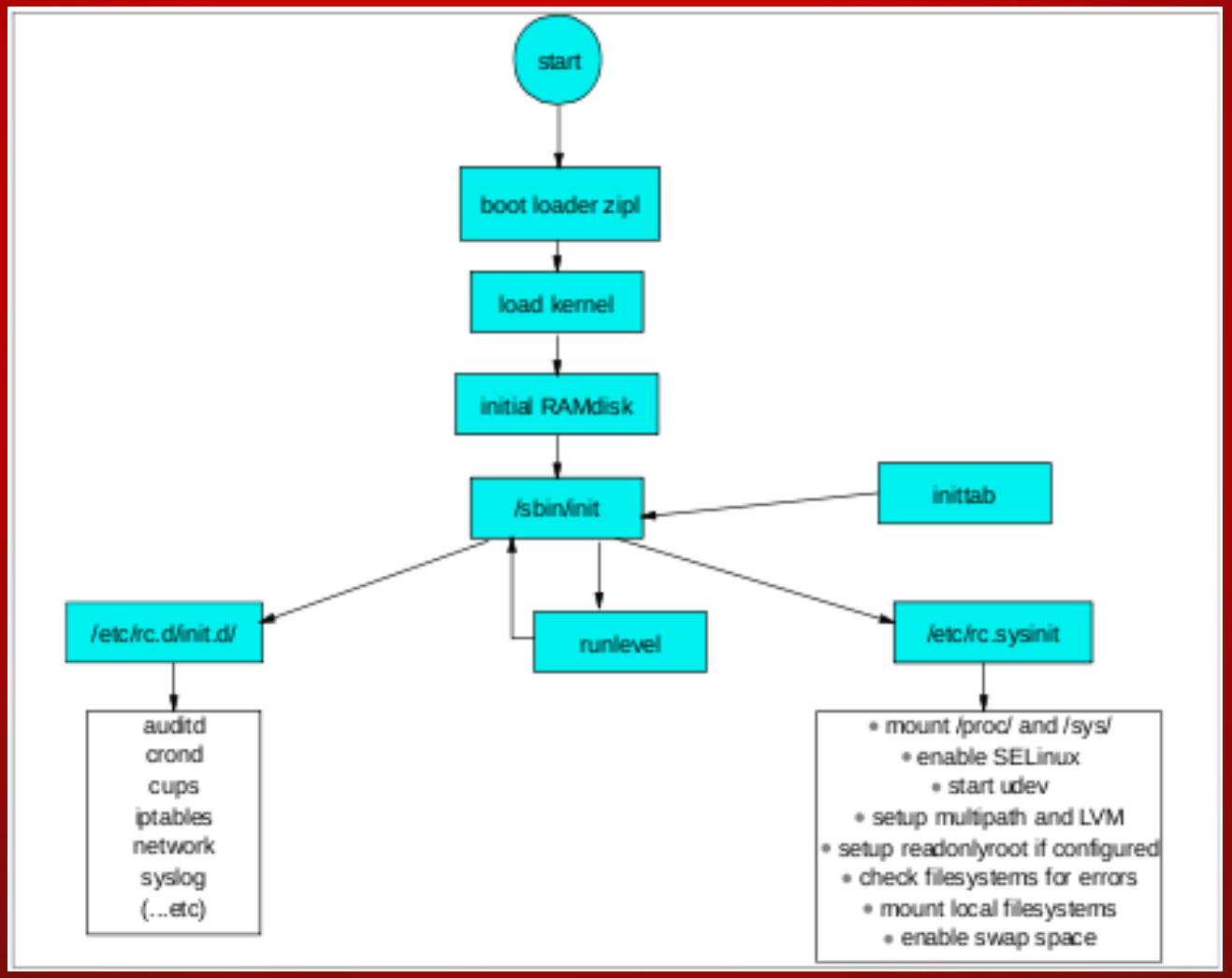
Read-only Root

- Overview and Background
- FHS: The Linux File Hierarchy Standard

/bin:	Essential command binaries
/boot:	Static files of the boot loader
/dev:	Device files
/etc:	Host-specific system configuration
/lib:	Essential shared libraries and kernel modules
/media:	Mount point for removable media
/mnt:	Mount point for mounting a file system temporarily
/opt:	Add-on application software packages
/sbin:	Essential system binaries
/srv:	Data for services provided by this system
/tmp:	Temporary files
/usr:	Secondary hierarchy
/var:	Variable data

