Managing a Penguin Farm on the VM Prairie

Share 106, Seattle WA
Session 9210
March 2006

Gordon W. Wolfe, Ph.D

vmlinux@locolobo.fastmail.fm

Disclaimer

- Use the contents of this presentation at your own risk. The author does accept any liability, implicitly or explicitly, direct or consequentially, for use, abuse, misuse, lack of use, misunderstandings, mistakes, omissions, mis-information for anything in or not in, related to or not related to, pertaining or not pertaining to this document, or anything else that a lawyer can think of or not think of.
- All information in this presentation is from the opinions and personal experiences of Gordon Wolfe alone, and may or may not represent the opinions or official stance of any other entity, specifically including, but not limited to, present and former employers and clients of the author.

Introduction

VM originally created to run many operating systems on one real machine

Perfect for running Linux guests. VM acts as “hypervisor’ or “Netbios” for Linu

Won’t go into reasons for using VM or Linux here.

Assume you are familiar with both VM and Linux.

Presentation will be specific to z/VM 5.1.0 and SuSE Linux SLES8/SLES9 as experienced by the author, with extensions by implication to other operating systems releases.

Object here will be to discuss how to run MANY linuxes as VM guests and have productive system

I will sometimes note availability of some commercial software products. I am not aware of them all. Sorry if I missed anyone. Nothing in this presentation is an endorsement of any product or Company.

The Problem

VM can support hundreds (or thousands) of virtual servers. W
estimate about 350 productive Linux servers per z800 IFL
engine.

When you get that many guest operating systems, how do you

- Keep everything consistent?
- Let them talk to each other and clients?
- Handle updates?
- Use VM's resources most efficiently?

You need a Plan!

Implementation at One Installation

One z800 with two IFL Engines and two s390 engines

Six z/VM 5.1 LPARs - One for Linux

OSA Express

SuSE SLES7 on 2 guests

SuSE SLES8 on 15 guests

SuSE SLES9 (64-bit) on 32 guests

Consistency!

Multiple Linux guests become impossible to manage if they are all different.

STANDARDIZE, STANDARDIZE, STANDARDIZE!

- Stick to one distribution, one release of Linux
- Try as much as possible to make every Linux guest work like every other Linux Guest.
- Keep similar files in the same places

Have Written Policies

- Have Formal Service Level Agreements
  - Times of Operation/time of maintenance
  - Guaranteed levels of performance
  - Software levels
  - General agreement, not one for each server.
  - On-line server request form
  - Help desk
  - Announcement bulletins

Have Written processes to follow

- For cloning a new server
- For fixing a damaged boot disk
- For upgrading software
- Adding disks, using LVM, etc.

Don’t Give Your Customers Root!

- You will know what and how software is installed
- Customer can’t modify the kernel
- Customer can’t modify security arrangements
- Give customer “su” if privileged commands are needed - on a command-by-command basis.
- Let your Linux support personnel handle all changes

Keep Policies and forms accessible on the web

Written policy on how to obtain a server. Maybe even for for requesting a server.

Lots of how-to documents for users:

- Setting up KDE
- Keeping Linux secure
- Running Samba
- Running Apache
- Etc.

Can use Linux Apache for this purpose!

Common 191 disk

Owned by clone server

PROFILE EXEC

- Choose boot from DASD (default) or Reader
- If boot from DASD, SWAPGEN EXEC for V-DISK
- <server> EXEC to couple VCTCA’s or set up Guest Lan

Contains files needed to boot from reader

- INITRD
- <server> PARMFILE
- IMAGE

See Appendix 1 in handouts

Memory and Swap

Linux swap to VM V-DISK in real/expanded storage

- Means VM does real paging - more efficient!
- Set up in PROFILE EXEC on common 191 disk
- Use SWAPGEN EXEC from sinenomine.com
- V-DISK defined in directory entry

Keep Linux Memory to a minimum (otherwise Linux fills up with buffe

- Reduce virtual memory until swapping just starts. (use TOP to see this)
- Only need 64MB for Apache, Samba server in SLES8, 96MB in SLES9.
- Need 256 MB for Oracle 9i - minimum!
- Need 384MB for WebSphere Application Server!

