openQRM, pluggable virtualization for modern data-centers

Fosdem 2008
A presentation by Matt Rechenburg
Agenda

➢ OpenQRM in short
➢ Pluggable architecture
➢ Virtualization layer in openQRM
➢ Details about the openQRM Virtualization-plugins
➢ Developing a Virtualization-plugin
➢ Time for questions and discussion
Project History

➢ Derived from a proven commercial product
➢ Open-source since beginning of 2006
➢ openQRM Project on Sourceforge.net
➢ Active development by the community
openQRM: Goals and Concepts

➢ Separation of different modules in the data-center
  ➢ Servers -› physical hardware
  ➢ Services -› Operation System + Applications
  ➢ Storage- and Network-devices
➢ Abstraction of modules via Virtual-environments
➢ Plug-able architecture, huge selection of plugins
➢ Automated mechanisms for enhanced monitoring, system-management and rapid deployment
➢ Support for different operation systems and Virtualization types
Virtualization layer in openQRM

- Unifies the different Virtualization types
- Transparent support for migrating from physical resources to virtual partitions from different types
- Server-images does not require any changes
Virtualization Host-management

- Not only a GUI for a single Virtualization Host
- Automated Host deployment
- Automatic installation of the Virtualization components on the Host VE
- Cluster of shared Hosts (SSI)
- Load-balancing and scalability
Virtualization Partition-management

- Partitions created on behalf of Host-resource
- Partitions are just another type of resource
- openQRM maps partition commands to actions on the Virtualization Host
- Administration of vm's just like physical servers
- Partitions can move easily from one Virtualization Host to another
- Transparent resource management
The Xen plugin

- Automatic installation of the Xen VE via a resource boot-service
- Adding/removing/mapping of virtual network-interfaces
- Mapping of the virtual CPUs
- Increasing/decreasing memory consumption “on-the-fly”
- Pause/Unpause
- Handing over block-devices (FC/LVM)
- Live-migration
- Xen-console within the openQRM user-interface
- Supports NFS and Iscsi storage-servers
Fedora Core release 4 (Stentz)
Kernel 2.6.18-xen on an 1686

puppetclient login: 

Connected to 192.168.88.199 9002 online
The Qemu plugin

➢ Automatic installation and pre-configuration of Qemu on the Host VE via a resource boot-service
➢ Support for kqemu and KVM
➢ Adding/removing/mapping of virtual network-interfaces
➢ Increasing/decreasing memory consumption
➢ Supports NFS and Iscsi storage-servers
➢ Does not require special boot-image
The Linux-VServer plugin

- Automatic installation and pre-configuration of the Linux-Vserver tools on the Host VE
- Adding/removing/mapping of virtual network-interfaces
- Increasing/decreasing memory consumption
- Supports NFS storage-servers
- Best for web-farms
The VMware plugin

➢ Provided and maintained by Qlusters
➢ Manages existing VMware-server
➢ Support VMware GSX and ESX
➢ based on VMware API
➢ Supports NFS, Iscsi and local-deployment
A web-hosting setup
An advanced setup

- openQRM kommt auf 430 MB/s (seq. Write)
- openQRM kommt auf 570 MB/s (seq. READ)
- per iSCSI kriegt jeder Server 86 / 78 MB/s (R/W)
- Per LAN kommen ca 35 MB/s CIFS/NFS
2 classes to implement
... for example the Xen-plugin

Namespace
main/code/java/com/qlusters/qrm/plugins/partitioning/xen/

➢ XenPartitionBridge.java
  ➢ Maps the vm-commands
  ➢ Runs vm-commands on behalf of the Virtualization Host

➢ XenMacAddressProvider.java
  ➢ Generates Mac-Addresses for partitions
  ➢ Mac-address namespace per technology
public class XenPartitionBridge extends BasePartitioning {
    private static XenPartitionBridge instance = new XenPartitionBridge();

    public void startFromOff(ComputeResourceData resource) {
        ComputeResourceData node = Finder.getComputeResourcesFinder()
            .getReadOnlyHostingResourceByPartition(resource);

        StartPartitionCommand spc = new StartPartitionCommand(node, resource);
        CommandsExecutor.executeNow(spc);
    }
}

How the StartPartitionCommand works

```java
public class StartPartitionCommand extends XenCommand {
    private static final String startPartition = Prefs.getPrefs()
            .getString(
                StartPartitionCommand.class,
                "startPartition", xenControlScript + "start -m \${" + MAC + "}");

    protected StartPartitionCommand(ComputeResourceData node,
            ComputeResourceData partition) {
        super(node, partition);
        createCommand(partition, startPartition);
    }
}
```
public class XenMacAddressProvider implements MacAddressProvider {

  private long getAddress(int vmId, byte forthByte) {

    long result = 0x000000L;
    result += (forthByte & 0xff) << 16;
    result += vmId & 0xffff;
    result += MAC_TEMPLATE;
    try {
      result = getAddress(vmId, ++forthByte);
    } catch (IllegalArgumentException e) {
      System.out.println("We have reached max forth byte");
      Collections.sort(macs);
      Long lastMac = (Long) macs.get(macs.size() - 1);
      result = lastMac.longValue() + 1;
    }
    macs.add(new Long(result));
    return result;
  }

  ...
Summary and Conclusion

- Open architecture / fully plug-able
- Conforms different Virtualization technologies via partition-layer abstraction
- Transparent resource management
- Supports mainstream Virtualization vendors
- Plugin-development is made easy
Future Roadmap

➢ Focus on Virtualization
➢ Create plugins for
  ➢ OpenVZ
  ➢ Virtualbox
  ➢ Virtuozzo
➢ Enhance Virtualization plugins actions
➢ Port to PHP!

Your code and contribution is welcome!
openQRM on the internet

openQRM project

http://sourceforge.net/projects/openqrm

m.rechenburg@t-online.de
Time for your questions
Many thanks and have a great time at Fosdem 2008!