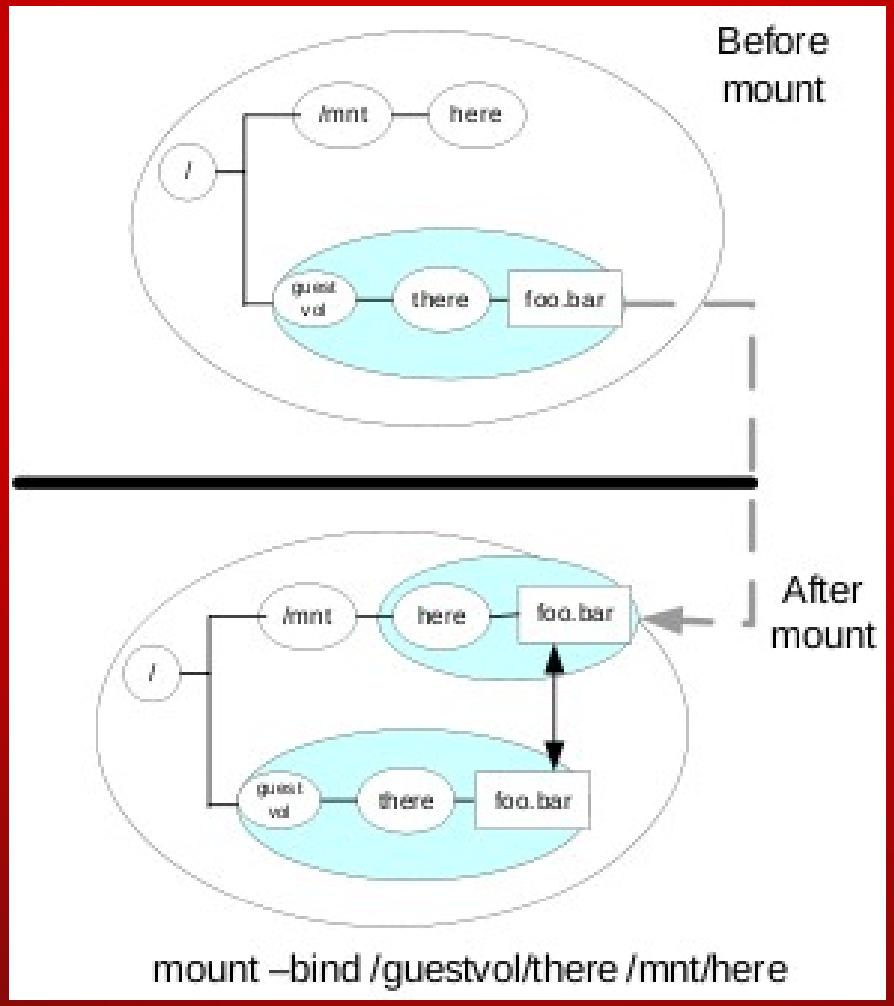
Read-only Root

- Boot Process



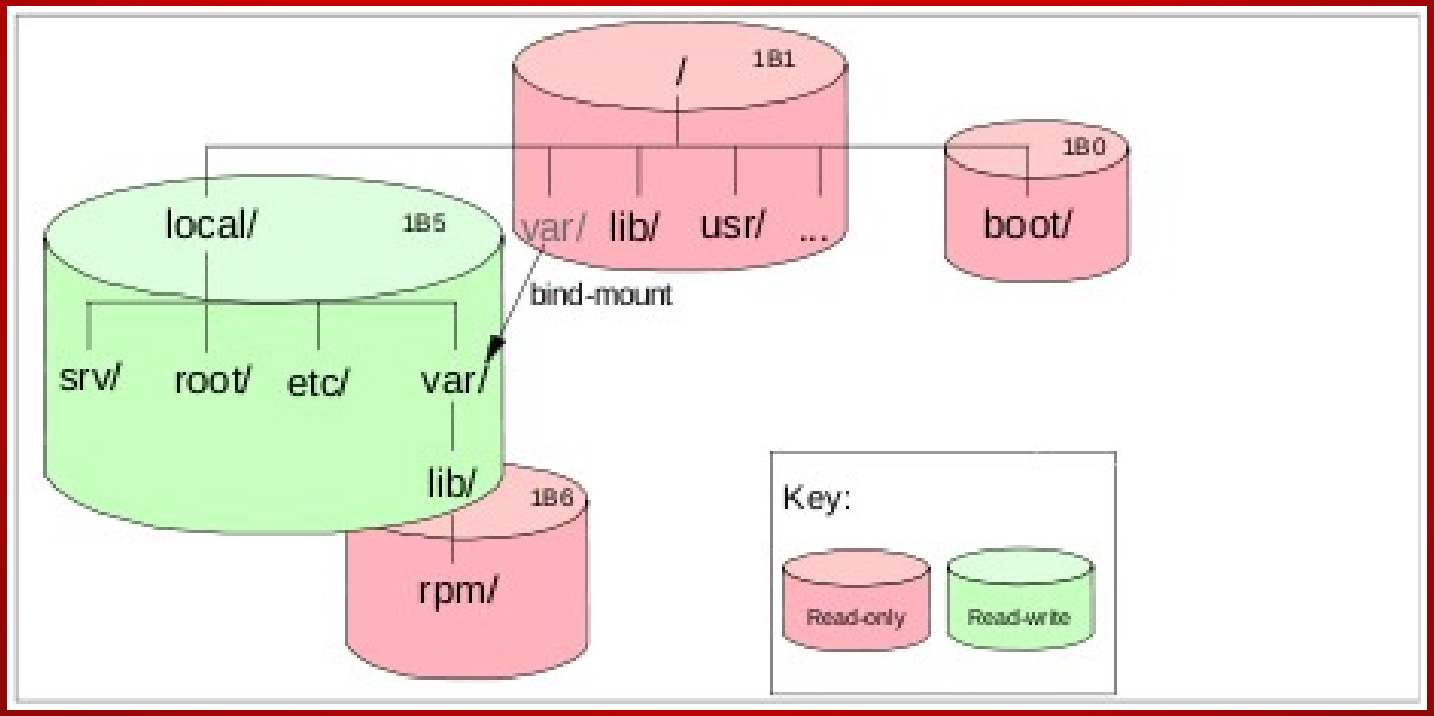
Read-only Root

- Bind Mounts (/etc, /root, /srv, /var)