Booting from the reader requires at least 128MB for SLES8 and SLE

Patch and CD server

NFS Server accessible r/o by any Linux administrator

- Use mount -t nfs -o patchsvr:/share /mountpoint from server1

/cd - to do installs of new software - copy install CDs to this directory

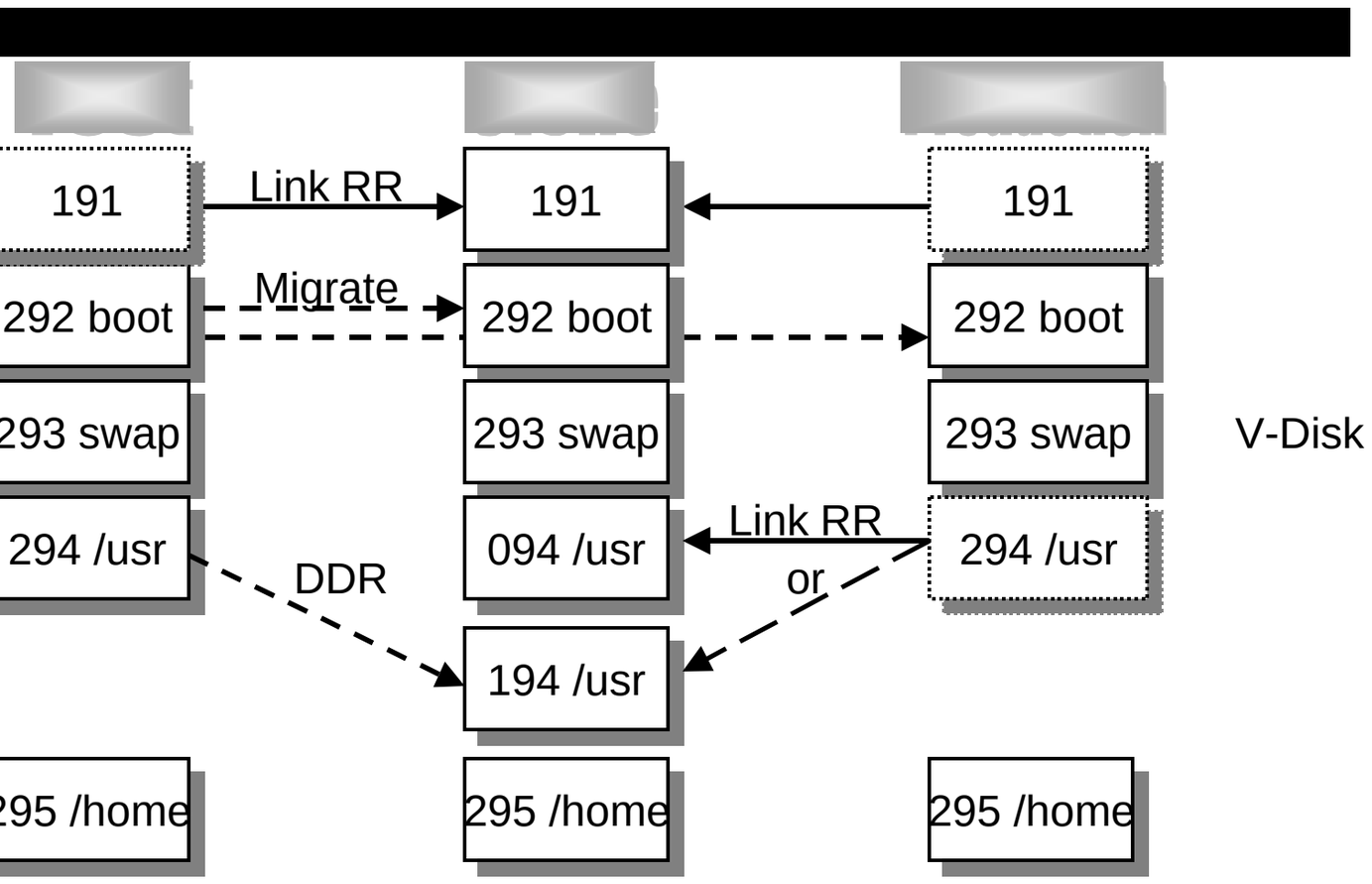
- **/cd/SLES8**
  - **../CD1**
  - **../CD2**

- **Use Mike MacIsaac's mksles9root.sh to create the directory tree. See linuxvm.org/patches**

