




# Extreme Filesystem Sharing

## Linux on Read-Only Root at Nationwide

Rick Troth <[trothr@nationwide.com](mailto:trothr@nationwide.com)>  
Nationwide Insurance

2008 August 14  
SHARE 110 Session 9216



## Disclaimer

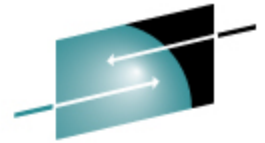
The content of this presentation is informational only and is not intended to be an endorsement by Nationwide Insurance. Each site is responsible for their own use of the concepts and examples presented.

Or in other words: Your mileage may vary. “It Depends.”  
Results not typical. Actual mileage will probably be less. Use only as directed. Do not fold, spindle, or mutilate. Not to be taken on an empty stomach.

When in doubt, ask! Still in doubt? Test it!

# Extreme Filesystem Sharing

- Herding the Flock
- Sharing Common Content
- A Shared Root Directory
- System maint and package management
- Relocatable Packages
- DASD on Demand – Disk Automounter
- DCSS on Demand – Virtual ROM Automounter

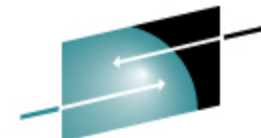


**S H A R E**

Technology • Connections • Results

# Issue: Penguins Populating

one by one, the penguins slowly  
steal my sanity...



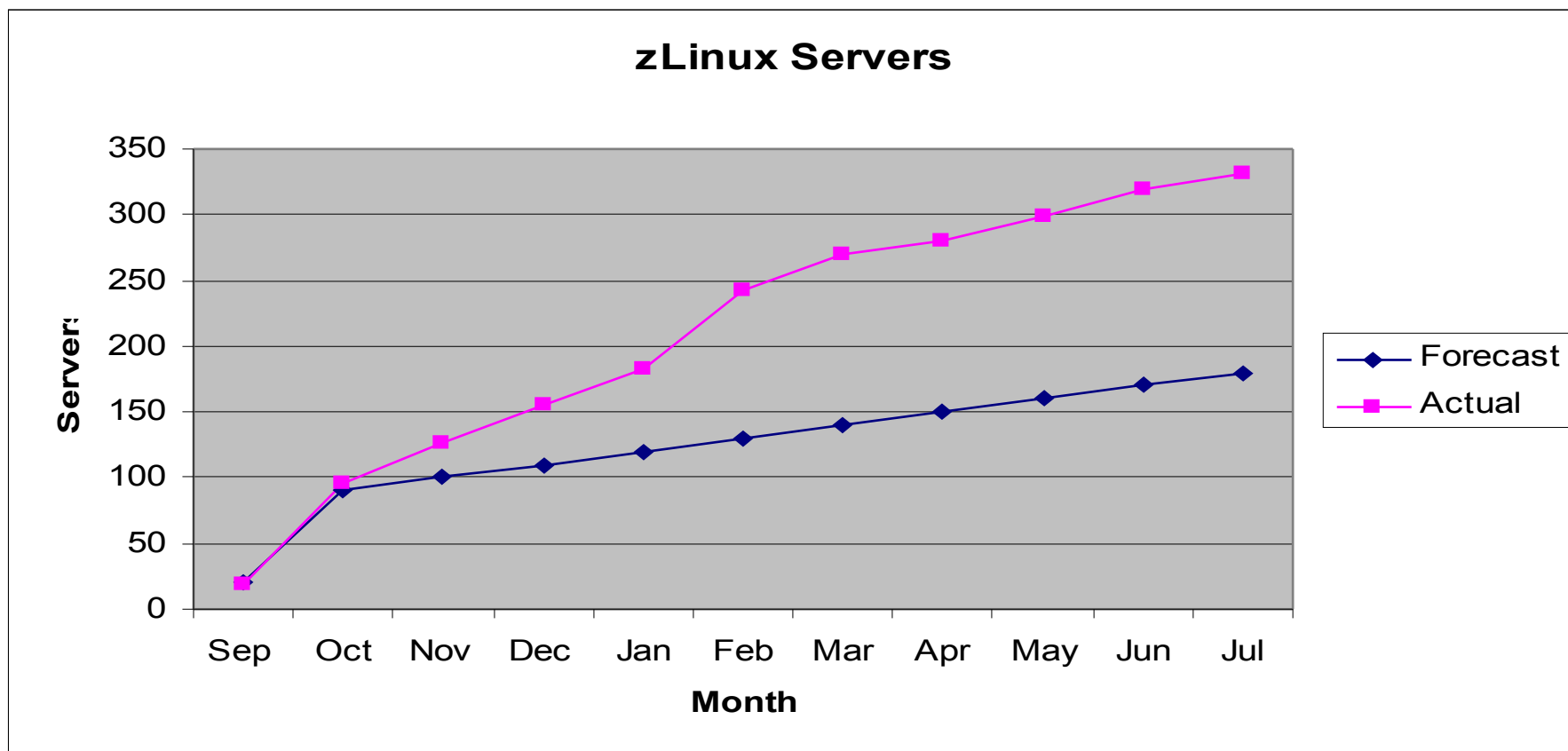
**S H A R E**

Technology • Connections • Results

# Issue: Penguins Populating

*And I thought we were busy **before** we got Linux!*

Rick Barlow, Aug 1, 2006



## Solution: Share More Stuff

- Install Once, Run Many
- An old Gospel, a prophecy fulfilled!
- We have been sharing `/usr`, `/opt`, and others so why not also share the root?

