Storage Management:
Pulling Together Management Across the Cloud and Bare Metal

Ric Wheeler
Senior Engineering Manager
File and Storage Team
Red Hat

The Good News

Linux File & Storage Systems

- Our storage stack is world class in many ways
  - LVM snapshots, RAID, remote replication, encryption
  - Used in embedded devices & enterprise servers
  - Frequently used to make storage appliances
- Supported across a wide variety of device types
  - SSD's, storage arrays, commodity disks
- Scales well in both performance and size
  - Multiple GB/sec of IO bandwidth for high end devices
  - Pushing the IOP rate for PCI-e devices
  - Supports 100's of TB of storage

Wide Variety of Choice

- Linux has a variety of choices at every level
  - Local file systems like ext*, XFS, btrfs
  - Networked file systems like NFS and CIFS
  - User space file systems (gluster, Ceph)
  - ISCSI, FC, FCOE, ... for block storage
  - DRDB (and NBD still!) for Linux specific block
- Support for high end array features
- All of this choice translates into complexity

These Strengths Make Us Very Popular

- Linux based storage has grown beyond our traditional expert level user base
  - It is the most common base for cloud storage today
  - Linux based Android is used in a huge number of phones and mobile devices
- New users and administrators are much less likely to be versed in our CLI based tools
  - Need to do a few basic things
  - Probably will never touch the “power tools”

The Bad News

Storage Devices Always Lie to Us

- Logical sector numbers do not represent physical layout
  - Hard drives remap bad sectors
  - Hardware RAID cards acknowledge writes that land in their cache but not on disk
  - SSD devices maintain massive mapping tables to translate virtual to physical layout
- RAID or device mapper like subsystems present an entirely virtual block device

Storage is Confusing

- Storage terms are unique to the domain
  - “target” (server) and “initiator” (client)
  - LUNs, partitions, zoning, ALUA, multipath, etc
  - Metadata, write barriers, crypto at the file system level
- Large sites tend to have experts in storage
  - Steeped in storage arcane knowledge
  - Someone else does networks, servers, etc
- Tiered implementations
  - Virtual block device backed by a file backed by....

Cloudy Complications

- Cloud storage has inverted the traditional popularity of a bare metal machine
  - S3 objects are very popular
  - Remote block device
  - Distant third is file support over NFS, SMB or other
- All of this is normally built as a layer on top of our traditional storage
  - HDFS, Ceph, Gluster, etc
- Someone still needs to manage the stack under the cloud

