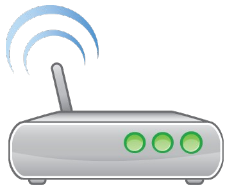
Ricardo M. Matinata – Software Engineer – Linux Technology Center - matinata@br.ibm.com
Aug/2011

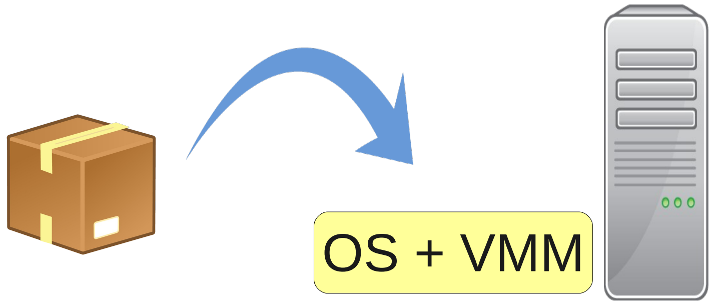
Implementing a Hardware Appliance Product:
Applied usage of qemu/KVM and libvirt

Basics

- Computer Appliance is usually a dedicated and separate piece of Hardware that:
  - Provide specific service(s) and/or make available a set of resources
  - Composed, usually, by a specific purpose hardware architecture
  - Customized OS and software stack
  - Tightly integrated hardware/software – perceived as an unity.
  - “Decouple and Share” Factor



- Virtual Appliance is:
  - Software and Services delivery model: single package/image.
  - Freedom of choice with regard to underlying hardware/hypervisor
  - Software stack and OS usually tightly integrated – no traditional installation required, increased control of configuration.
  - Reuse, Reuse, Reuse paradigm



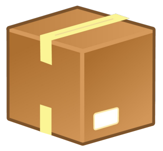
Revisited Hardware Appliance

- Combine the strengths of both Computer Appliances and Virtual Appliances
  - Develop your software/service as a virtual appliance
    - Single software stack testing stream
    - Single virtualization abstraction layer
    - Shorter time to market
  - Customer standpoint:
    - Flexibility

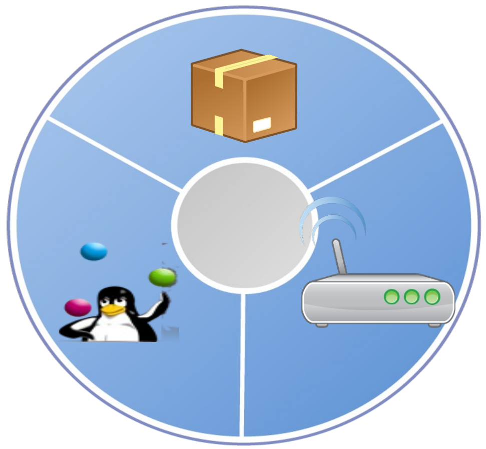
      Deployment

      Scalability

      Serviceability
    - Tailored costs



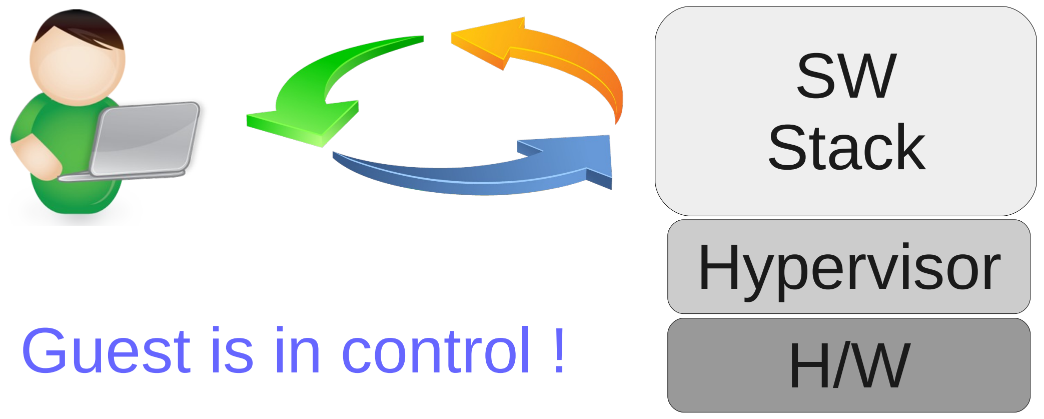
OR



Requirements

Basics

- As any other appliance, all the user sees is the software/service being provided:
  - – No awareness of virtualization
  - – Need for “owning” resources
- Compromise between decoupling and tight integration
  - – Maintain virtual appliance isolation
  - – Host need to react to guest operations
- New integration levels on:
  - – Out of the box experience
  - – Appliance life cycle



Guest is in control !