# Untouchable root? Sounds Weird

- Solaris/SunOS does NFS root including read-only `/usr` content
- “Live CD” Linux uses bulk R/O content
  - Knoppix, Ubuntu, Kubuntu, recovery tools
- USS does ROR already (Unix on z/OS)

Not weird, Not even new

People still ask, “Why???”

# Stability and Manageability

- R/O media is incorruptible
- R/O content is centrally maintained
- R/O packages are available on-demand
- Better D/R – less per-server replication, less to back up
- WYTIWYG

R/O zLinux no different from R/O PC Linux



# Shared OpSys Partitions

- Multiple R/O shared disks
  - Selected filesystems can revert to R/W
  - boot, root, `/usr`, and `/opt`
- Up to three partitions per disk
  - Remains true for CKD (CDL), FBA, and SAN
- Glacial stability
  - Glaciers move, but you can walk on them without drowning

## How to ... reference

1b0 == boot

1b1 == root

1b5 == /local

1be == /usr

1bf == /opt

2b0-2bf == LVM phys vols and/or maint

320-33f == “User Space” LVM phys vols

100,200 == FCP “HBAs” for SAN

## How to ... reference

**1b0 == boot, bootable and /boot**

**1b1 == root**

**1b5 == /local**

contains **etc**, **dev** and others

is the “personality” of the system

**1be == /usr**

**1bf == /opt**



**SHARE**

Technology • Connections • Results

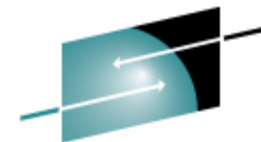
# How it Looks / How it Works

```
szvmjt005 # df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/dasdb	278960	108424	156136	41%	/
tmpfs	124696	0	124696	0%	/dev/shm
/dev/dasda1	21512	18232	2172	90%	/boot
/dev/dasdo	1231672	610656	558448	53%	/usr
/dev/dasdp	161088	952	151820	1%	/opt
tmpfs	124696	0	124696	0%	/tmp
/dev/dm-0	253920	82840	157976	35%	/var
/dev/dm-1	253920	121804	119012	51%	/home
/dev/dasdbn1	23216172	20420196	1616660	93%	/dasd/340

```
szvmjt005 # touch /FFFF
```

```
touch: cannot touch `/FFFF': Read-only file system
```



# How it Looks / How it Works

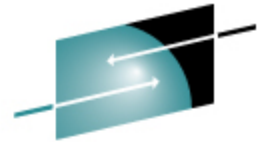
```
szvmjt005 # df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/dasdb	278960	108424	156136	41%	/
tmpfs	124696	0	124696	0%	/dev/shm
/dev/dasda1	21512	18232	2172	90%	/boot
/dev/dasdo	1231672	610656	558448	53%	/usr
/dev/dasdp	161088	952	151820	1%	/opt
tmpfs	124696	0	124696	0%	/tmp
/dev/dm-0	253920	82840	157976	35%	/var
/dev/dm-1	253920	121804	119012	51%	/home
/dev/dasdbn1	23216172	20420196	1616660	93%	/dasd/25f

```
szvmjt005 # df /local
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
-	209216	75688	122728	39%	/local
/dev/dasdf	209216	75688	122728	39%	/local

/dev, /etc, and /root all live under /local



**S H A R E**

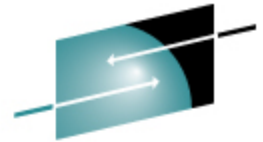
Technology • Connections • Results

# How it Looks / How it Works

```
szvmjt005 # cat /proc/mounts
rootfs / rootfs rw 0 0
/dev/root / ext2 ro 0 0
proc /proc proc rw 0 0
sysfs /sys sysfs rw 0 0
devpts /dev/pts devpts rw 0 0
tmpfs /dev/shm tmpfs rw 0 0
tmpfs /tmp tmpfs rw 0 0

/dev/dasda1 /boot ext2 ro 0 0
/dev/dasdo /usr ext2 ro 0 0
/dev/dasdp /opt ext2 ro 0 0

/dev/dasdf /local ext3 rw 0 0
/dev/dasdf /etc ext3 rw 0 0
/dev/dasdf /root ext3 rw 0 0
/dev/dasdf /dev ext3 rw 0 0
```



# How it Looks / How it Works

```
szvmjt005 # mount
/dev/dasdb on / type ext2 (ro)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,mode=0620,gid=5)
tmpfs on /dev/shm type tmpfs (rw)
tmpfs on /tmp type tmpfs (rw)
```

```
/dev/dasda1 on /boot type ext2 (ro)
/dev/dasdo on /usr type ext2 (ro)
/dev/dasdp on /opt type ext2 (ro)
```

```
/dev/dasdf on /local type ext3 (rw)
/local/etc on /etc type bind (rw)
/local/dev on /dev type bind (rw)
/local/root on /root type bind (rw)
```



# How it Looks / How it Works

## Define mount points for R/W stuff.

for DB2/UDB ...

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/mapper/wdvg--db2test-lvdb2bin					
	3096336	372752	2566300	13%	/opt/IBM/db2
/dev/mapper/wdvg--db2test-db2fslv					
	1032088	519140	460520	53%	/db2fs
/dev/mapper/wdvg--db2test-db2logfslv					
	1032088	32876	946784	4%	/db2logfs

for WAS ...

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/mapper/3390-33901v01					
	3096336	1455196	1515312	49%	/u01
/dev/mapper/3390-33901v02					
	1548144	563456	906048	39%	/webdata



# What we Changed

- Move `bp.conf` to a non-shared place:

```
cd /usr/opensv/netbackup  
mv bp.conf /etc/.  
ln -s /etc/bp.conf .
```

- Move LVM lock file:

One line change to `/etc/lvm/lvm.conf`

While still R/W

# What we Changed

- Move `init.d` to a shared place:

