Revolution in Virtualized Workload Management

http://veillard.com/Talks/LinuxConJapan2014.pdf

Daniel Veillard
veillard@redhat.com

redhat

Containers are back !

- History of containers
  - VServers, OpenVZ
  - Solaris zones, BSD jails
- Cheaper virtualization for cloud services
- Hardware independent
- LXC(s) containers
  - Improvements at the Linux kernel level

redhat

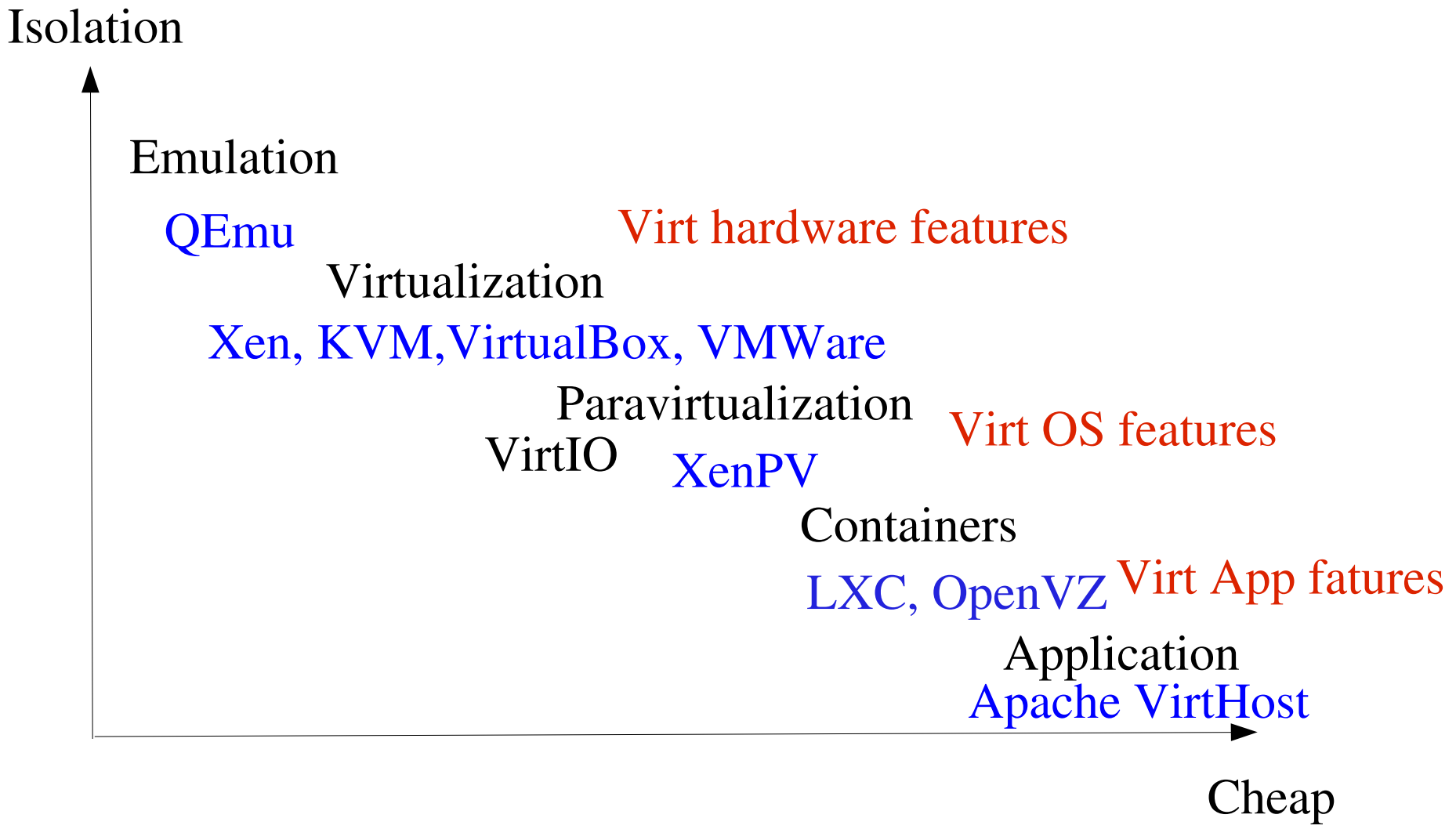
Containers technical points

One kernel to run them all !