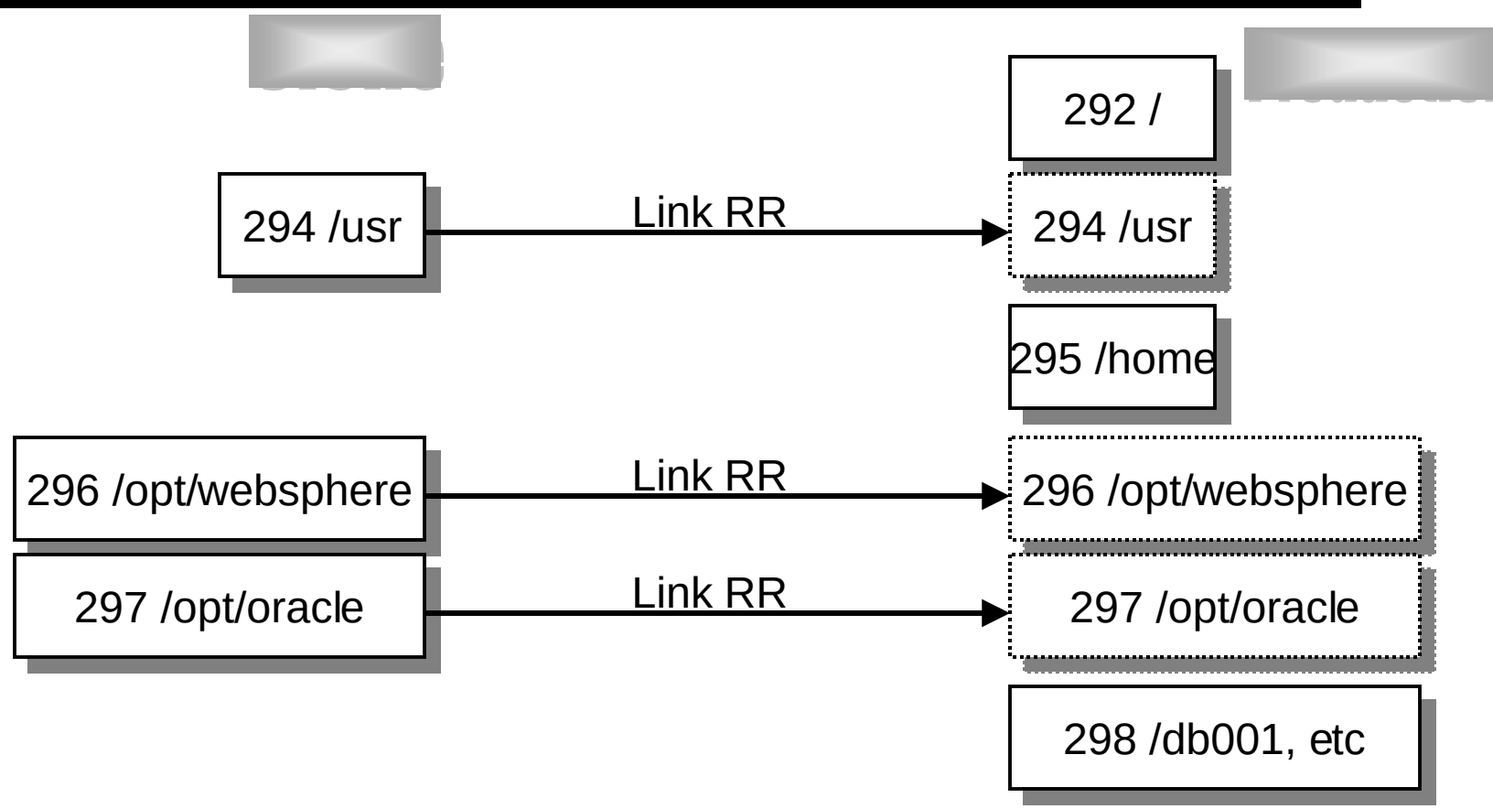
Method One: Shared read-only /Usr disks

- Advantages:
  - Reduces usage of disk space
  - All updates to /usr done in one place
- Disadvantages:
  - Have to have multiple disks for each release new/old
  - Have to separate out upgrade files on /usr and elsewhere
  - Rather labor intensive for upgrades.
  - Rpm database not necessarily kept in sync

Test, Clone and Production



Different Filesystems for Different Applications



Shared /usr and other Filesystems

Disk(s) owned by clone server

094 for production , 194 for new/updated self-link 194 as 294 RW

Clone machine is shut down virtually all the time, except when updating files.

Link clone 094 disk as 294 RR in directory of server

/etc/fstab should mount /usr “ro”

Parm line in /etc/zipl.conf should read dasd=0293,0292,294(ro),295-2AF root=/dev/dasdb1

Keep extra DASD devices in parm in case you need to add one later.

When clone machine done updating 194, do

- mount /usr -o ro,remount
- sync;sync
- Shutdown

Same for other R/O filesystems

Routing Updates and Changes

- Use scp and ssh to route new files around. (RPM -ql <package> gives list of all files contained in <package.rpm>)
- Have to set up each server to allow no-password ssh/scp using public keys from a userid on your test machine.
- Sample scripts in appendix 4 for automated method.
- Shut down server and swap LINK LXCLONE 094 294 RR for LINK LXCLONE 194 294 RR
- Reboot and hope it comes up.

Read/Write on a Read-Only Directory

- R/O user means some functions have to be moved to a R/W disk, e.g.
  - Apache Webserver
  - Move /usr/local/httpd to /home/httpd or other R/W location
  - Update location in /etc/httpd/httpd.conf

