Volume Management in Linux with EVMS

Kevin Corry
Steve Dobbelstein
April 21, 2003

http://evms.sourceforge.net/
Overview

- Volume management basics.
  - Variety of types of volume management.
- Kernel drivers that help implement volume management.
- EVMS
  - Overview.
  - How does EVMS work with the kernel drivers.
What is volume/storage management?

- Provide a logical abstraction of the physical storage devices.
- Filesystems and applications do not need to know about the organization of the physical devices.
Where To Use Volume Management

• Systems with lots of physical storage
  – Lots of disks (tens, hundreds, thousands).
  – Combine many disks into a single pool of storage.
    • Increased total storage space.
    • Redundancy to protect against hardware failures.

• Systems with little physical storage
  – Single disk (most PCs, laptops).
  – Divide up disk to provide logically separate storage for different uses.
Disk Partitioning

- Divide a disk into one or more logical sections.
- Simple, widely used.
- Fixed sizes, difficult to resize.
- No redundancy.

Diagram:
- Disk
- Partitions: hda1, hda5, hda6
RAID

• Redundant Array of Inexpensive Disks
• Combine several disks
  – Increase total storage space
  – Provide redundancy
  – Improve performance
• Can be done in hardware or software
RAID-Linear

- Linear concatenation of several disks
- Increased total storage space
- No redundancy or performance improvement
RAID-0

- Striping
  - Data is interleaved across all disks.
- Increased total storage space
- Improved performance with parallel I/O
- No redundancy
RAID-1

- Mirroring
- Redundancy
  - Multiple copies of all data.
- No extra storage space
  - Device size is equal to a single disk.
- Improved read performance.
- Reduced write performance
RAID-4

- Striping with parity
- Redundancy, but less than with mirroring.
  - One "chunk" of parity bits per stripe.
- Increased storage space (minus size of one disk)
- Improved performance, but at cost of CPU overhead
RAID-5

- RAID-4 creates a bottleneck on the parity disk.
- Spread parity among all disks for better performance.
- Same total size as RAID-4.
Volume Groups

• A collection of devices (disks, partitions, RAID)
• The space of all devices is combined in the group, but not directly available as a device.
Volume Groups

- Combined space is divided into fixed-sized chunks
  - Physical Extents (PEs)
  - Similar to memory page frames
Volume Groups

• Create volumes from free-space in the group.
• Volumes consist of Logical Extents (LEs)
  – Each LE maps to a PE
Volume Groups

- Simple resizing of groups
  - Add new devices to the group to expand total available free-space.
  - Remove devices that aren't used by any volumes.
Volume Groups

- Simple resizing of volumes
  - Add or remove extents at the end of the volume.
Bad Block Relocation

- Detect I/O errors
- Remap bad blocks to a reserved area of the device.

![Diagram showing BBR Device, Normal Data Area, and Replacement Blocks]

BBR Device

Normal Data Area

Replacement Blocks
Snapshotting

• Frozen image of a volume.
  – Useful for performing consistent backups without needing to take filesystem off-line.

• Copy-On-Write to save old data.

• "Origin" volume is always up-to-date.

• Snapshot capacity can be smaller than origin.

• Multiple simultaneous snapshots of same origin.
Snapshotting

COW Table

Chunks

Origin

Chunks

Snapshot
Writing To The Origin

Write Request

Origin

Snapshot
Writing To The Origin

Write Request: Queued up to be finished later

Read chunk from origin

Origin

Snapshot
Writing To The Origin

Write Request: Queued up to be finished later

Write chunk to snapshot

Origin

Snapshot
Writing To The Origin

Write request:
Queued up to
be finished later

New COW table
entry: Map chunk 3
to chunk 1
Writing To The Origin

Release all I/Os waiting on this copy
Reading From The Snapshot

Reading from a unmapped chunk:
Get data from origin
Reading From The Snapshot

Reading from a re-mapped chunk:
Get data from snapshot
Multiple-Devices Driver (MD)

• In the Linux kernel, Software-RAID is handled by the MD driver.

• One core module to coordinate all activities.

• Each RAID level is handled by a different submodule ("personality").
  – RAID-4 and RAID-5 are handled by the same personality.