- Partitioning and containment
  - Single OS image
  - Memory, disk and network
- Use 'recent' kernel features
  - Cgroups, namespaces, SELinux
- Very cheap:
  - No penalty at run time!
  - Near instant provisioning

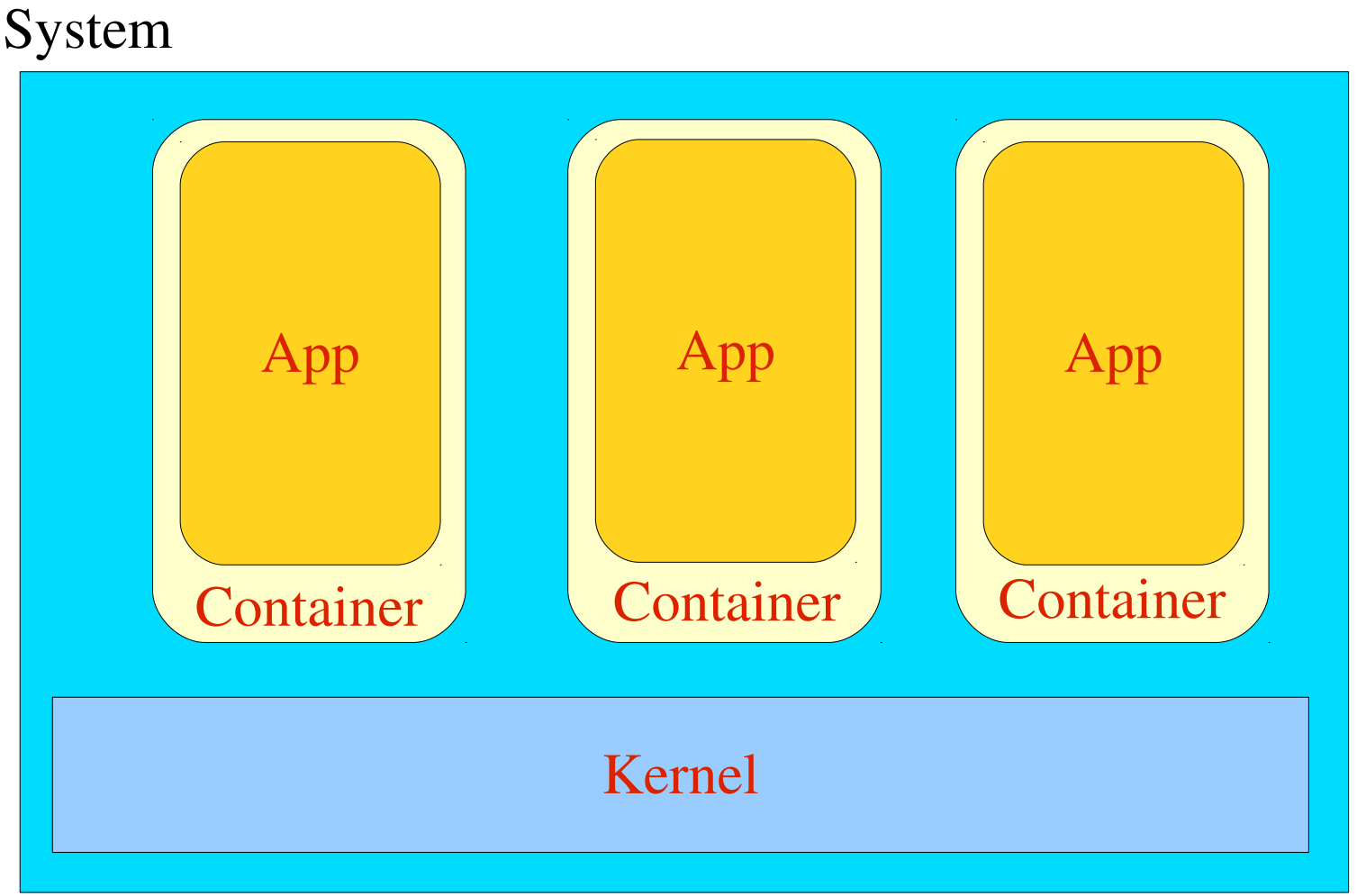
redhat

A spectrum of 'virtualizations'