How We See Storage

Users Think of Storage as Use Cases

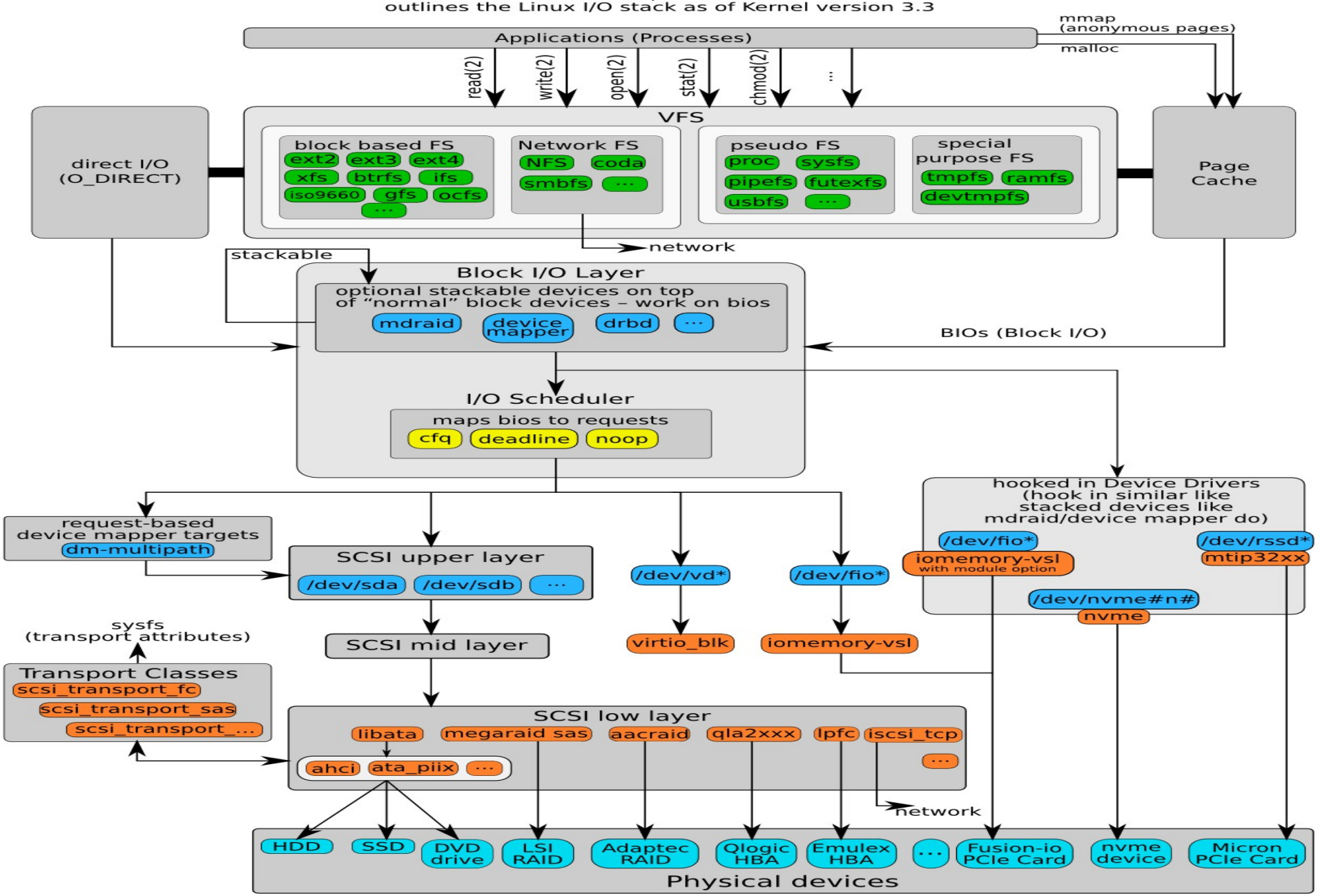
- Storage for my movies on my laptop?
  - At least 1TB?
  - Might want more space later?
  - Shared with my desktop and my TV?
- High performance backing for a database
  - Replicated and must be able to do 20,000 IOPs/sec
- Space in the cloud to backup my songs or photos?
- What do I do about that red light in my disk array?
  - Did I lose data? Do I need a new disk?

Kernel File and Storage System Developers

- Focus on specific areas of expertise in the kernel
  - IO scheduler?
  - Block layer?
  - SCSI
  - Specific file system expertise?
- Not a lot of cross component expertise
  - Even less focus on end to end “use cases”

The Linux I/O Stack Diagram

version 1.0, 2012-06-20
outlines the Linux I/O stack as of Kernel version 3.3



The Linux I/O Stack Diagram (version 1.0, 2012-06-20)
http://www.thomas-krenn.com/en/oss/linux-io-stack-diagram.html
Created by Werner Fischer and Georg Schönberger
License: CC-BY-SA 3.0, see http://creativecommons.org/licenses/by-sa/3.0/