```
cd /etc
mv init.d ../sbin/
ln -s ../sbin/init.d .
```

- Modify `/etc/init.d/boot` script:

```
#bootrc=/etc/init.d/boot.d
bootrc=/sbin/init.d/boot.d
```

While still R/W

# What we Changed

Replace `boot.rootfsck` with `boot.readonlyroot`

- Does not check root (`1b1` disk)
- Checks and mounts `/local` (`1b5` disk)
- Bind mounts `/etc`, `/dev`, and `/root`
- Happens during the “boot” run level

This is the R/W to R/O switch

*This is the point of No Return*

## `/etc/init.d` Overlay During Startup

`/etc/init.d/boot.d` is special

- Customer cannot change (real) `boot.d`
- Runtime `'chkconfig'` appears to change `boot.d`
- Runtime `boot.d` is R/W but not used at boot time
- Actual `boot.d` is `/sbin/init.d/boot.d`
- All other run-levels work normally (R/W is boot time copy)

# /etc/init.d Overlay During Startup

```
# cd /etc/init.d/boot.d ; ls -la
S01boot.proc -> ../boot.proc
S02boot.shm -> ../boot.shm

S03boot.readonlyroot -> ../boot.readonlyroot

S06boot.device-mapper -> ../boot.devi...
S06boot.md -> ../boot.md
S07boot.lvm -> ../boot.lvm
S08boot.localfs -> ../boot.localfs
```

# Bind Mounts

- Scripted ...

```
mount -n -o bind /local/etc /etc
```

```
mount -n -o bind /local/dev /dev (goes away with SLES10)
```

```
mount -n -o bind /local/root /root
```

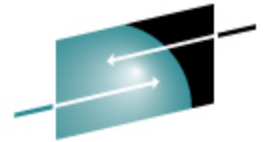
- Specified in `/etc/fstab` ...

```
/local/var /var bind bind 0 0
```

```
/local/home /home bind bind 0 0
```

## Even Easier in SLES10

```
6c6
< # /etc/init.d/boot.rootfsck
---
> # /etc/init.d/boot.roroot
96,97c96,99
<         echo "Checking root file system..."
<         fsck $FSCK_PROGRESSBAR -a $FSCK_FORCE $ROOTFS_BLKDEV
---
> #*             echo "Checking root file system..."
> #*             fsck $FSCK_PROGRESSBAR -a $FSCK_FORCE $ROOTFS_BLKDEV
>                 echo "Checking /local file system..."
>                 fsck $FSCK_PROGRESSBAR -a $FSCK_FORCE /local
```



## Even Easier in SLES10

150c152,156

```
<          mount -n -o remount,rw /
```

```
---
```

```
> #*          mount -n -o remount,rw /
```

```
>          mount -n /local
```

```
>          mount -n -o bind /local/etc /etc
```

```
>          mount -n -o bind /local/root /root
```

162c168,172

```
<          mount -n -o remount,rw /
```

```
---
```

```
> #*          mount -n -o remount,rw /
```

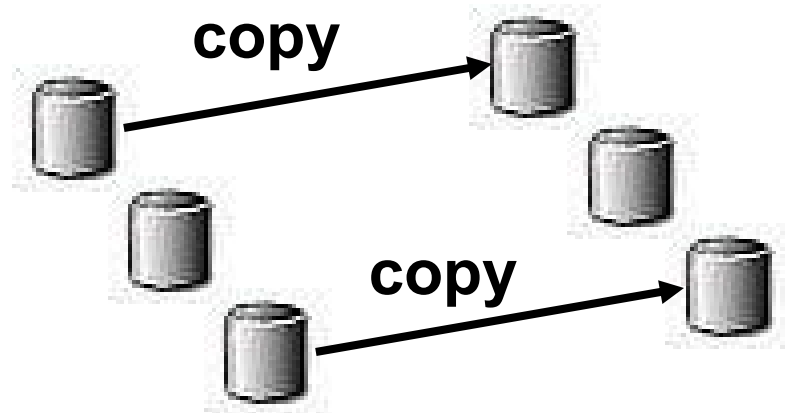
```
>          mount -n /local
```