Read-only Root

- File Systems and Swap Spaces



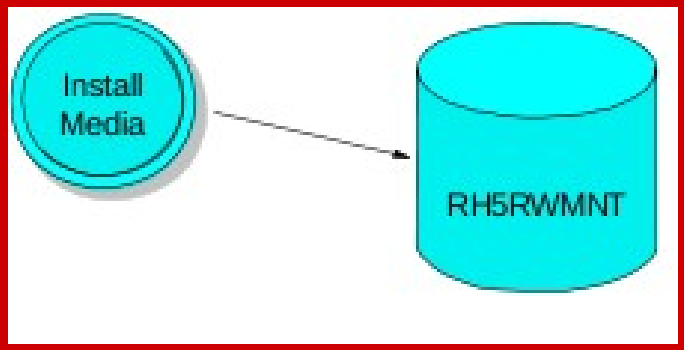
Read-only Root

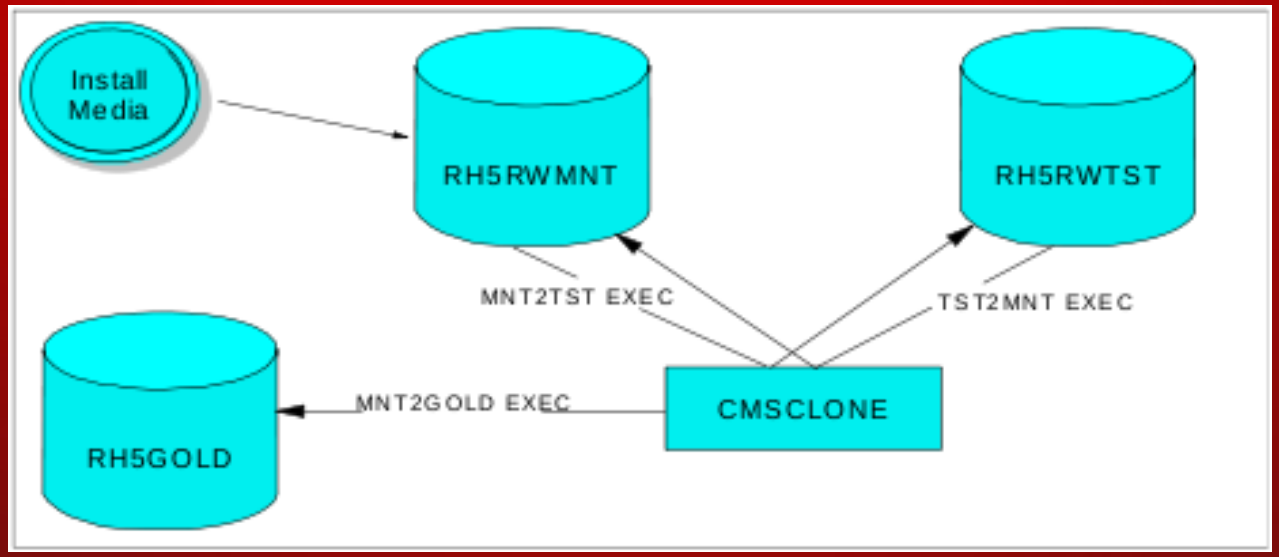
- File Systems and Swap Spaces

Directory	FS type	Attributes	Device	Vaddr	Notes
/	ext2	R/O	/dev/dasdb1	1B1	read-only root, 3200 cylinder (~2.2 GB) minidisk
/bin/	ext2	R/O			Part of root file system
/boot/	ext2	R/O	/dev/dasda1	1B0	60 cylinder (~41 MB) minidisk
/dev/	udev	R/W			The device file system
/etc/	ext3	R/W			Bind mounted from /local/etc/ to /etc/
/home/	automount	R/W			Discussed in "Implementing /home/ with automount, NFS and LDAP" on page 65
/lib/, /lib64/	ext2	R/O			Part of the root file system
/local	ext3	R/W	/dev/dasdf1	1B5	1119 cylinder minidisk (~706MB) - contains R/W /etc/, /root/, /srv/ and /var/
/mnt/	ext2	R/O			R/W directory can be mounted over R/O
/opt/	ext2	R/O			Part of the root file system
/proc/	procfs	R/W			In memory kernel file system
/root/	ext3	R/W			Bind mounted from /local/root/ to /root/
/sbin/	ext2	R/O			Part of root file system
/srv/	ext3	R/W			Bind mounted from /local/srv/ to /srv/
/sys/	sysfs	R/W			In memory file system
/tmp/	tmpfs	R/W			In memory file system - contents are lost at shutdown
/usr/	ext2	R/O			Part of the root file system
/var/	ext3	R/W			Bind mounted from /local/var/ to /var/
/var/lib/rpm/	ext2	R/O	/dev/dasdg1	1B6	Mounted read-only over read-write /var/
swap 1	swap	R/W	/dev/dasdc1	1B2	64 MB in memory VDISK
swap 2	swap	R/W	/dev/dasdd1	1B3	128 MB in memory VDISK
swap 3	swap	R/W	/dev/dasde1	1B4	550 cylinder minidisk (~384MB)