redhat

OS, Container and Applications



redhat

Container systems

- Usually minimal base OS software
  - Minimize footprint and update rate
  - Minimize incompatibilities with user applications
  - No extra software means less risks
- Each container usually embeds one application
  - One service per container
  - All dependencies are dragged in the container
  - Containers are cloned to handle load
- The isolation is based on kernel properties
  - Kernel namespaces
  - Kernel cgroups
  - SELinux

redhat

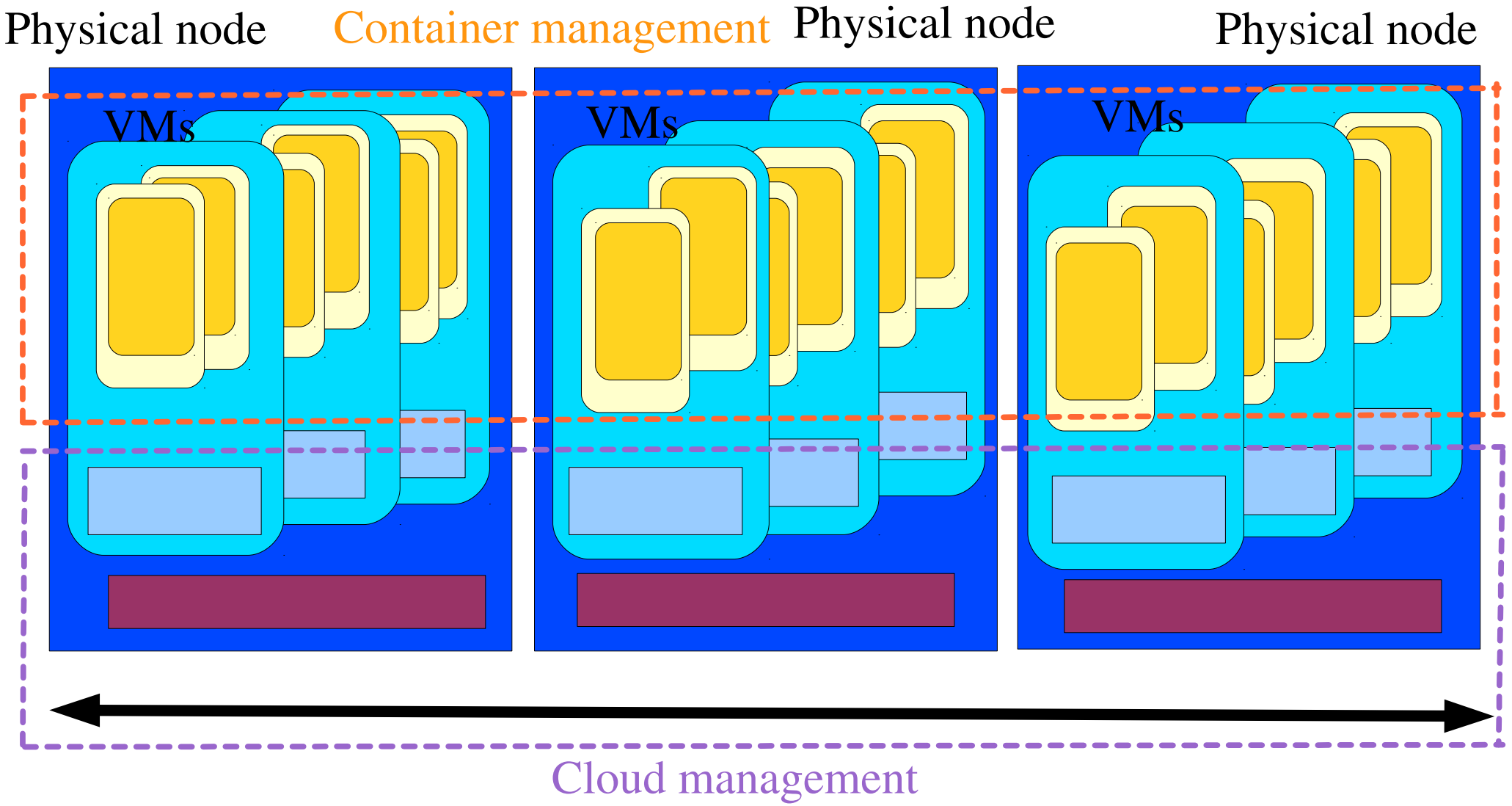
In a cloud context

- Conflicting requirements:
- For containers:
  - Container system kernels need to be recent
  - Container system base os should be minimal
- For cloud:
  - Cloud base OS need to be fairly stable
  - Cloud management code is rather complex
- As a result

  Containers are usually not run on bare metal

redhat

3 layered cloud



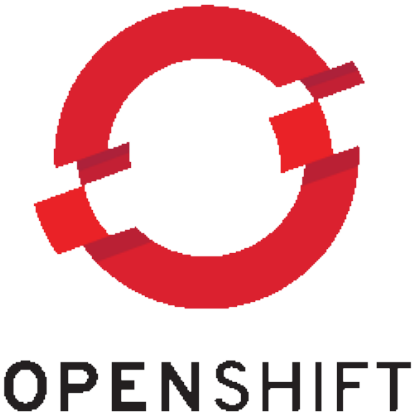
redhat

Managing Applications

- Packaging
  - Application centric
  - No dependency hell
  - Full control of versions
- Provisioning
  - Catalog, identity, signing
  - Container setup
  - Service registration
- Scaling
  - Load monitoring and balancing
- Plus reporting, accounting ...

redhat

Example: OpenShift



- Openshift PaaS