• Available in all 2.4 and 2.5 kernels
Device-Mapper Driver (DM)

• Kernel driver for performing various volume management tasks.
  – More modular and robust than MD

• Create devices that can arbitrarily map to other devices.

• New driver
  – Added in 2.5.45
  – Version available for recent 2.4 kernels
Device-Mapper Driver (DM)

- Every DM device is a concatenation of one or more "targets".
  - Similar to RAID-Linear
- Each "target" provides a specific type of mapping
- Several "target types" supported
  - Linear (similar to a disk partition)
  - Striped (similar to RAID-0)
  - Snapshot
  - BBR
Device-Mapper Driver (DM)

- Disk Partitions
  - DM device with single linear target
- Volume Groups
  - DM device with multiple linear or striped targets
Block Device Drivers

• MD and DM are considered "virtual" drivers
  - Produce logical block devices, instead of physical block devices.
  - Slightly different behavior than regular block drivers.
Block Device Drivers

- Regular, Physical Block Drivers
  - IDE, SCSI, others
  - Collect I/O requests on a queue
  - Sort requests for most efficient physical organization.
  - Drive requests across a bus to a physical device.
Block Device Drivers

• "Virtual" Block Drivers
  − Handle each I/O request individually.
    • No queues.
  − Decide how the request needs to be remapped.
    • Change target device number.
    • Change target sector location.
  − Return the I/O request to the block layer for submission to the new device.
Block Device Drivers

bio
dev: md0
sector: 532
data: ???

bio
dev: hdc
sector: 20
data: ???

md0

chunk size: 32k

hda  hdb  hdc
Enterprise Volume Management System

- Modular, extensible system for managing storage on Linux.
- Integrates all aspects of volume management into a single package.
  - Disk Partitioning (fdisk)
  - Volume Groups (LVM)
  - Software RAID (MD)
  - File Systems (mkfs, fsck, resizefs)
EVMS: General Architecture

User Interfaces
- GUI
- Text-Mode
- CLI

Plug-Ins
- DOS
- LVM
- RAID

User-Space
Kernel-Space
- RAID-0
- RAID-1
- MD
- DM
- Linear
- Snapshot
EVMS: General Architecture

• Engine
  – Core library
  – Coordinates all activities
  – Defines common set of possible tasks
    • Creation, deletion, resize, configuration changes
• User Interfaces
  – Communicate with Engine through well-defined API.
EVMS: General Architecture

• Plugins
  – Each plugin recognizes a specific volume format
    • Disks
    • Partitions
      – DOS, GPT, BSD, Mac, s/390
    • LVM Volume Groups and Volumes
    • Software RAID
    • BBR
    • Snapshot
    • Filesystems (FSIMs)
EVMS

• Operates in user-space
  - Volume discovery
    • "Disk" plugin discovers all disk devices.
    • Each plugin examines current list of devices.
      - Claims a device by removing from the list.
      - Creates new devices and adds to the list.
  - Creation, deletion, other administrative tasks
• Communicate with MD and DM kernel drivers to activate volumes.
Communication With Kernel

- Ioctls (I/O Control)
  - Commands passed from user-space to kernel-space
  - Driver-specific system-call
  - Directed at a particular driver
    - Specify a command
    - Specify additional arguments for each command
MD Ioctls

• Start-Array
  - Specify one child device.
  - MD will locate remaining child devices based on super-block information from first

• Stop-Array

• Set or Get Array-Info

• Hot-Add and Hot-Remove Disks

• Mark-Disk-Faulty
Device-Mapper Ioctls

- Create
- Remove
- Reload
  - Change mapping for a live device
- Suspend, Resume
- Rename
- Get-Info and Get-Status
Sample Device Activation

1) Discovers partition hda1 on disk hda, starting at sector 63

2) "Activate" hda1

3) "Create" new DM device
   - Name: hda1
   - One target: "linear hda 63"

4) Create a target for hda1

5) Map hda1 to hda starting at sector 63
EVMS Project Information

• Hosted on SourceForge
  - http://evms.sourceforge.net
  - Live CVS tree
  - Mailing lists: evms-devel@lists.sf.net
  - Bug tracking
  - Installation instructions
  - Documentation