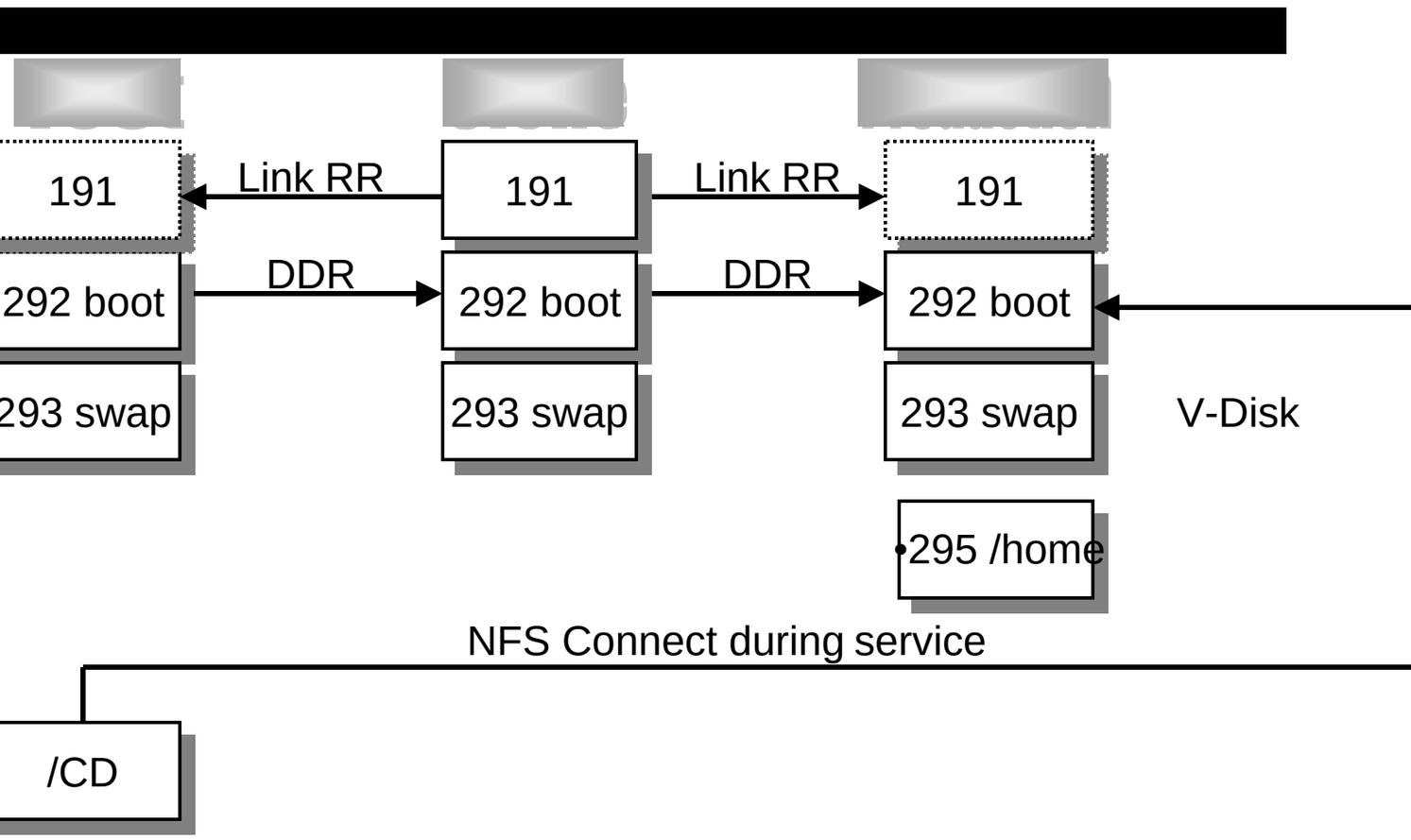
Additional Files in a Read-Only Directory

- Create a new subdirectory on a R/W disk
  - Mkdir /home/mystuff
- Copy files from R/O directory (/usr/mystuff) to it
  - Cd /usr/mystuff
  - tar cf - . | tar xpf - -C /home/mystuff
  - Add or change anything you want to here
- Mount R/W subdirectory over R/O subdirectory cd /home
  - mount -o rw --bind /home/mystuff /usr/mystuff
- A variation is the Basevol/Guestvol schema described at http://linuxvm.org/Info/HOWTOs/basevol9.html

Method 2: Every user gets nonshared Read-Write disks

- Advantages
  - Upgrades are made easy - just install rpm’s from central nfs server.
  - Rpm database in sync
  - Less labor intensive
  - Able to share free space on disk.
- Disadvantages:
  - /usr alone requires a full-pack 3390-3 for each linux. (“Disk is cheap, labor is not.”)
  - Can instead dedicate half a 3390-9 for all system areas.

Test, Clone and Production



Upgrades

Have to set up each server to allow no-password ssh/scp using public keys from a userid on your test machine.

Ssh command to mount central nfs repositiory of rpm’s

Ssh commands to install rpms

Umount nfs

Can place all this in a single script or use automation in appendix.

Cloning Penguins: Part 1

- reate exec to do couples (ctc, guest lan or vswitch
- arm file on 191
- et up TCPIP to talk to server
- reate the directory entry
  - Link RR to 191, /usr, other product disks
  - Add in V-disk entry for swapping
  - Add in disk for /home
- DR the boot (and /usr?) disk from the clone serve