  - Runs on AWS, VMs or Cloud
  - Cartridge to package apps
  - Containers based on cgroups and SELinux
- Broker balancing the load and scaling the applications
  - Number of gears scale with load
- Change in application data are maintained though git
  - Pushes are then pulled to all running copies

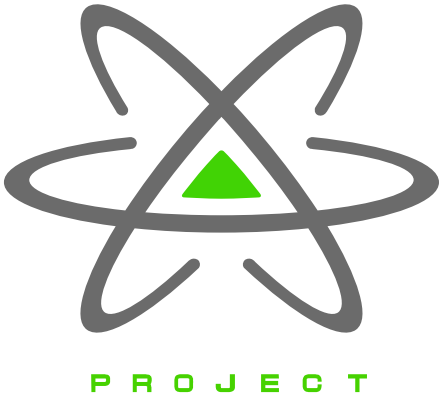
redhat

Example Docker

- Managing the application
- Integrated packaging
  - Self sufficient (embed dependencies)
  - Building tools
  - Runtime based on containers (LXC)
- Specifics:
  - A daemon runs on the node
  - Docker client to manage interactions
  - Registry of images
- Not yet at 1.0, Work in Progress but getting close !

redhat

Example Project Atomic



- Minimal systems:
  - Based on Fedora (RHEL-7, CentOS)
  - 120MB memory footprint
  - Systemd, Dbus, sshd and docker
- Uses SELinux to increase protections between containers
- New daemon geard to help manage applications
  - Developed from OpenShift
  - Manage app installation and replication
- New GUI Cockpit to handle the instances

redhat

How to update: rpm-ostree

- Keep existing distro mechanisms
- Smaller set of installed packages
- Parallel installation:
  - Initial tree
  - Updated tree
  - Shared files are hardlinks
- Install of updates
  - New subtree
  - 3-way merge