Read-only Root

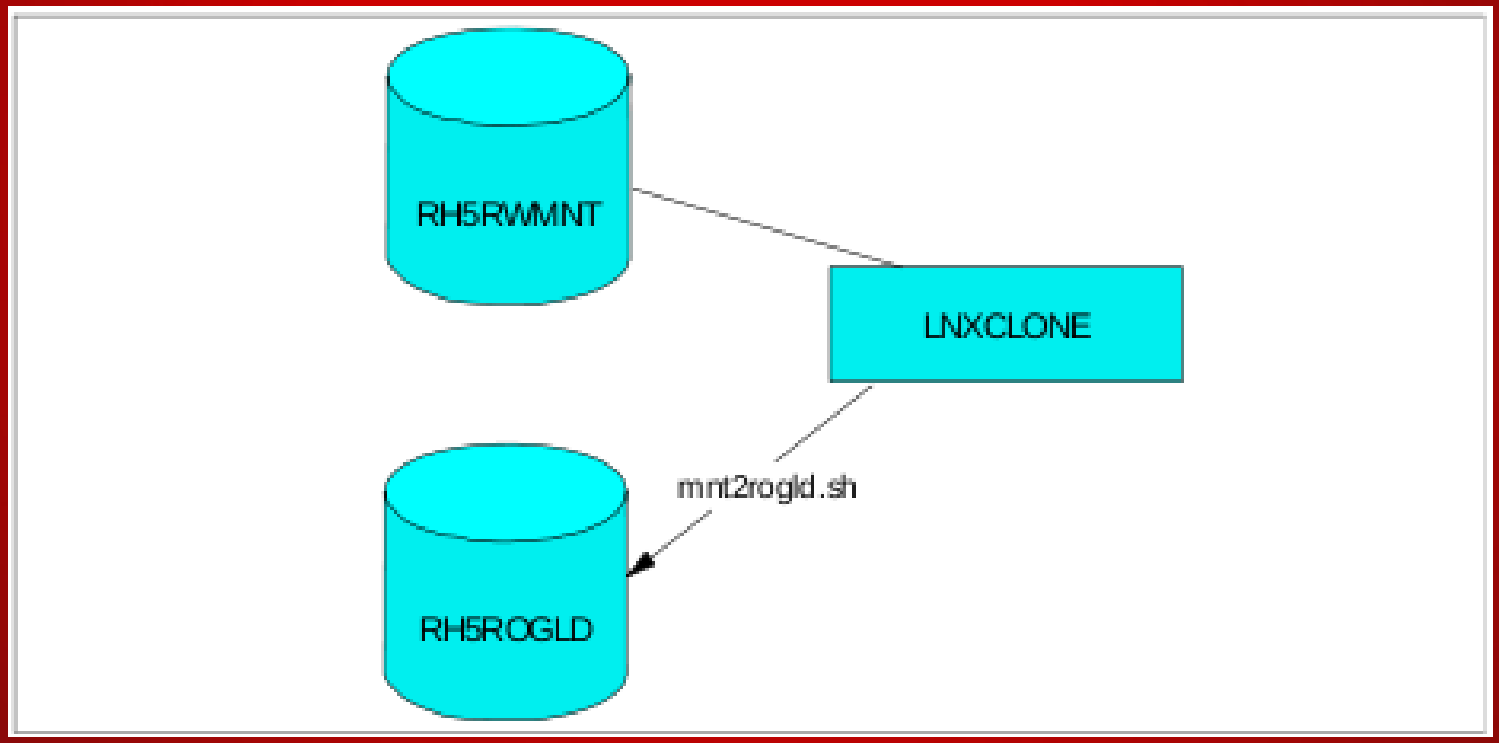
- Virtual Guests, High Level





Read-only Root

- Virtual Guests, High Level



Read-only Root

- mnt2rogold.sh:

- checkID $sourceID
- checkID $targetID
- linkSourceDisks
- linkTargetDisks
- enableSourceDisks
- enableTargetDisks
- copySystem
- mountSourceRoot
- mountTargetDisks
- ModifySystem <---***
- cleanUp
- exit

Read-only Root

- mnt2rogold.sh, ModifySystem:

- /etc/modprobe.conf:
- Add “ro” to dasd= parameter:
- options dasd_mod dasd=1b0-1bf***(ro)***,1b1***(ro)***,...

- /etc/fstab:
- Define /tmp as tmpfs:
- “tmpfs /tmp tmpfs defaults 0 0”

- Remake initrd:
- chroot $target mkinitrd -v -f /boot/initrd.img...

Read-only Root

- mnt2rogold.sh, ModifySystem:

- /etc/zipl.conf:
- Add “readonlyroot” to kernel parameter line(s)
- chroot $target /sbin/zipl

- /etc/sysconfig/readonly-root
- READONLY=yes
- STATE_MOUNT=/local

- /etc/rc.local
- Set $hostname

Read-only Root

- mnt2rogold.sh, ModifySystem:

