ro-root with SUSE Linux Enterprise Server

experiences and best practices

Ihno Krumreich
Project Manager for Linux on System z
Ihno@SUSE.de
SHARE Atlanta 2012
Session 10741

ro-root with SUSE Linux Enterprise Server
Agenda

- Motivation / History
- Requirements
- Convert an existing System
- Maintenance

Motivation

ro-root with SUSE Linux Enterprise Server
Motivation

- Reuse the same disk image multiple times
  - Reduce needed disk space
  - Speedup of application start / operation
- Control, which files are modified during operation
- Ease the administration of machine, which have the same software installed, but differ only in user data

ro-root with SUSE Linux Enterprise Server
Motivation

Red paper showing a setup with SLES 10
see
http://www.redbooks.ibm.com/abstracts/redp4322.html

- Shows how to create a ro-root system
- How to clone an existing system
- how to adapt such a system

ro-root with SUSE Linux Enterprise Server
Motivation

DCSS and xip2fs

- **D**is**C**ontinous **S**aved **S**egment is a memory segement saved to the spool area, which can be loaded be the linux guest
- xip2fs (e**x**ecute **i**n **p**lace filesystem) is a filesystem developed by IBM. It is a readonly filesystem, where read operation are done by returning the memory address, not by copying the data. The filesystemlayout is identical to ext2, so all tools for ext2 can be used.
- see “Saving real storage with xip2fs and DCSS”, Session 9230, SHARE August San Jose 2008

ro-root with SUSE Linux Enterprise Server
Motivation

readonlyroot is now integrated into

SUSE Linux Enterprise Server 11

Requirements

ro-root with SUSE Linux Enterprise Server
requirements

ro-root on SLES 11 SP1 / SP2

- a scratch partition / disk for data that can be deleted at every system reboot
- a state partition / disk for data that is specific to the instance and need to survive a reboot of the system
- adjust /etc/rwtab to your needs. This defines which files / directories on the readonly system need to be replaced by files / dreictories on the state / scratch disk
- modified kernel command line
  readonlyroot scratch=/dev/dasdb1 state=/dev/dasdc1

Convert an existing System

ro-root with SUSE Linux Enterprise Server
convert an existing system

- find out which files need to be read/write
- add the scratch and state disk/partition to the system, format them and put a filesystem on.
- edit /etc/rwtab and put the files/directories in that need to be read/write
- modify /etc/zipl.conf, run zipl
- reboot the system

ro-root with SUSE Linux Enterprise Server
analyse an application

- Problem -- How to analyse arbitrary applications to find out where they need writable files to work.

  ==> Use Apparmor to find out which files are accessed readonly and which read/write

ro-root with SUSE Linux Enterprise Server
add scratch and state disk

- use YaST dasd to add the disks to the system (Lets call the partitions dasdb1 and dasdc1)
- mkfs -t ext2 /dev/dasdb1
- mkfs -t ext2 /dev/dasdc1

ro-root with SUSE Linux Enterprise Server
/etc/rwtab

the following four entries may be in /etc/rwtab:

- empty – create a directory (and all needed parents) on the scratch file system. do nothing if this directory does not exist on the ror system
- dirs – like empty, but copy all directories of the ror-directory to the scratch directory
- files – like empty, but copy all files and directories to the scratch directory
- state – if the file/directory exists in the ror and the state system bind mount them. If the file/directory do not exist and INIT_STATE_AUTOMATICALLY is set, init state system by copying the ror files

ro-root with SUSE Linux Enterprise Server
/etc/sysconfig/readonlyroot

```
# Where to mount scratch writable space
SCRATCH_MOUNT=/var/lib/readonlyroot/scratch
# Mount options to use for the scratch filesystem
SCRATCH_MOUNT_OPTS=
# Where to mount persistent writable data
STATE_MOUNT=/var/lib/readonlyroot/state
# Mount options to use for the state filesystem
STATE_MOUNT_OPTS=
INIT_STATE_AUTOMATICALLY=yes
```

ro-root with SUSE Linux Enterprise Server
/etc/zipl.conf

- **add the following string to parameter in /etc/zipl.conf**

  readonlyroot scratch=/dev/dasdb1 state=/dev/dasdc1
- run zipl to make change activ
- and finally reboot the system

Maintenance

Setup of multiple Guests with ro-root

A “Goldmaster” is created

- it contains all RPMs, application needed for all systems
- All guests are created by linking the Goldmaster and doing modifications to the read/write part which is on the state disk.
- the Goldmaster is linked to the guests that should run it.

How to do Maintenance?

Maintenance for a standard Linux guest:

- download the RPM from the vendor side
- shutdown services/applications touched by the RPM
- apply the RPM
- restart services / guest (depending on the RPM)

  --> RPM has full wirte access to the system

  --> pre and post installscripts have full wirte access

Problems doing maintenance to readonly-root system?

With readonly-root the majority of the system is readonly and could not be updated.

- Part of the system is readonly
- Part of the system is read/write
- The readonly part exists only once
- the read/write part has to be updated n times
- the read/write part may be modified by the sysadmin

How to do Maintenance ?

- The goldmaster (GM1) is copied to create a new goldmaster (GM2)
- run the cmp command on the files of GM1 and GM2 to find out, which file have been changed. Call this list GMDIFF
- remove all files from this list which are not listed in /etc/rwtab and call this list GMDELTA
- The files in GMDELTA need to be checked by a person
- after the changes have been applied to the state disk. the readonly-root could be linked to the guest.

Questions??

Thank you.


suse®