```
>          mount -n -o bind /local/etc /etc
```

```
>          mount -n -o bind /local/root /root
```

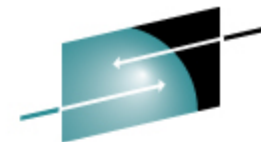


# How to Build Read-Only Root



**target disk set  
intended to be shared  
roughly MAINT 19x  
apply ROR switch here  
(`boot.readonlyroot` script)**

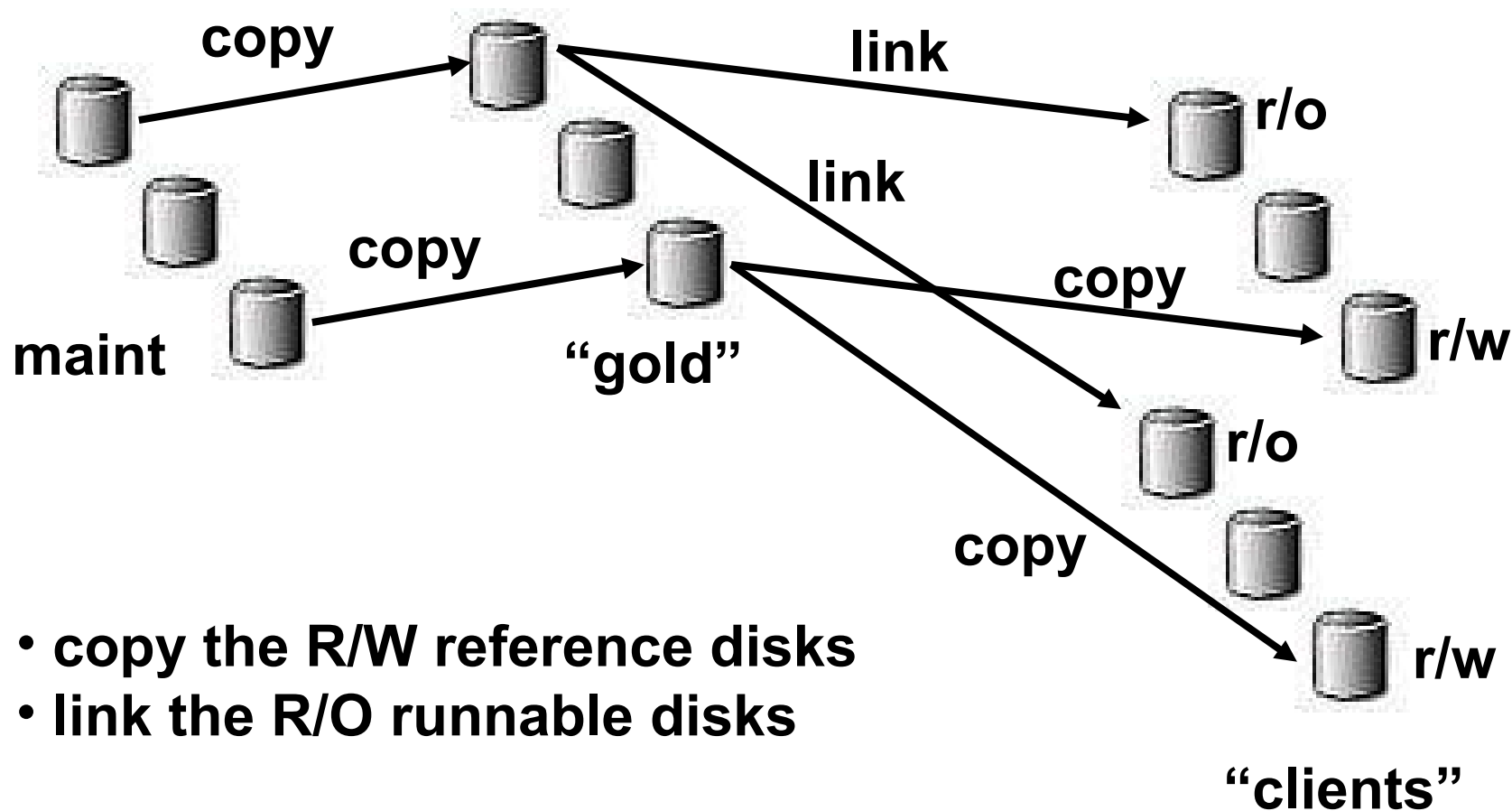
**source disk set  
use distributor tools as-is  
roughly MAINT 49x**



**SHARE**

Technology • Connections • Results

# How to Build Read-Only Root



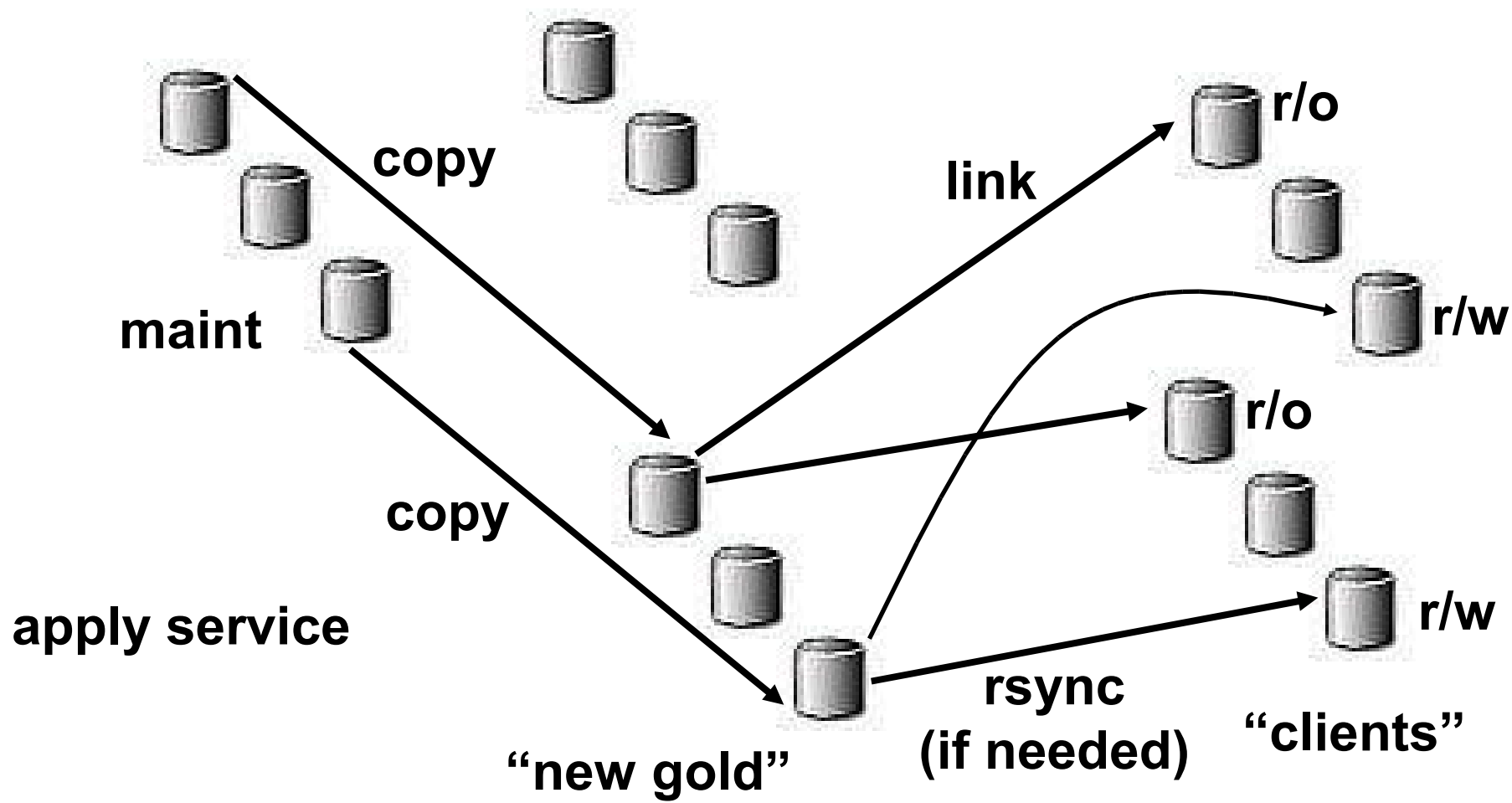
- copy the R/W reference disks
- link the R/O runnable disks

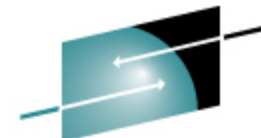


**SHARE**

Technology • Connections • Results