Cloning Penguins: Part 2

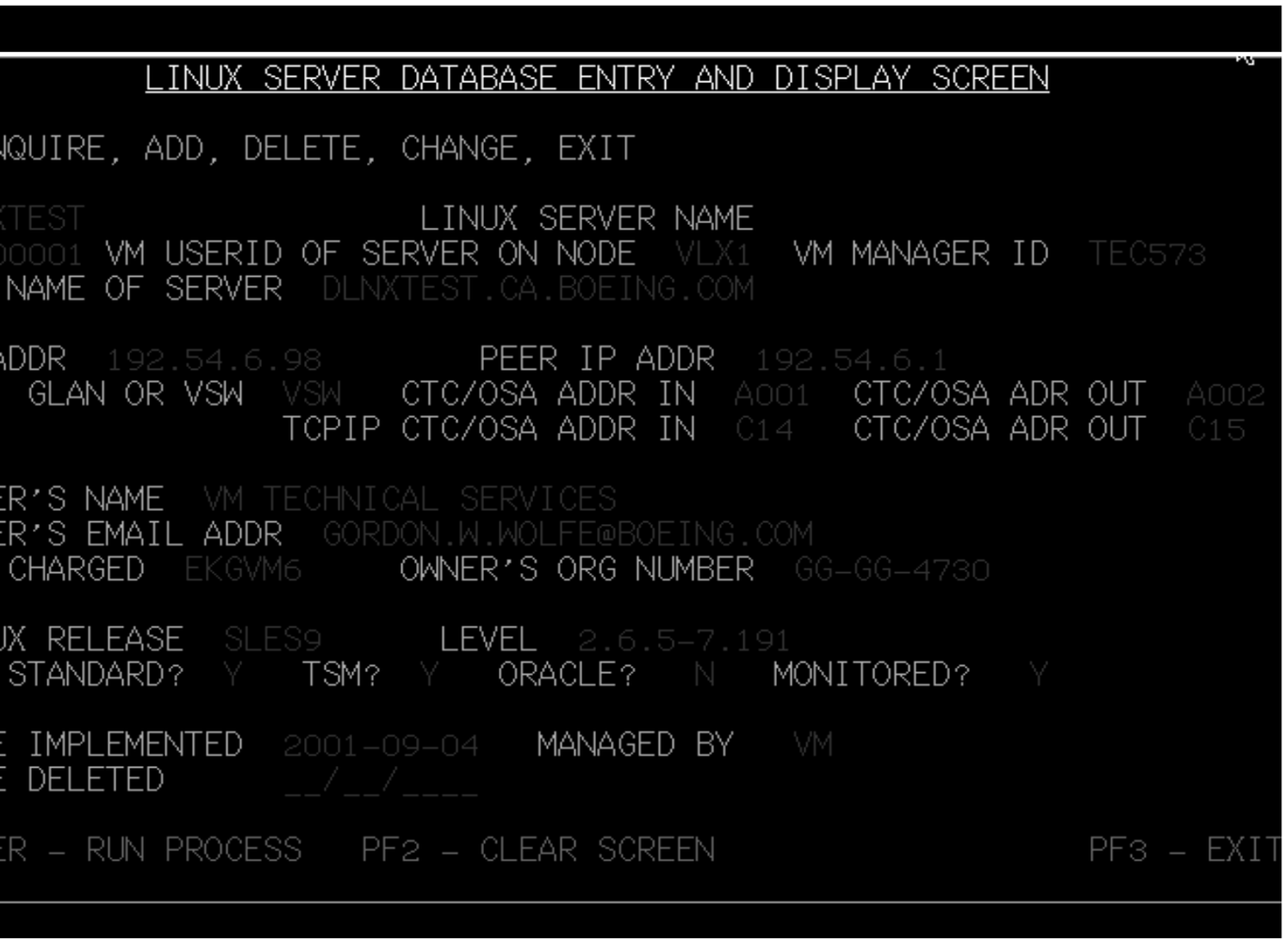
Boot the new server from the reader

- **Mount /dev/dasdb /mnt**
- **Chroot /mnt**
- **Mount /dev/dasdc /usr**
- **dasdfmt /dev/dasdd (and fdasd, mke2fs)**
- **Mount /dev/dasdd1 /home**
- **Update files to make new server (SuSE)**
  - **/etc/rc.config**
  - **/etc/httpd/httpd.conf**
  - **/etc/route.conf**
  - **/etc/hosts**
  - **/etc/fstab**
  - **/etc/smb.conf**
  - **Run /sbin/SuSEconfig**
  - **Run zipl**
- **Exit, umount everything and shut down**

Cloning Penguins: Part 3

- Boot the new server from DASD
- Set up authorities to start and stop server
- Place DNS name in DNS server
- Whole thing takes about 2-3 hours. Can be automated with CMS execs and shell scripts preinstalled on the clone server.
  - See Leland Lucius' E2SH EXEC and E2CMD MODULE (On Sine Nomine's site) which allows editing of Linux ext2 and ext3 filesystems from CMS. You can create an entire exec to clone linux servers! (takes about 15 seconds to run)
- Be Sure to keep a database of servers!
  - DNS, userid, IP address, owner-name, comm method, usage type

Database of Servers



Cloning Software Available (not an endorsement)

VMLINMAN
http://www.glasshousesystems.com/home.html

STK SNAPVANTAGE
http://www.storagetek.com/prodserv/products/software/svan/

LINUXCARE LEVANTA http://www.linuxcare.com

ADUVA at http://www.aduva.com

Unsupported demo software from Tung-Sing Chong at IBM Endicott

- Code at http://www.vm.ibm.com/devpages/chongts/
- Instructions at http://www.vm.ibm.com/devpages/chongts/tscdemo.html

Starting Servers

- Create a server start/stop userid
- Allow SMSG commands via WAKEUP
- Have list of authorized users
- Exec to do actual AUTOLOG of server
- PROFILE EXEC in Clone machine defaults to “boot from disk” if no virtual console attached.
- User/owners of servers require second VM id to start system

Stopping Servers (part 1)

- Put VMPOFF=LOGOFF into parm file /etc/zipl.conf to log off Linux userid when linux O/S quits
- Use journaled filesystems (ext3, reiserfs, jfs) in case something breaks and you can’t shut down cleanly

Stopping Servers. Part 2

- With z/VM 4.3's SERVC facility and Linux 2.4.7 or later
  - Linux can be patched to shut down automatically at CP SHUTDOWN or CP SIGNAL SHUTDOWN
  - Put ca:12345:ctrlaltdel:/sbin/shutdown -t1 -h now in /etc/inittab - One process runs in Linux all the time.
  - Included with SLES8, SLES9

Linux and Guest Lans or VSWITCH

uest Lans are faster and easier than CTC or IUCV, but require z/VM 4.3.0 for nction, and SLES8. VSWITCH is faster and easier yet (no TCPIP stack involv t requires z/VM 4.4 or later and SLES8 or SLES9.