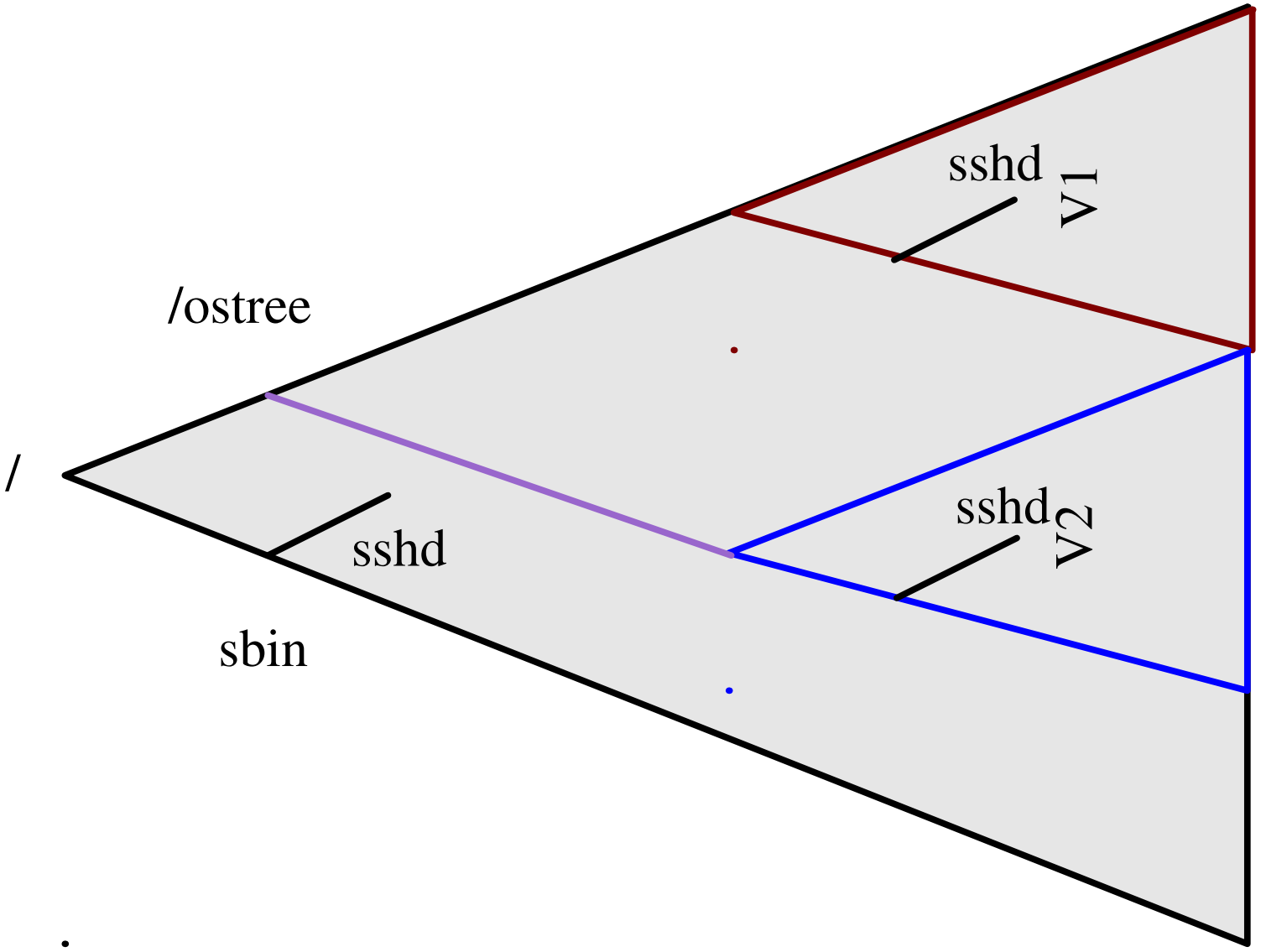
redhat

Rpm-ostree

- On server : build ostree updates
  - from an 'usual' rpm flow (repos)
  - A content description
  - Generate a flow of ostree versions
- On target systems
  - Rpm-ostree 'update'
  - Pull the changes, create a new tree
  - Merge in the changes, preserving modifications
    prepare for update on reboot
  - Rpm-ostree 'rollback'
- At any time the system is running a defined state !

redhat

Atomic updates



redhat

Work in progress

- Containers are only now deemed stable enough
- Docker established his format in less than a year
- APIs and integration of containers in the base OS
- Scaling and performance
- Automating the management
- Some applications Just Don't Scale
- Security standards and best practices
- Keeping the user data available and safe
- ...

redhat

Conclusion

- What is so revolutionary ?
  - 3 level of systems
  - Change in packaging
  - New OS update models

- Challenges ahead:
  - Stabilize
  - Integration and APIs
  - Smart management layer
  - Security