# How to Build Read-Only Root

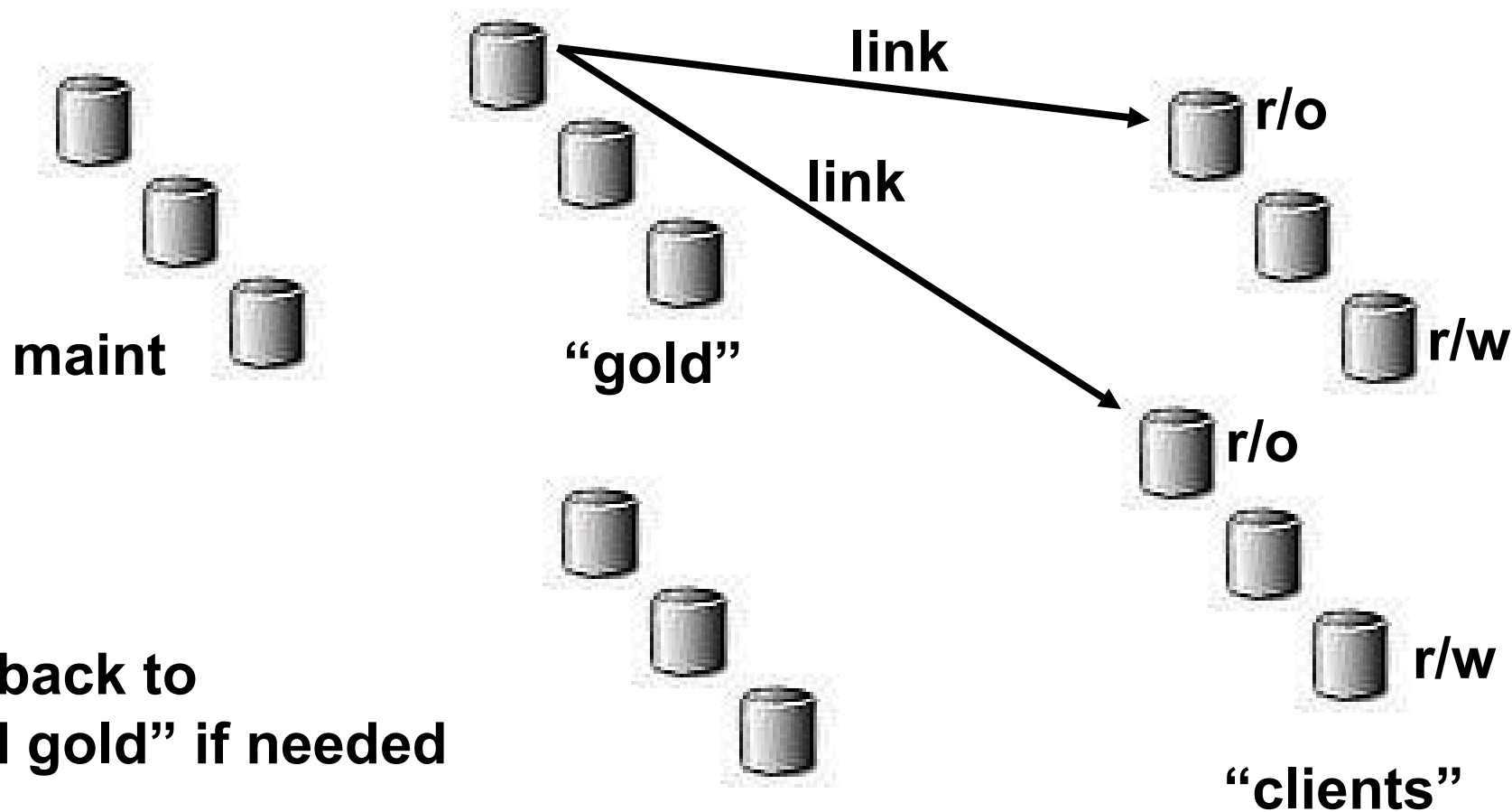




**SHARE**

Technology • Connections • Results

# How to Build Read-Only Root



# How to Build Read-Only Root

- Start with unmodified distributor's installation
- Minor prep (see prior slides)
- Create a reference `/local` (with `/etc` and so forth)
- Replace `boot.rootfsck` with `boot.readonlyroot`
- Copy to eventual R/O target disks (and R/W ref disks)

<http://www.redbooks.ibm.com/abstracts/redp4322.html>



**S H A R E**

Technology • Connections • Results

# Reconciling RPM Database

- Initial RPM DB matches master
- “Client” systems may vary
- Master may get updates

... now what?

# Reconciling RPM Database

- Extract master package list