- Up to 700 mb/sec
- Let SYSTEM own the guest lan. CP runs VSWITCH
- Define Guest lans in TCPIP's PROFILE TCPIP
  - DEVICE HIPR3 HIPERS 1F00 PORTNAME LINUXLAN AUTORESTART
  - LINK QDIO3 QDIOIP HIPR3
- Define the guest lans or VSWITCH in AUTOLOG1
  - DEFINE LAN LINUXLAN MAXCONN INF OWNERID SYSTEM TYPE HIPER
- Be sure Linux is set up to use guest lans
  - /etc/sysconfig/network/ifcfg-hsi0 /etc/sysconfig/network/routes
  - /etc/modules.conf /etc/chandev.conf
- In server startup exec, issue
  - CP COUPLE A001 TO SYSTEM LINUXLAN (guest lan)
  - CP COUPLE A001 TO SYSTEM VLINUX1 (vswitch)

Tuning Linux to work with VM

On-Demand Timer: echo "0" >/proc/sys/kernel/hz_timer in SLES8, SLES9

V-DISK for swap, minimum virtual storage (Use the SWAPGEN EXEC from Sine Nomine's site) In SLES8 or SLES9 31-bit, use dasd_diag_mod. Not available in SLES9 64-bit.

MINIOPT CACHE RECORDMDC for real DIAG minidisks. CMS FORMAT/RESERVE + mke2fs. No dasdfmt or fdasd! 13X throughput for 50 increase in I/O CPU.

Use "dasdfmt -d cdl" or reserve first cylinder of pack for VM disk label. This prevents Linux servers from changing the VOLSER of the VM pack. Otherw next time you IPL VM, the Linux server may not work!

- MDISK 294 3390 1 3338 V163E1 MR LINUX USR DASDC

Use INCLUDE files and ACIGROUPS in the directory to better manage

Stagger file-level backups of Linux servers.

If you have SNAPSHOT capability on your DASD

- Use it instead of DDR to create the clone of the boot disk.
- Keep pre-formatted 3390-3 and 3390-9 volumes handy. (dasdfmt and fdasd, but no mke2fs in case you want to use LVM.) Instead of formatting the disk, just SNAPSHOT a copy. 5 seconds instead of 15 minutes per volume.

Updates and Changes

Load patches, CD’s onto a Linux Patch server

- Easier to use than CD or Windows server
- Makes it available to those Linux owners who do their own updates

Update test server first

When working, move to clone server

Route around to production servers

Using YAST2 and VNC, can do updates in place from each server updated.

“Outsource” Linux maintenance to someone else

- Unix gurus
- NT server people
- End user as last resort - use at least for creating own userids and groups
- Set up formal Service Level Agreements to follow.

Backing up Linux with VM

- VM:Backup will do physical (track) backups of Linux, but must restore an entire filesystem disk set to recover one file. E.g. all of /usr.
- Problem is compounded if using LVM for multiple-volume filesystem. Have to restore ALL volumes of a logical volume to restore one file!
- Same for DDR or any other VM backup system. No file-level backups.

Backing up Linux with TSM

TSM (Server on VM or z/OS) works well for file-level backups

- VM server release 3, z/OS at release 5.
- but uses lots of network capacity, especially first time!
- Don’t use software compress in client machine. CPU hog!
- Can do full or incremental, keep multiple generations.
- Stage to DASD first, then move to tape to minimize number of tape drives used.

Other ways to back up Linux at file level

- CA-Brightstor.
- Veritas NetBackup
- Other products on near horizon.
- If tape available to linux, can use amanda.

Roll your own Backups

Give Linux server capability to use BFS, OpenExtensions in directory
BFS server

Use Linux NFS, VMNFS server to mount BFS

Use Tar to back up to NFS-mounted BFS

Use VM:Backup to back up BFS

Can do with 4 shell scripts and 3 files, but very labor intensive, requir
root privilege, and uses LOTS of disk space! (Contact me if you wan
the scripts)

With Neale's cpint package and tape support in 2.4.7 (Both included i
SuSE SLES8) can go direct from Linux direct to tape with "tar" or
"dump", but tape scheduling is a nightmare. Cpint can be used to SM
VMTAPE MOUNT (no response to Linux) and CP DETACH

Accounting

Most Unix-type systems do not do job accounting very well.

Hooks and packages available but require extensive kernel mods

Future kernel may have job accounting in it.

Under VM it's simple! Use a separate server for each account! If you have to share data among servers, use NFS!

VM:Account will create (with exits) and collect charge records for CPU, DASD, BFS, tape mounts, and so on.

VM:Account will also do ad-hoc reports on usage. Cumbersome to set up, but well worth it in the end. Beware! Users will want you to run reports for them!