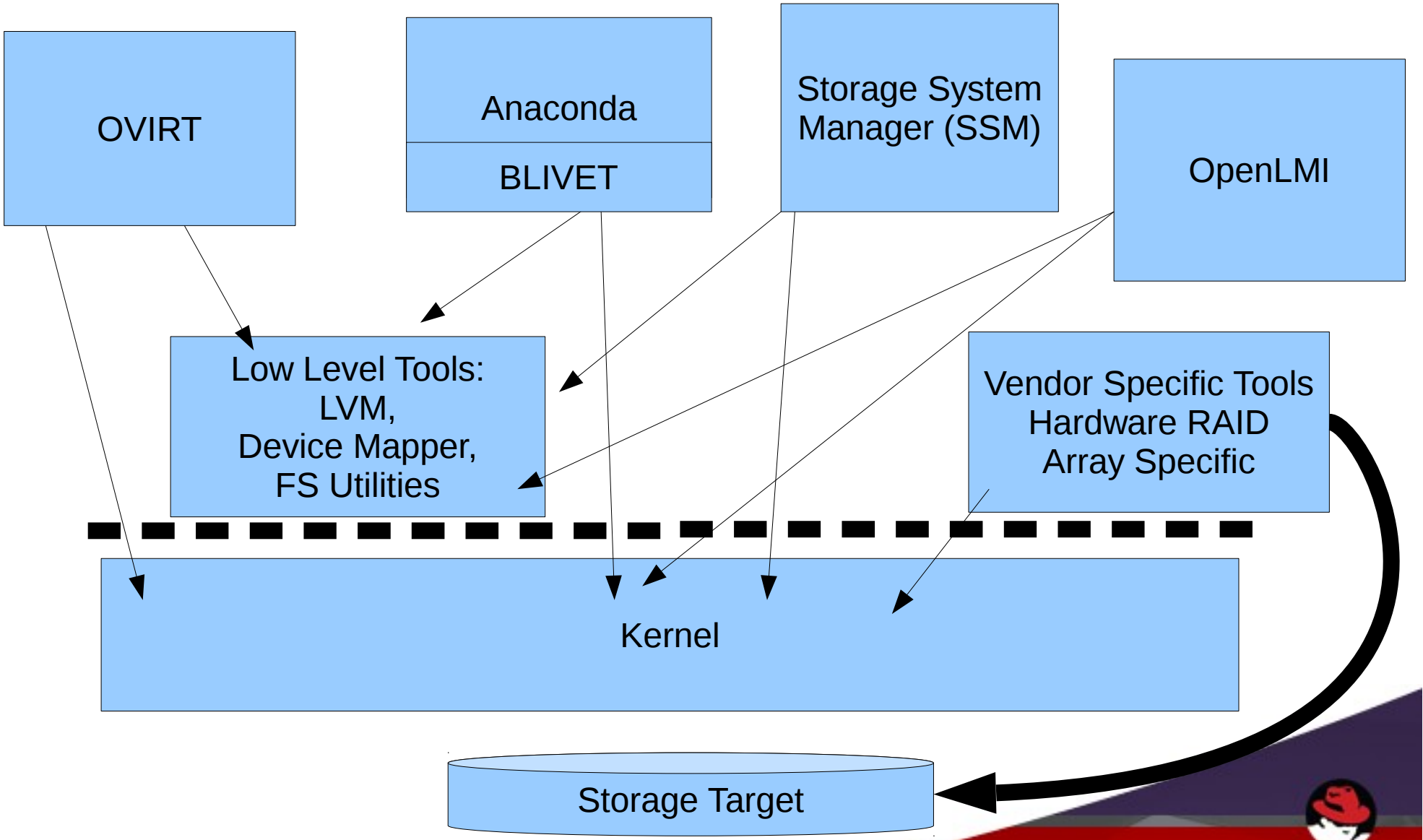
Open Source Storage Management

- Development community is fragmented by phase of life
  - Installer team writes installation code
  - Kernel people write low level CLI's
  - Run time management entirely different
  - Different tools for different distros
- Leads to multiple implementations
  - Each with its own use cases, names for concepts, etc
  - Gives us all a chance to recreate the same bug a bunch of times!

Management is Not the Most Popular Task

- Kernel Developers Don't Do GUI's
  - Unless emacs counts?
  - Most developers of storage and file systems manage them via CLI tools
- Enterprise scale management applications
  - Tend to be expensive and complex with steep learning curve for simple use cases
  - Produced by large, proprietary companies
  - Can support multiple vendors and types of gear (storage targets, SAN switches, etc)
  - Tend to be very full featured

Current Red Hat Storage Management Stack



Help is On The Way....