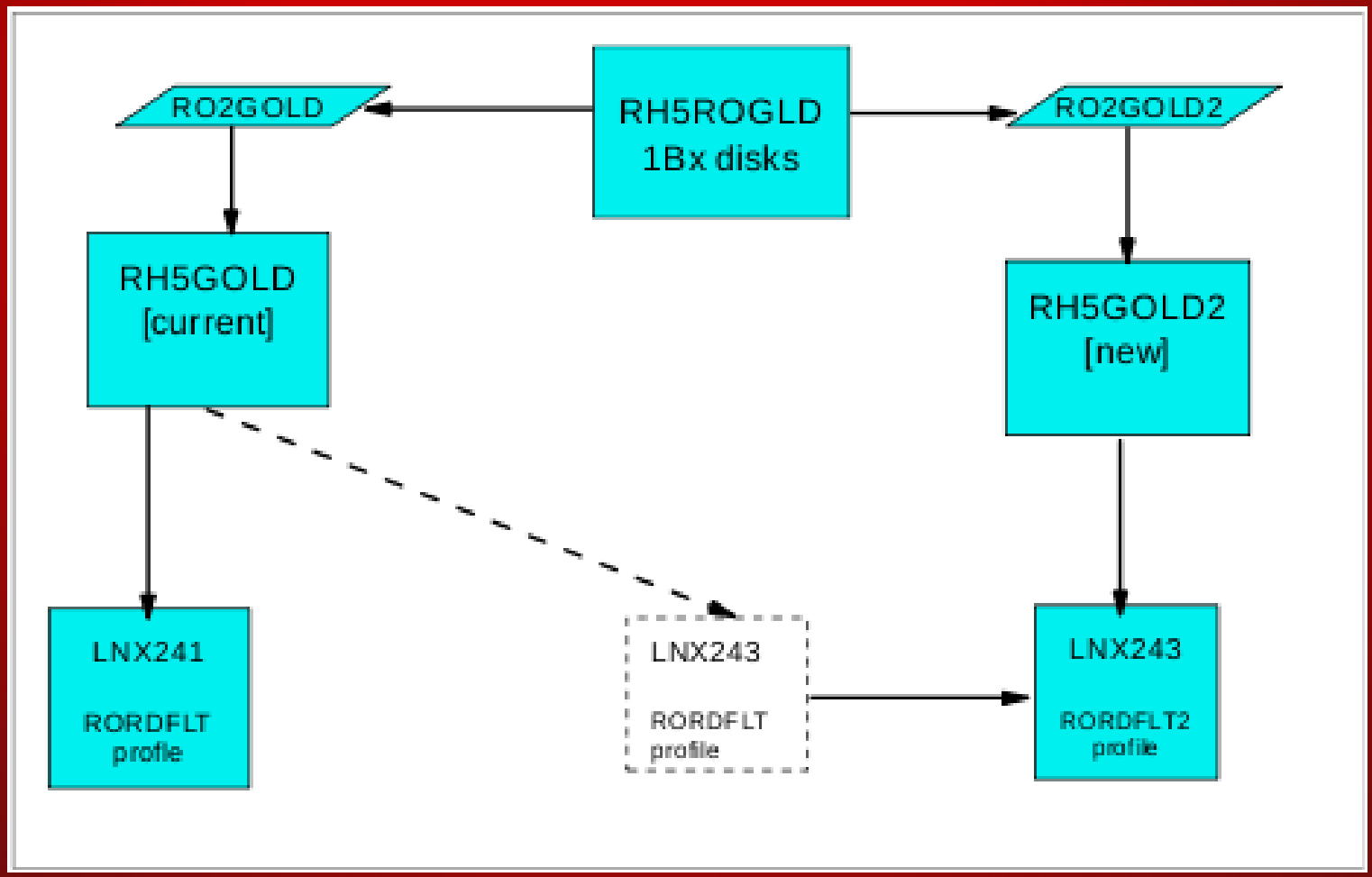
- Copy (with cp -a) directories /etc, /root, /srv, and /var to /local
- /local/files has 4 lines:
  - /etc
  - /root
  - /srv
  - /var

- After reboot, 'mount' command shows:

/dev/dasdb1 on / type ext2 (rw)
/dev/dasdg1 on /var/lib/rpm type ext2 (ro)
/dev/dasda1 on /boot type ext2 (ro)
tmpfs on /tmp type tmpfs (rw)

Read-only Root

- Maintenance



Read-only Root

- Other Considerations

- /var on LVM
- Recommended if you expect /var to grow
- Code provided, addition to ModifySystem
  - fdasd to partition
  - pvcreate, vgcreate, lvcreate, mke2fs
  - vgscan, mount

- /etc/sysconfig/readonly-root:
- Instead of STATE_MOUNT=/local, use tmpfs
- default: RW_MOUNT=/var/lib/stateless/writable
- /etc/rwtab: writable files

Read-only Root

- Other Considerations

- Stateless Linux supported as Tech Preview in RHEL 5.5

- One open issue
- https://bugzilla.redhat.com/show_bug.cgi?id=214891
- Move /etc/mtab to /var/lock
- Long term, update libmount to use /proc/mounts instead of /etc/mtab

- Links:

- http://linuxvm.org/present/misc/ro-root-RH5.pdf
- http://linuxvm.org/present/misc/ro-root-RH5.tgz

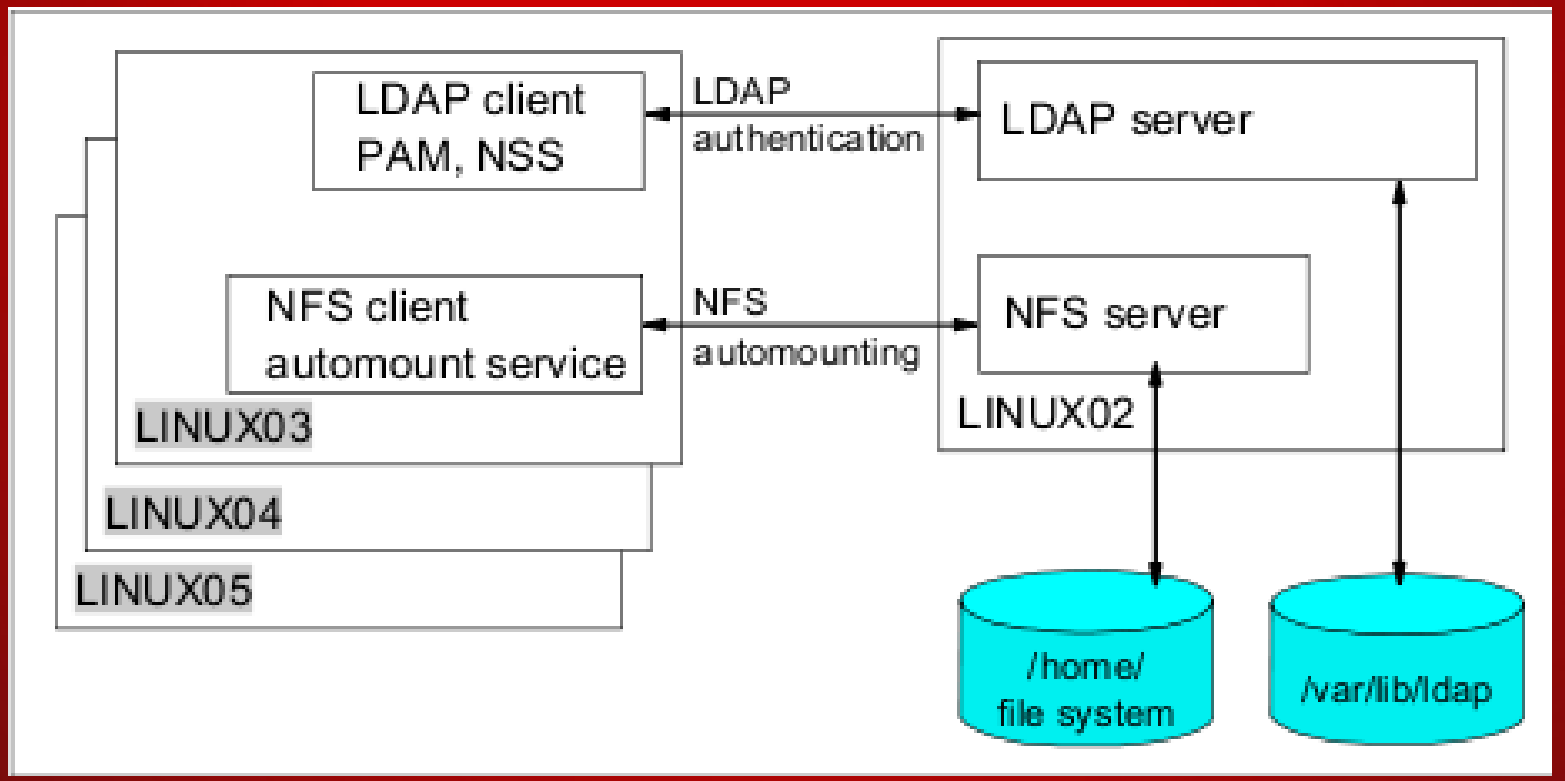
- http://fedoraproject.org/wiki/StatelessLinux