```
# rpm -q -a > master.rpm1
```

- Update client RPM database

```
# for P in `cat master.rpm1` ; do  
    rpm -U --justdb $P.rpm ; done
```



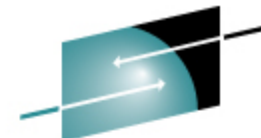
# CYGWIN on CD-ROM

- Why not other environments besides Linux?

```
nw848944 # df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
D:\CYGWIN	405150	405150	0	100%	/
D:\CYGWIN\bin	405150	405150	0	100%	/usr/bin
D:\CYGWIN\lib	405150	405150	0	100%	/usr/lib
C:\etc	28678528	19825140	8853388	70%	/etc
C:\var	28678528	19825140	8853388	70%	/var
C:\tmp	28678528	19825140	8853388	70%	/tmp
C:\docume~1	28678528	19825140	8853388	70%	/home
C:\progra~1	28678528	19825140	8853388	70%	/opt
C:\boot	28678528	19825140	8853388	70%	/boot

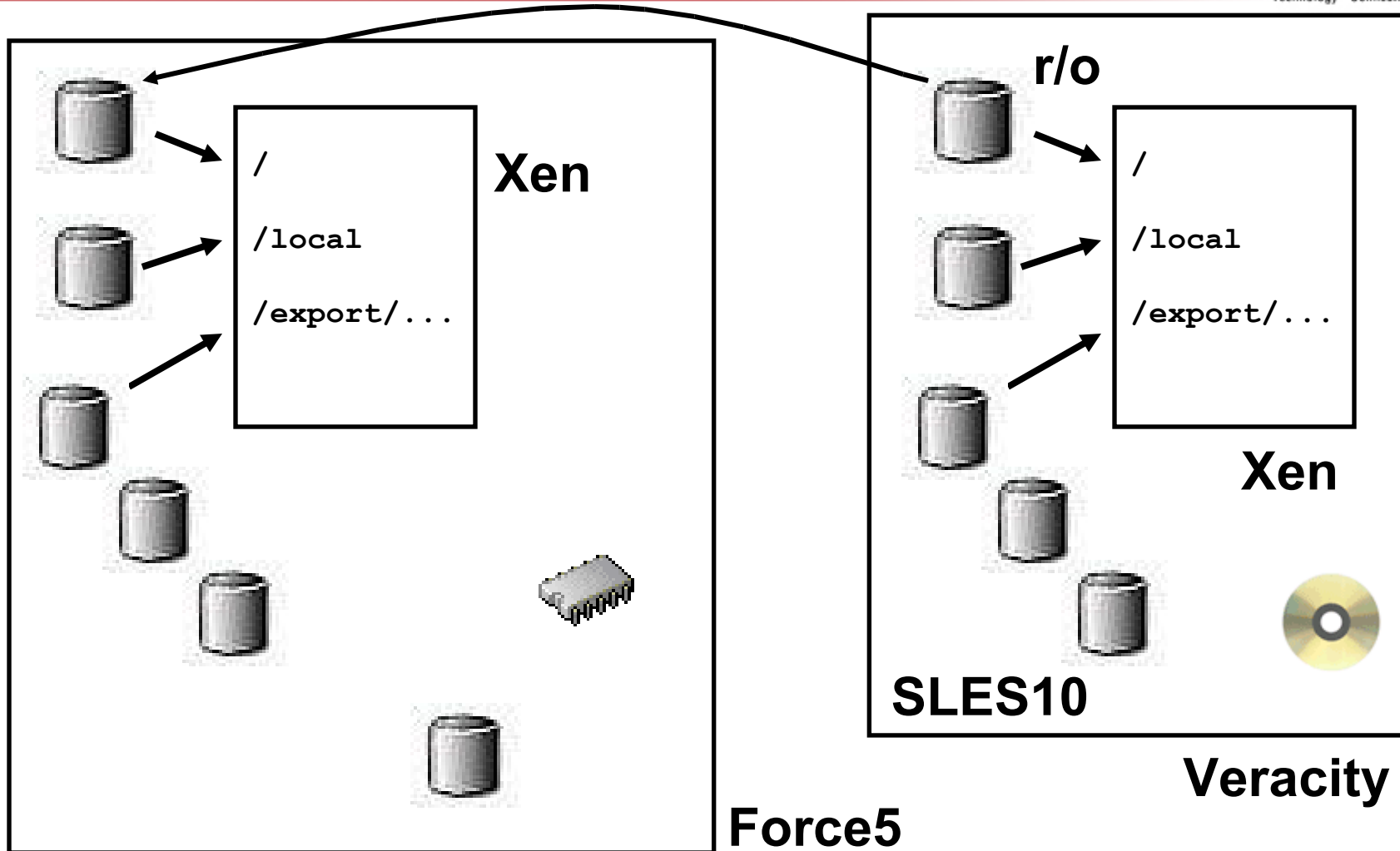




**SHARE**

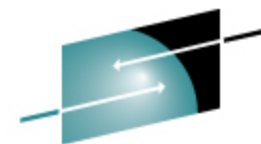
Technology • Connections • Results

# Casita.Net “main”



## Casita.Net “main”

- “main” is an alias; actual hostname varies
- Services: IMAP, SMTP, DNS, NFS, CUPS, NTP, ...
- Host systems differ, “main” guest identical on both
- `/usr` and `/opt` not split in this project
- “service” filesystems replicated



**S H A R E**

Technology • Connections • Results



# Relocatable Packages

## On-Demand Software, Ready to Run



# Relocatable Packages

- Immediate deployment
- Simplified back-out
- Non-intrusive
- Multiple release concurrency
- Variable platform detail (per build)
- Reduced “scatter”
- Think `‘vmlink’`

## Relocatable Packages – versus today

currently we enjoy (ie: read-write) ...