Management Focus in Recent Years

- Talks at LinuxCon Prague in 2011
  - Initial work on storage system manager
- Large gathering at Linux Plumbers in 2012
  - Brought together developers from across the space
- Enterprise distributions have staffed up projects

High Level Approach

- Identify key use cases
  - Support these in CLI, remote and GUI flavors
  - Fall back to power CLI tools for complicated operations
- Provide low level libraries in C & Python to replace need to always invoke CLI
- Provide infrastructure for asynchronous alerts
  - “add more space to your thin pool!”
  - Allows us to be reactive instead of having to monitor
- Pull developers together across the various efforts
  - Restructure code to maximize reuse

Ongoing Work on Storage Management

- Storage and file systems are difficult and complex to manage
  - Each file system and layer of the stack has its own tools
  - Different tools at installation time and during run time
  - No C or Python bindings
- Multiple projects have been ongoing
  - SUSE Snapper manages btrfs and LVM snapshots
  - libstoragemgt, liblvm,targetd libraries being developed
  - System Storage Manager
- http://lwn.net/Articles/548349

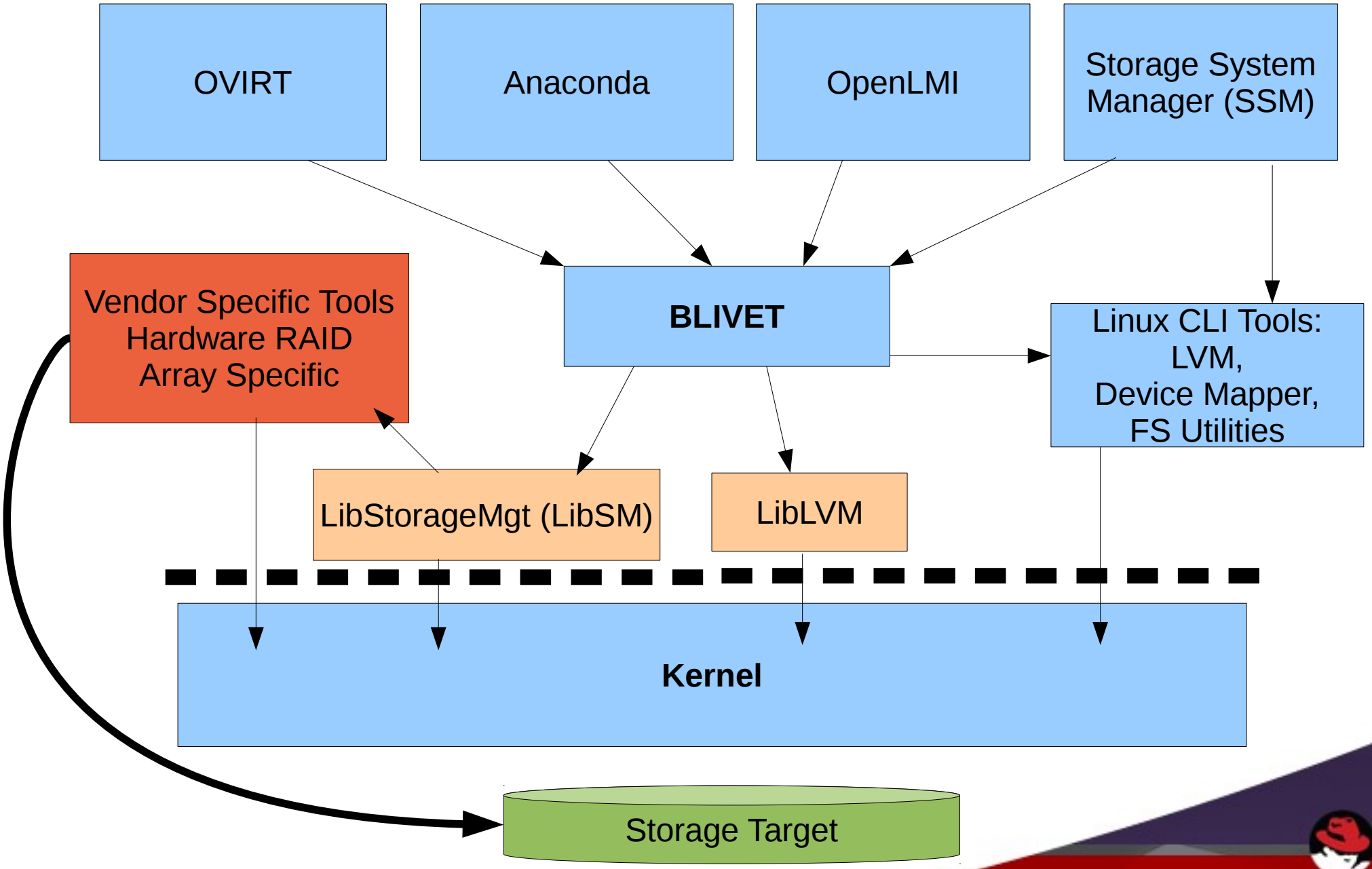
Low Level Storage Management Projects

- Blivet library provides a single implementation of common tasks
  - Higher level routines and installers will invoke blivet
  - https://git.fedorahosted.org/git/blivet.git
  - Active but needs documentation!
- libstoragemgt provides C & Python bindings to manage external storage like SAN or NAS
  - http://sourceforge.net/p/libstoragemgmt/wiki/Home
  - Plans to manage local HBA's and RAID cards
- Liblvm provides C & Python bindings for device mapper and lvm
  - Project picking up after a few idle years

High Level Storage Management Projects

- Storage system manager project
  - CLI for managing a variety of storage configurations
    - btrfs, ext4/xfs on LVM, encrypted block, etc
  - http://storagemanager.sourceforge.net
- openlmi allows remote storage management
  - http://www.openlmi.org
  - http://events.linuxfoundation.org/images/stories/slides/lfcs2013_gallagher.pdf
- Ovirt project focuses on virt systems & their storage
  - http://www.ovirt.org/Home
- Installers like yast or anaconda

Future Red Hat Stack Overview



New Features and Updates

Keeping Up with the Competition

- Vmware & Microsoft invest heavily in management
  - Vmware management is especially loved by users
  - Storage management is key here
