Managing KVM with CIM

Kaitlin Rupert
Linux Plumbers Conference 2009
Topics

• What is CIM
• CIM glossary
• What CIM provides for virtualization
• Managing KVM with libvirt-cim
• libvirt-cim versus libvirt
• Why CIM?
• Drawbacks of CIM
What is CIM?

- Stands for: Common Information Model
- An open standard defined by the DMTF
  - Distributed Management Task Force
- Describes how to control / exchange info about managed elements
- Profiles - model various operations and ways of representing concepts
- Uses a class hierarchy to represent objects and to show inheritance
CIM glossary

- **Class** – a collection of the definitions of state, behavior, and/or identity of a manageable items in a system
  - Contain:
    - Methods – functions that act on a class
    - Properties – represent attributes of a manageable item

- **Associations** – relationship between classes or instances of classes
  - Represents: dependency, identity, aggregation, composition
CIM glossary

- **Objects** – instantiation of a class, usually just called instances

- **Provider** – a library that represents a given class or classes
  - Implements an API for retrieving instances, invoking methods

- **CIMOM** - Common Information Model Object Manager
  - Server that facilitates communication between management application and providers
CMPI

- Common Manageability Programming Interface
- Technical standard developed by the Open Group
- Defines a C-based programming interface
- Prior to CMPI
  - Providers had to use CIMOM specific API
  - This tied provider sets to a specific CIMOM
What CIM provides for virtualization

- DMTF established for modeling virtualization:
  - Server Partitioning, Virtualization, and Clustering (SVPC) workgroup

- Workgroup developed profiles that describe:
  - Per guest:
    - Define / destroy / change power state / migrate
    - Add / remove / modify virtual resources
    - Representation of guest and resource configuration data
What CIM provides for virtualization

- Workgroup developed profiles that describe:
  - Host wide:
    - Create / delete / modify resource pools
    - Representation of pool configuration data
    - Generate events when a change occurs
Example of the SVPC schema
Managing KVM with libvirt-cim

- A provider set that manages KVM
  - Also Xen and Linux Containers
- Uses libvirt for hypervisor abstraction layer
  - Providers don't talk directly to hypervisor
  - Avoid platform specifics – most code paths are virtualization platform neutral
  - Map CIM objects, methods to libvirt abstractions / services where possible
Managing KVM with libvirt-cim
libvirt-cim versus libvirt

- **libvirt-cim:**
  - Parses XML, stores data in objects
  - User can listen for event objects using subscriptions
  - VNC sessions represented
- **Drawbacks:**
  - Features lag behind libvirt

- **libvirt**
  - Most info is returned in XML format
  - User must register a callback and poll a file descriptor to get events
  - Only VNC config info represented
Why CIM?

- Allows the management application to control different hypervisor types and even different host types with a single API
- Open standard – all provider sets should work in a known way
- Interoperability between vendors
- Existing open source providers, CIMOMs, and testing tools make for easy development
Drawbacks of CIM

- No mechanism for certifying an implementation conforms to the profiles
- Profiles don't cover all attributes needed
  - Can lead to too much specialization in providers
  - Reduces interoperability between provider and management app
- Profiles don't exist for all scenarios
  - Slow to be published, as they must go through a formal review process
  - Developed largely by volunteers