Agenda

- ✔ ~~Read-only Root~~
- Shared Home Directories
- Shared Kernel (NSS)
- Discontinuous Saved Segment (DCSS)
- Cooperative Memory Management (CMM)

Shared Home Directories

- High Level Overview



Shared Home Directories

- Configure LDAP Server

- Step-by-step detailed in RHEL 5.2 Redbook
- http://www.redbooks.ibm.com/abstracts/sg247492.html
- Section 12.2, page 164 (PDF page 182)

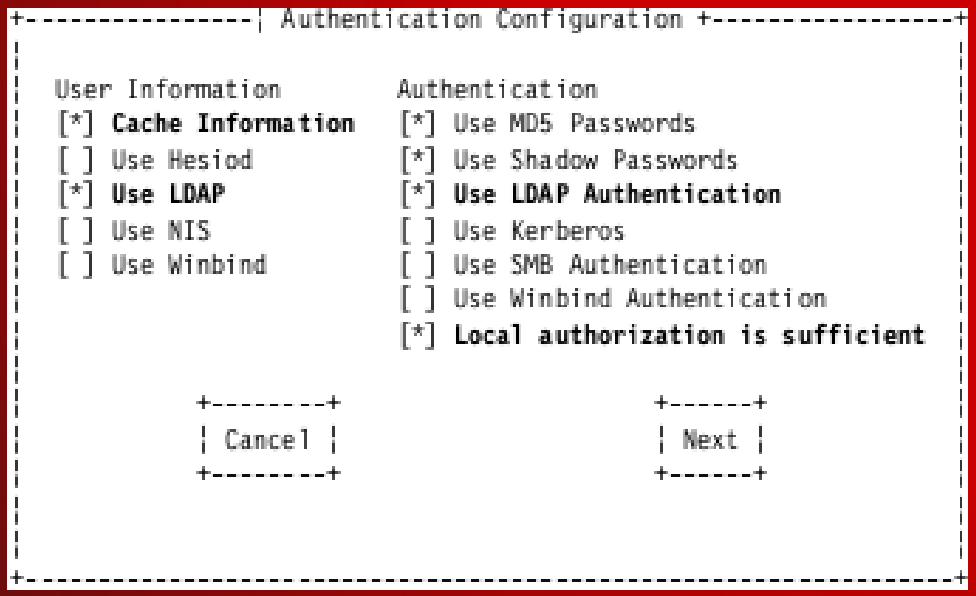
- Configure NFS Server

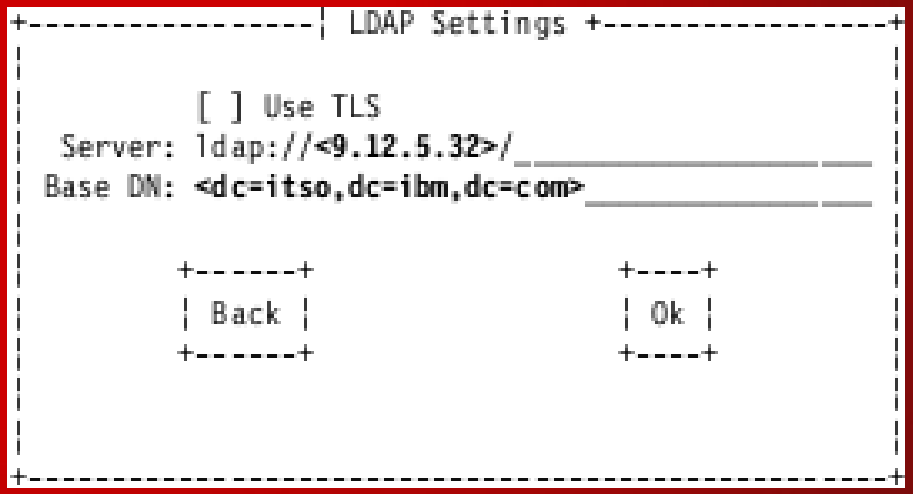
- /etc/exports:
- /home *(rw,sync)