- Standards bodies are implementing new array offload functions
  - Offload for copy to be used in migration
  - Standards driven by non-Linux companies
  - Specification based on vendor implementations
- Vendors drive de facto standards with key partners
  - Vmware has its own set of storage API's

Thinly Provisioned Storage & Alerts

- Thinly provisioned storage lies to users
  - Similar to DRAM versus virtual address space
  - Sys admin can give all users a virtual TB and only back it up with 100GB of real storage for each user
- Supported in arrays & by device mapper dm-thinp
- Trouble comes when physical storage nears its limit
  - Watermarks are set to trigger an alert
  - Debate is over where & how to log that
  - How much is done in kernel versus user space?
- User space policy agent was slightly more popular
- http://lwn.net/Articles/548295

Copy Offload System Calls

- Upstream kernel community has debated “copy offload” for several years
  - Popular use case is VM guest image copy
- Proposal is to extend the splice() system call
  - Offload copy to SCSI devices, NFS or copy enabled file systems (reflink in OCFS2 & btrfs)
- Updated patches posted by Zach Brown this week
  - http://lwn.net/Articles/566263/
- Older LWN.net article with background
  - http://lwn.net/Articles/548347

State of BTRFS

- Distribution support for btrfs
  - SLES supports btrfs in restricted configurations for system partitions
  - Red Hat has btrfs in tech preview in RHEL6
- Upstream is focused mostly on bug fixing
  - Waiting on the user space tools 1.0 release
  - 3.12 RC kernels should pull in lots of stability patches
- Fully production ready by the end of 2013?
  - https://lwn.net/Articles/548937

New NFS Features

- Labeled NFS allow fine grain security labels for selinux
  - Server and client code should be upstream by 3.11
  - Patches have been tested over several years
- Support for NFS V4.1 largely finished
  - Client side support optional pNFS mode, server does not
- Starting to work on implementation of NFSv4.2
  - Support for the copy offload is in V4.2
  - Hole punching, support for fallocate over NFS
  - V4.2 features mostly optional
- https://lwn.net/Articles/548936

Resources & Questions

- Resources
  - Linux Weekly News: http://lwn.net/
  - Mailing lists like linux-scsi, linux-ide, linux-fsdevel, etc
- SNIA groups for system management initiative
  - http://www.snia.org/forums/smi
- Storage & file system focused events
  - LSF/MM workshop
  - Linux Foundation & Linux Plumbers Events