- Packages must be [re]deployed on each system
- Deployment causes multiple disruptions
- Installation demands private (local, R/W) file storage
- Upgrade and/or removal is “messy”
- Installed files are vulnerable
- More things need to be backed up

# Relocatable Packages

we can have (with read-only) ...

- Instant Deployment
- Protected copies (R/O to each client)
- Less content to be backed up
- Non-intrusive (to the guest op sys)
- Non-disruptive (to the users and work)
- Mixed releases as needed

# Relocatable Packages

sharing options ...

- NFS (LAN or WAN) (or AFS for planetary)
- SMB (eg: SAMBA)
- VM minidisk (usually the same DC)
- SAN (usually the same datacenter)
- DCSS (same DC, same CEC, same LPAR)

R/O packages do not require R/O root

## Relocatable Packages – How

- Distinguish ***software residence*** from ***software reference***
- Inst must distinguish program from data
- Installation must tolerate running from R/O filesystems
- Package must be runnable from a customer-settable locatoin



# Relocatable Packages – Concept

`$APPROOT/bin`

`$APPROOT/lib`

`$APPROOT/otherstuff`

`APPROOT=/usr/opt/x3270-3.3`

- Use *package-version* syntax or similar

## Relocatable Packages – Build

If you have the source, it is probably trivial!

What is the “standard recipe”?

- `extract`
- `./configure --prefix=$APPROOT`
- `make`
- `make install`

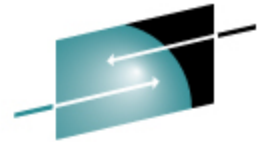
# Relocatable Package Example

Build with the standard recipe:

- `extract`
- `./configure --prefix=/usr/opt/x3270-3.3`
- `make`
- `make install`

`/usr/opt` is ready and writable.

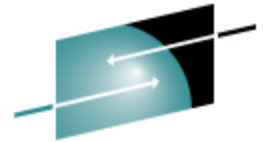
But ... you intercept `/usr/opt/x3270-3.3` to your own place!



# Relocatable Package

```
$ ls -atl /home/trothr/x3270-3.3

drwxr-xr-x 6 trothr ... CYGWIN
drwxr-xr-x 6 trothr ... Linux-s390x
drwxr-xr-x 6 trothr ... Solaris-sparc
drwxr-xr-x 7 trothr ... x3270-3.3
lrwxrwxrwx 1 trothr ... src -> x3270-3.3
-rwxr--r-- 1 trothr ... makefile
-rwxr-xr-x 1 trothr ... setup
```



# Relocatable Package Example

```
$ /home/trothr/x3270-3.3/setup
```

... does this ...

```
+ ln -s  
  /home/trothr/x3270-3.3/Solaris-sparc /usr/opt/x3270-3.3  
+ ln -s x3270-3.3 /usr/opt/x3270  
+ ln -s /usr/opt/x3270/bin/x3270 /usr/bin/.  
+ ln -s /usr/opt/x3270/bin/x3270if /usr/bin/.  
+ ln -s /usr/opt/x3270/bin/pr3287 /usr/bin/.
```

# Relocatable Pkgs – Multiple Versions

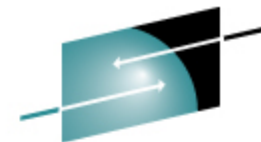
```
lrwxrwxrwx ... gcc -> gcc-3.2.3 (production)
```

```
lrwxrwxrwx ... gcc-3.2.3 ->  
  /import/opt/gcc-3.2.3/Linux-s390x
```

```
lrwxrwxrwx ... gcc-3.4 ->  
  /auto/apps/gcc-3.4/Linux-2.6-s390x
```

- Simple `PATH` change to access the variant:

```
PATH=/usr/opt/gcc-3.4/bin:$PATH
```



**SHARE**

Technology • Connections • Results



# Disk and DCSS Automount On-the-Fly Mainframe Media



# Disk Automounter: Purpose

Automate best practice media access

- z/VM supports dynamic devices
- Linux supports dynamic devices but with different semantics
- Automounter bridges the gap and eliminates operator error



# Disk Automounter: Misconceptions

NOTE: DOES NOT REQUIRE NFS

- Most automounter is for networked FS
- Other FS also good for on-demand use (CD-ROM, flash media, USB disk, etc)
- No network requirement in automounter

# Dynamic Disk on Linux on z/VM

How it works, manually:

- Attach the disk (`hcp link`)
- Find where Linux slotted it
- Vary it on-line (`chccwdev`)
- Mount it

Convolutated and error prone

# Automating Disk Attachment

```
#  
# /etc/auto.master  
#  
/home    /etc/auto.home  
/misc    /etc/auto.misc  
/dasd    /etc/auto.dasd  
/dcss    /etc/auto.dcss
```