- service nfs start
- chkconfig nfs on

Shared Home Directories

- Configure LDAP Client Authentication
- authconfig-tui (text) or system-config-authentication (graphical)





Shared Home Directories

- Configure Client Automount

- /etc/auto.master:
- /home /etc/auto.home

- /etc/auto.home:
- * <nfs_server_hostname>:/home/&

- service autofs {re}start
- chkconfig autofs on
- (note: already enabled by default)

Shared Home Directories

- Test it all out

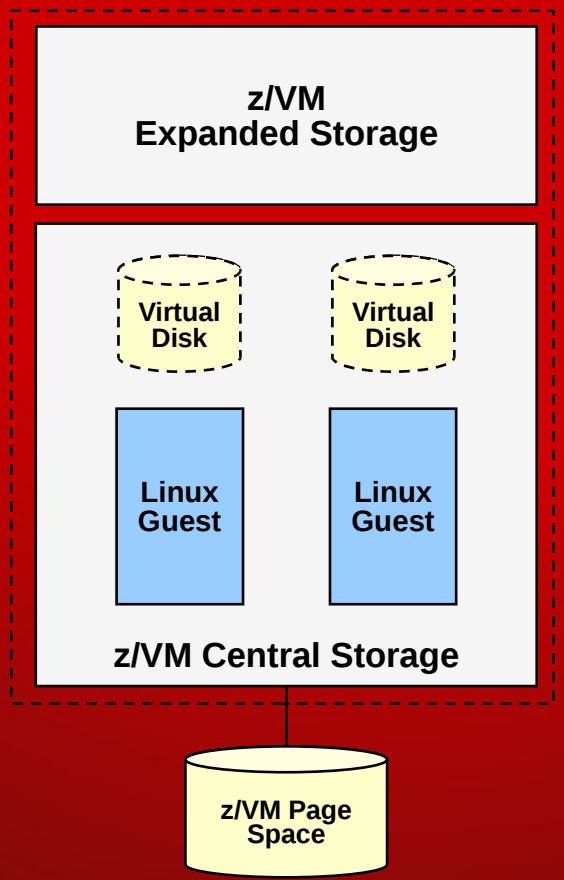
```
# service autofs restart
Stopping automount: [ OK ]
Starting automount: [ OK ]
# su - ldapuser1
$ pwd
/home/ldapuser1
$ mount | grep ldapuser1
9.12.5.32:/home/ldapuser1 on /home/ldapuser1 type nfs
(rw,addr=9.12.5.32)
```

Agenda

- ✔ ~~Read-only Root~~
- ✔ ~~Shared Home Directories~~
- Shared Kernel (NSS)
- Discontinuous Saved Segment (DCSS)
- Cooperative Memory Management (CMM)

Shared Kernel

- Named Saved Segment (NSS)
- Boot from NSS, run a single copy of Linux kernel in shared real memory pages available to z/VM guest virtual machines.



Shared Kernel

- NSS Requirements
- Class E privilege in z/VM user definition to create NSS
- Kernel built with CONFIG_SHARED_KERNEL=y
- RHEL 5.4
- Identical disk layout
- Identical kernel parameters (*)

Shared Kernel

- Create NSS

- /etc/zipl.conf:
- parameters=”... savesys=nss_name”
- /sbin/zipl
- (reboot)

- Shut down, IPL from NSS
- IPL nss_name
- IPL nss_name PARM extra_parms

- Update NSS: same process

- Delete NSS: CP PURGE NSS nss_name

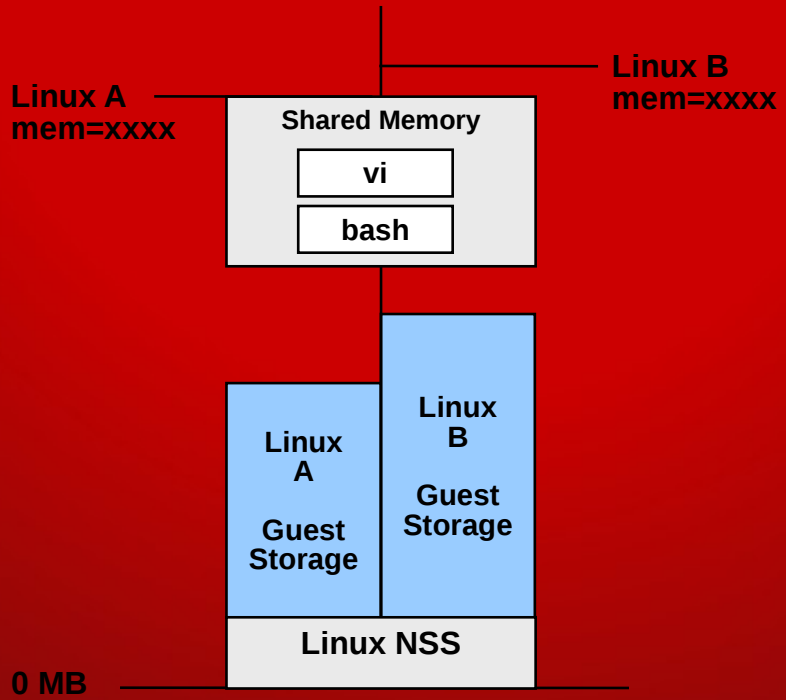
Agenda

- ✔ ~~Read-only Root~~
- ✔ ~~Shared Home Directories~~
- ✔ ~~Shared Kernel (NSS)~~
- Discontinuous Saved Segment (DCSS)
- Cooperative Memory Management (CMM)