Close

- VM can be used as “hypervisor” for many linux guests
- The more guests you have, the more work maintaining them is.
- Many members of Linux-VM community have come up with some ideas for managing many servers.
- Commercial software solutions are still on the horizon.
- See Mark Post’s LinuxVM page at http://linuxvm.org
- Join the Linux-390 Listserver! (address on LinuxVM page above)

The Author

- Can be reached at:

vmlinux@locolobo.fastmail.fm

- This presentation and related files can be downloaded from:
  - Http://locolobo.fastmail.fm/download.html

pendix 1 - Starting up Linux

```
------------------------------------------------------------*/
ommon Profile Exec for Linux Server Machine                  */
y Gordon Wolfe, VM Technical Services            05/22/00*/
------------------------------------------------------------*/
ess command

 RDYMSG SMSG'
SET ACNT OFF'
SET RUN ON'
SET RELPAGE OFF'
SET EMSG ON'
LIMIT CLEAR CP'

SET PF11 IMMED FILEL'
SET PF12 RETRIEVE'
TERMINAL LINESIZE 255'
TERMINAL CHARDEL OFF'
SET EMSG ON'

et up for this particular linux server machine                   */
ATE' userid() 'EXEC A'
c <> 0 then exit rc
C' userid()

etermine if we start up Linux now.                               */
tflag = 'N'
lag   = 'D'

re we running disconnected? if so, start linux.                  */
E CP QUERY' userid() '| var usrline'
e value usrline with . . term .
erm = 'DSC' then startflag = 'Y'
```

pendix 1 - Continued

```
ot disconnected? ask to start up.                                    */
lso find out where to start up from.  Reader IPL or DASD ipl.    */
 do
y 'Do you want to start up LINUX now? (Y/N)'
ll ans .
 left(ans,1) = "Y" then startflag = 'Y'
 startflag = 'Y' then do
 say 'Do you want to IPL from DASD or from the reader? (D/R)'
 say 'The default is to IPL from DASD.'
 pull ans .
 select
   when left(ans,1) = 'R' then iplflag = 'R'
   when left(ans,1) = 'D' then iplflag = 'D'
   when ans         = ''  then iplflag = 'D'
   otherwise do
     say ans 'is an invalid choice.'
     exit 8
   end
 end
d


tartflag = 'Y' then do
 iplflag = 'R' then queue 'EXEC SLES7IPL'
 iplflag = 'D' then do
queue '1'
queue 'LXSWAP'
'FORMAT 293 E ( BLK 4096'          /* format/reserve V-disk      */
if rc <> 0 then exit rc            /* for swap space             */
queue '1'
'RESERVE LINUX SWAP E6'
if rc <> 0 then exit rc
queue 'EXEC IPLDASD'
d
```

pendix 1 - Continued

```
ample LINUX002 EXEC for Linux userid LINUX002  */
ess command
COUPLE C16 TO TCPIP C20'
COUPLE C17 TO TCPIP C21’

COUPLE A001 TO SYSTEM LCMEXT'

mfile file Linux userid LINUX002)
isk_size=32768 root=/dev/ram0 ro ctc=0,0xC16,0xC17,ctc0

PLDASD EXEC */
ESS COMMAND
E o
CLOSE RDR'
PURGE RDR ALL'
DETACH 190'
DETACH 19E'
IPL 292 CLEAR’

LES7IPL EXEC                                          */
xec to IPL Linux from the reader and run from a ramdisk */
y Gordon Wolfe, VM Technical Services         08/17/01  */
ess command
e o

o we have enough virtual storage to do this?          */
EBUF'
 = rc
CIO 1 CP ( STRING Q V STOR'
 . . stor .
PBUF' buf1
 = STRIP(stor,'L','0')
 = STRIP(stor,'T','M')
tor < 64 then do
y 'Virtual storage must be 64M or greater to IPL from reader'
y 'Perform CP DEF STOR 64M and IPL CMS.'
it 8
```

pendix 1 - Continued

```
o we have the files we need?                                    */
ATE' userid() 'PARM *'
c <> 0 then do
y 'File' userid() 'PARM * not found.'
it 28


ind the filemode for the files we need                          */
E CMS LISTF SLES7 IMAGE * |',
ake 1 |',
ar imageloc'
e value imageloc with . . fm .
 left(fm,1)

ll looks okay, proceed.                                         */
CLOSE RDR'
PURGE RDR ALL'
SPOOL PUN * R'
CH SLES7 IMAGE' fm '( NOH'
CH' userid() 'PARM' fm '( NOH'
CH SLES7 INITRD' fm '( NOH'
CHANGE RDR ALL KEEP NOHOLD'
IPL 00C CLEAR'
```

pendix 3 - Shutting Down Linux from VM

```
XEC to send shutdown commands to a Linux guest                    */
ssumes root password is same as Linux VM userid password.         */
lso assumes server has a parmline containing vmpoff=LOGOFF        */
serid running this exec must be a VMSECURE administrator          */
y Gordon Wolfe, VM Technical Services                  03/09/2001*/

ess command

server .
E CP Q SECUSER' server '| drop 2 | var bkupsecuser'
c <> 0 then do
y server 'is an unknown Linux server.'
it 8

e var bkupsecuser with oldsecuser .
ldsecuser = 'not' | oldsecuser = 'NOT' then oldsecuser = 'OFF'


 = 'shutdown -h now'

et the secondary userid to ourself.                               */
SET SECUSER' server userid()

et the console logged on as root                                  */
 linuxpwd server
esult <> 0 then exit result

nd send the shutdown command                                      */
SEND' server line

lean up and quit.                                                 */
:
SET SECUSER' server oldsecuser
```

pendix 3 - Continued

```
xpwd: procedure
ess command

linuxmach .
inuxmach = '' | linuxmach = 'LINUXMACH' then do
y 'No Linux machine specified'
turn 8


et the password for the server        in the proper case.       */
name = 'root'
 getpass linuxmach
esult = 28 then do
y 'no password on file for' linuxmach
turn 8

 pw = lowercas(result)


=1 to 3
P SEND' linuxmach rootname
P SLEEP 1 SEC'
P SEND' linuxmach pw
P SLEEP 1 SEC'

rn 0



ass: procedure
rocedure to query VMSECURE for the password of the server  */
finduser .
```