# Automating Disk Attachment

```
# parse off the partition number, if any:
PART=`echo "$1" | awk -F. '{print $2}'`

# normalize the device number:
DASD=`echo "0000$1" \
| awk -F. '{print $1}' \
| tr A-Z a-z \
| awk '{print "0.0."
      substr($1,length($1)-3,4)}'`
```

# Automating Disk Attachment

```
# find the pseudo file to control this dev:
CTRL=`ls -d
    /sys/devices/css0/*/DASD/online
    2>/dev/null | head -1`

# is the disk on-line (is it ATTACHED)?
if [ ! -f "$CTRL" ] ; then
    hcp "link * $DASD $DASD rr"
    # and re-set CTRL shell var as above
fi
```

# Automating Disk Attachment

```
# vary it on-line to Linux:  
echo 1 > $CTRL  
  
# and find the block dev assigned:  
BDEV=`ls -d  
    /sys/devices/css0/*/DASD/block  
    2>/dev/null | head -1`  
  
# also clean-up that file path
```

# Automating Disk Attachment

```
# voi-la! create a directory and mount it
```

```
mkdir -p -m 555 $1
```

```
# mount command varies per the following
```

- Unqualified, try partition 0 or partition 1
- Qualified partition 1, 2, or 3, try as-is
- Qualified partition 0 is “the whole disk”



# Disk Automounter Examples

```
zservx01:~ # df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/dasde1	7098008	817616	5919824	13%	/
tmpfs	124700	0	124700	0%	/dev/shm
/dev/dasda1	52200	8940	40568	19%	/boot

Initial state of the system



# Disk Automounter Examples

```
zservx01:~ # cd /dasd/25f/sles9
```

```
zservx01:/dasd/25f/sles9 # df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/dasde1	7098008	817616	5919824	13%	/
tmpfs	124700	0	124700	0%	/dev/shm
/dev/dasda1	52200	8940	40568	19%	/boot
/dev/dasdg1	23216172	18301524	3735332	84%	/dasd/25f

Automounter did the following:

- Found the “25F” disk, varied it on-line
- Happened to find slot “dasdg” and partition 1
- Mounted FS in the requested location



# Disk Automounter Examples

```
vst $ df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/dasdb2	222464	98332	112648	47%	/
/dev/dasda1	20908	8948	10880	46%	/boot
/dev/dasda2	2126020	531716	1486304	27%	/usr
/dev/dasda3	214096	27624	175420	14%	/opt
tmpfs	124700	20	124680	1%	/tmp
/local/home	104608	34944	64264	36%	/home
/local/var	104608	34944	64264	36%	/var

Initial state (round two)



# Disk Automounter Examples

```
vst $ cd /dasd/1bd.1 ; cd /dasd/1bd.2 ; cd /dasd/1bd.3
```

```
vst $ df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/dasdb2	222464	98336	112644	47%	/
/dev/dasda1	20908	8948	10880	46%	/boot
/dev/dasda2	2126020	531720	1486300	27%	/usr
/dev/dasda3	214096	27624	175420	14%	/opt
tmpfs	124700	0	124700	0%	/tmp
/local/home	104608	34976	64232	36%	/home
/local/var	104608	34976	64232	36%	/var
/dev/dasdn1	849696	24752	781780	4%	/dasd/1bd.1
/dev/dasdn2	566936	7140	530996	2%	/dasd/1bd.2
/dev/dasdn3	948184	92696	807320	11%	/dasd/1bd.3

The “doc disk”: `man, info, doc`

# Automating DCSS Attachment

```
#  
# /etc/auto.master  
#  
/home    /etc/auto.home  
/misc    /etc/auto.misc  
/dasd    /etc/auto.dasd  
/dcss    /etc/auto.dcss
```



# Automating DCSS Attachment

```
# first try to mount the requested segment as XIP
modprobe xip2 1> /dev/null 2> /dev/null
mount -t xip2 -o ro,memarea="$1" "$1" $1 2> /dev/null
```

- EXecute-In-Place increases sharing of real memory
- Programs are not loaded from FS into memory, but are already *in memory* and simply executed where they sit
- XIPFS filesystem driver does all the DCSS work

# Automating DCSS Attachment

```
# if XIP fails, then activate the DCSS we are after
echo "$DCSSNAME" > /sys/devices/dcscblk/add

# find the block device for that segment
DCSSBDEV=`ls -ld /sys/devices/dcscblk/$DCSSNAME/block* \
  2> /dev/null | xargs -r -n 1 | tail -1 | xargs -r basename`
DCSSBDEV=/dev/$DCSSBDEV

# mount it the old fashioned way
mount -t ext2 -r $DCSSBDEV $1 2> /dev/null
```

# Extreme Filesystems Summary

## Sharing Common Content

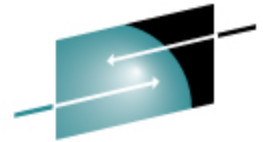
- Coordinated Operating System Configuration
- Centralized System Maint and Package Management
- On Demand Relocatable Packages
- On Demand Shared Content
  - `auto.dasd` for automounted disk
  - `auto.dcss` for automounted “Virtual ROM”

# Summary

- The real advantage is *not* space savings but is management of myriad systems
- Start with one read-only package or directory or disk and grow from there



*Thank You!!*



**S H A R E**  
Technology • Connections • Results



**Richard Troth**  
Senior VM Systems Programmer

**Nationwide Services Co., LLC**

One Nationwide Plaza, MB-02-201

Columbus, OH 43215-2220

Voice: 1-614-249-7642

Cell: 1-614-849-8255

[trothr@nationwide.com](mailto:trothr@nationwide.com)