Discontinuous Saved Segment

- Shared memory for binaries that looks like a disk

- Examples: Using DCSS and XIP2 Filesystems
- http://www.redbooks.ibm.com/abstracts/sg247285.html



Discontinuous Saved Segment

- Creating a DCSS

- Verify free space in z/VM
- CP QUERY NSS ALL MAP

- CP DEFSEG dcss_name <range> <type> <options>
- <type>: first character: S=shared, E=exclusive
- <type>: second character:
- R=Read-only
- W=Read-write
- N=Read-write, but no data saved
- C=Read-write for CP, Read-only for virt machine, no data saved

Discontinuous Saved Segment

- Creating a DCSS

- CP DEFSEG dcss_name <range> <type> <options>
- Requires z/VM Class E permission
- <options>:
- SAMErange: definition is same as one previously saved
- RSTD: restricted, requires NAMESAVE directory statement for access
- LOADNSHR: shared with no restriction, no NAMESAVE required
- SECURE: only creator can dump/restore from tape
- SPACE space_name: 1-8 character name for segment space

- Example (16 MB DCSS in 240MB-256MB range):

- CP DEFSEG dcss_name F000-FFFF SR LOADNSHR
- CP SAVESEG dcss_name

Discontinuous Saved Segment

- Accessing from Linux

- DCSS block driver
- # modprobe dcssblk segments=dcss_name1,dcss_name2,...
  or
- # modprobe dcssblk
- echo dcss_name1:dcss_name2:... > /sys/devices/dcssblk/add

- List active DCSS:
- cat /sys/devices/dcssblk/seglist

- Set access mode
- # echo {access_mode} > /sys/devices/dcssblk/dcss_name/shared
- access_mode=0 for exclusive-writable, 1 for shared

Discontinuous Saved Segment

- Accessing from Linux

- Changes are volatile until saved
- # echo 1 > /sys/devices/dcssblk/dcss_name/save
- (echo 0 to purge an existing save request)

- Remove a DCSS:
- # echo dcss_name > /sys/devices/dcssblk/remove

Discontinuous Saved Segment

- Recap so far: Creating a DCSS

- Create DCSS within guest's memory range
- Edit /etc/zipl.conf, add mem=<current memory value>
- Run /sbin/zipl
- Shutdown guest
- DEF STOR <memory value – DCSS size>
- (Long term: update user directory entry)
- Boot guest, insert DCSS driver (dcssblk)

Discontinuous Saved Segment

- DCSS with XIP
- XIP: Execute In Place
- Executable on traditional file system: read from disk into memory, execute
- XIP: Execute right from “disk”

Discontinuous Saved Segment

- Configure XIP

- Get exclusive access to DCSS:
- # echo 0 > /sys/devices/dcssblk/dcss_name/shared

- Create ext2 file system:
- # mke2fs -b 4096 /dev/dcssblk0
- -b: set block size equal to memory page size

- Mount, and copy files like any other file system
- Unmount

- Save the DCSS:
- # echo 1 > /sys/devices/dcssblk/dcss_name/save

- Free the old DCSS
- CP PURGE NSS <file_id>

Discontinuous Saved Segment

- Configure XIP:

- Now, add DCSS and mount with option “xip”
- # echo dcss_name > /sys/devices/dcssblk/add
- # mount -o ro,xip /dev/dcssblk0 /mnt/xip

- Next steps:

- Add to DCSS driver /etc/modprobe.conf:
- options dcssblk segments=dcss_name

- Recreate initial RAMdisk:
- # mkinitrd --with dcssblk -v -f /boot/initrd-$(uname -r).img $(uname -r)
- # /sbin/zipl

- add to /etc/fstab

Agenda

- ✔ ~~Read-only Root~~
- ✔ ~~Shared Home Directories~~
- ✔ ~~Shared Kernel (NSS)~~
- ✔ ~~Discontinuous Saved Segment (DCSS)~~
- Cooperative Memory Management (CMM)

Cooperative Memory Management

- Reduce available memory to a Linux guest
- Reuse these pages for other guest

- Reduce Linux memory footprint

- Requirements:
- z/VM 5.3, or z/VM 5.2 with APAR VM64805
- RHEL 4.7 or RHEL 5.1 or later

Cooperative Memory Management

- Two parts:
- VM Resource Manager (VMRMSVM)
- Linux driver for CMM processing

- z/VM Setup:
- Logon user ID VMRMSVM
- ==> XEDIT VMRM CONFIG
- ADMIN MSGUSER VMRMADMN
- NOTIFY MEMORY LNX* RH5*

- xautolog VMRMSVM in AUTOLOG1's profile exec

- Linux Setup:
- # modprobe cmm
- # cat /proc/sys/vm/cmm_pages
- # cat /proc/sys/vm/cmm_timed_pages

- cpuplugd daemon from s390-tools (s390utils) supports CMM

Cooperative Memory Management

- Links:

- Device Drivers, Features, and Commands
- ibm.com/developerworks/linux/linux390/documentation_dev.html
- Chapter 24: Cooperative Memory Management

- Overview
- http://www.vm.ibm.com/sysman/vmrm/vmrmcmm.html

Agenda

- ✔ ~~Read-only Root~~
- ✔ ~~Shared Home Directories~~
- ✔ ~~Shared Kernel (NSS)~~
- ✔ ~~Discontinuous Saved Segment (DCSS)~~
- ✔ ~~Cooperative Memory Management (CMM)~~

Q&A

Brad Hinson
World Wide Lead, Linux on System z

E : bhinson@redhat.com
P : +1 919 360 0443 (US EST)