pendix 3 - Continued

```
SE LINUX TEMP A'
EBUF'
 = rc
e 'SSAVE LINUX TEMP A'
e 'QQUIT'
ECURE EDIT' finduser '( NOPROF'
c <> 0 then do
y finduser 'not found in VMSECURE'
RASE LINUX TEMP A'
turn 28

PBUF' buf1
E < LINUX TEMP A |',
ocate /USER / |',
ake 1 |',
pecs word 3 1 |',
ar pw'
SE LINUX TEMP A'
rn pw


owercas                                                          */
ranslates input argument to lowercase                            */
y Gordon Wolfe, Vm Technical Services                  06/23/98 */
rcas: procedure
inp
= translate(inp,'abcdefghijklmnopqrstuvwxyz','ABCDEFGHIJKLMNOPQRSTUVWXYZ','.
rn out
```

pendix 4 - Routing Updates to Servers
om a Central Maintenance Server

```
 /root/updates/hosts - Place the names of the hosts to which updates will be routed
his file.
kname in hosts file or DNS name    VM userid
-------------------------------    ---------
e                                  LINUX000
                                   LINUX001
h                                  LINUX002
erver                              LINUX003


 /root/updates/files - Place the fully-qualified filenames of files that will be rout
he above hosts:
/hosts
/profile.local
uct-20020505.rpm


 /root/updates/precommands - Place commands that you want the above hosts to execute
re moving any files
r /root/test

 /root/update/postcommands - Place commands that you want the above hosts to execute
r moving files:

-Uvh product-20020505.rpm


 run /root/updates/update to send all files to all hosts then execute all commands on
hosts:
unupdate
 updates.rex $1
d 770 runupdate
t/updates/runupdate
```

pendix 4 - Continued

```
 calls the rexx program updates.rex, which is

pdates.rex                                                          */
n exec to create a shell script to take a list of files             */
rom the file ./files and send them to a group of linux servers      */
isted in a file named ./hosts                                       */
hen execute a number of commands taken from a file ./commands       */
alled from shell script "update"                                    */
ssumes:                                                             */
ile names in ./files are fully qualified path names                 */
ost names in ./hosts are resolvable through /etc/hosts              */
he shell script created will be run from userid root                */
serid root on the receiving host has ssh configured to allow        */
ile copies and commands without a password from root on server      */

e off
al off error

e arg onehost .
nehost <> '' then say "Processing for" onehost "only.”

irst, get names of files into stem variable                         */
es = 0
 = queued()
 files >FIFO"
ntil queued() = oldq
rse pull file
 left(file,1) = "#" then iterate
iles = jfiles + 1
rse value file with fname.jfiles fnewname.jfiles .
 fnewname.jfiles = '' then fnewname.jfiles = fname.jfiles
```

pendix 4 - Continued

```
ext, get commands to execute into stem variable                    */
ds  = 0
 = queued()
 commands >FIFO"
ntil queued() = oldq
rse pull file
 left(file,1) = "#" then iterate
cmds  = j1cmds  + 1
rse value file with precommand.j1cmds

mds  = 0
 = queued()
 commands >FIFO"
ntil queued() = oldq
rse pull file
 left(file,1) = "#" then iterate
cmds  = j2cmds  + 1
rse value file with postcommand.j2cmds



ext get list of hosts to send files to                             */
 = queued()
ts=0
 hosts >FIFO"
ntil queued() = oldq
rse pull line
 left(line,1) = "#" then iterate
osts=nhosts+1
rse value line with hostname.nhosts vmuserid.nhosts .
 onehost = hostname.nhosts then khost=nhosts

nehost <> '' then do
userid.1 = vmuserid.khost
osts = 1
stname.1 = onehost
```

pendix 4 - Continued

```
or those hosts that are logged on, build a script                */
s the host logged on?                                            */
=1 to nhosts
dq = queued()
cp q" vmuserid.j ">FIFO"
 until queued() = oldq             /* get just last line        */
parse pull line
d
  Host is not logged on.  Ignore it for now.                     */
 strip(line) <> 'Ready;' then say '+++' hostname.j 'not running.'
se do
  Host is indeed running.                                        */
   Build the shell script to execute the precommands             */
"echo -e echo -e Executing precommands on" host ">> runupdate"
do i=1 to j1cmds
  newline = ”ssh"
  newline = newline “root@”host precommand.i
  "echo -e" newline ">> runupdate"
end                                /* do i=1 to j1cmds          */
  Build the shell script to actually send the files              */
host = hostname.j
"echo -e echo -e                          "  ">> runupdate"
"echo -e echo -e Sending files to" host ">> runupdate"
do i=1 to jfiles
  newline = "rsync -pogt -r -e ssh -l" fname.i
  newline = newline host":"fnewname.i
  "echo -e" newline ">> runupdate"
end
  Build the shell script to execute the postcommands             */
"echo -e echo -e Executing postcommands on" host ">> runupdate"
do i=1 to j2cmds
  newline = ”ssh"
  newline = newline “root@”host postcommand.i
  "echo -e" newline ">> runupdate"
end                                /* do i=1 to j2cmds          */
d                                  /* else do                   */
                                   /* do j=1 to nhosts          */
```