Pushing the Envelop on Requirements

Out of the Box

- Configuration
- Graphical console
- Hardware detection and information
- Connectivity

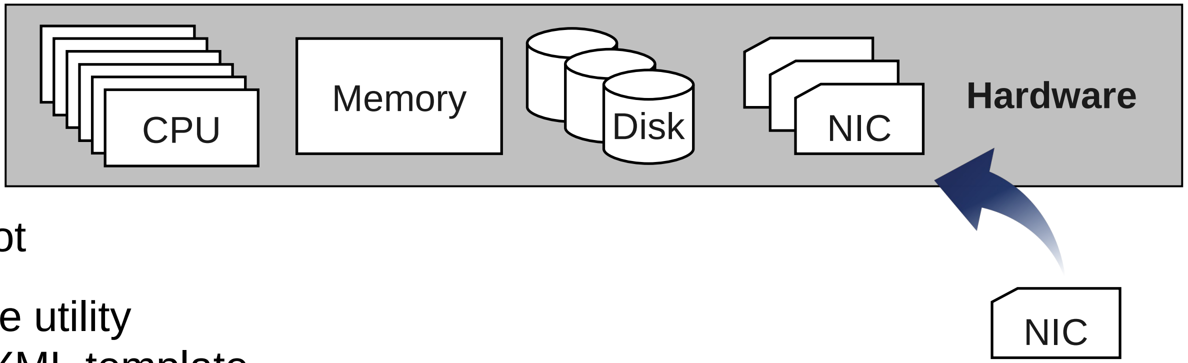
Life Cycle

- Updates
- Power cycle
- Backup/Restore
- Serviceability
- Hardware events

Implementation

Configuration

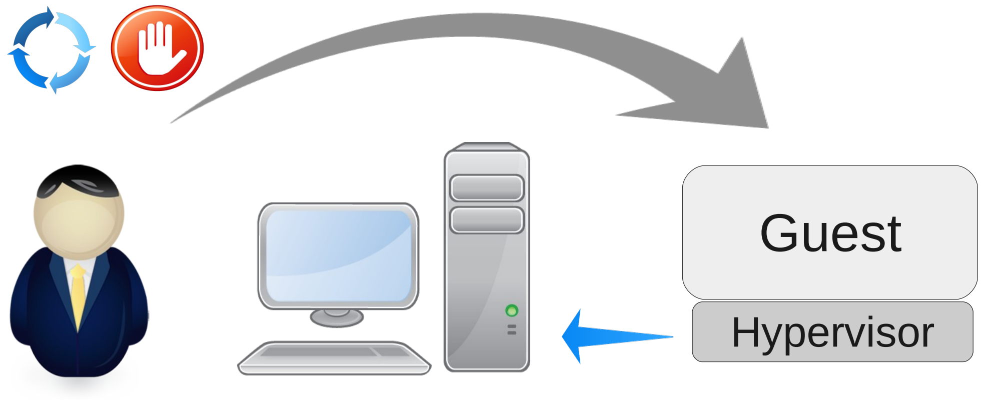
- Inspects host resources and automatically generates a domain.xml
  - – Memory
  - – CPUs
  - – NICs
  - – SMBIOS



- Renewed each reboot
- Solution: Find/replace utility
  - – Operates on an XML template
  - – Inspects H/W and apply pre-defined formulas

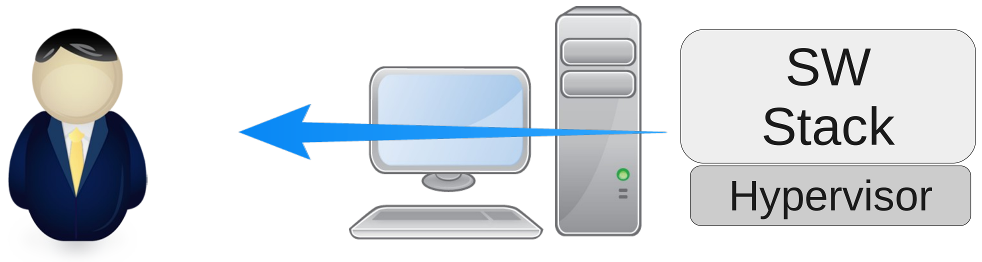
Power Cycling

- Since guest is in control, power cycling the guest needs to be reflect in the host
- Libvirt provides virConnectDomainEventRegister()
  - At the time, no distinction across shutdown/reboot
- Solution: Libvirt events watchdog
  - Register an Action Handler against libvirt
  - Guest “sets a bit” on the host to differentiate across shutdown and reboot



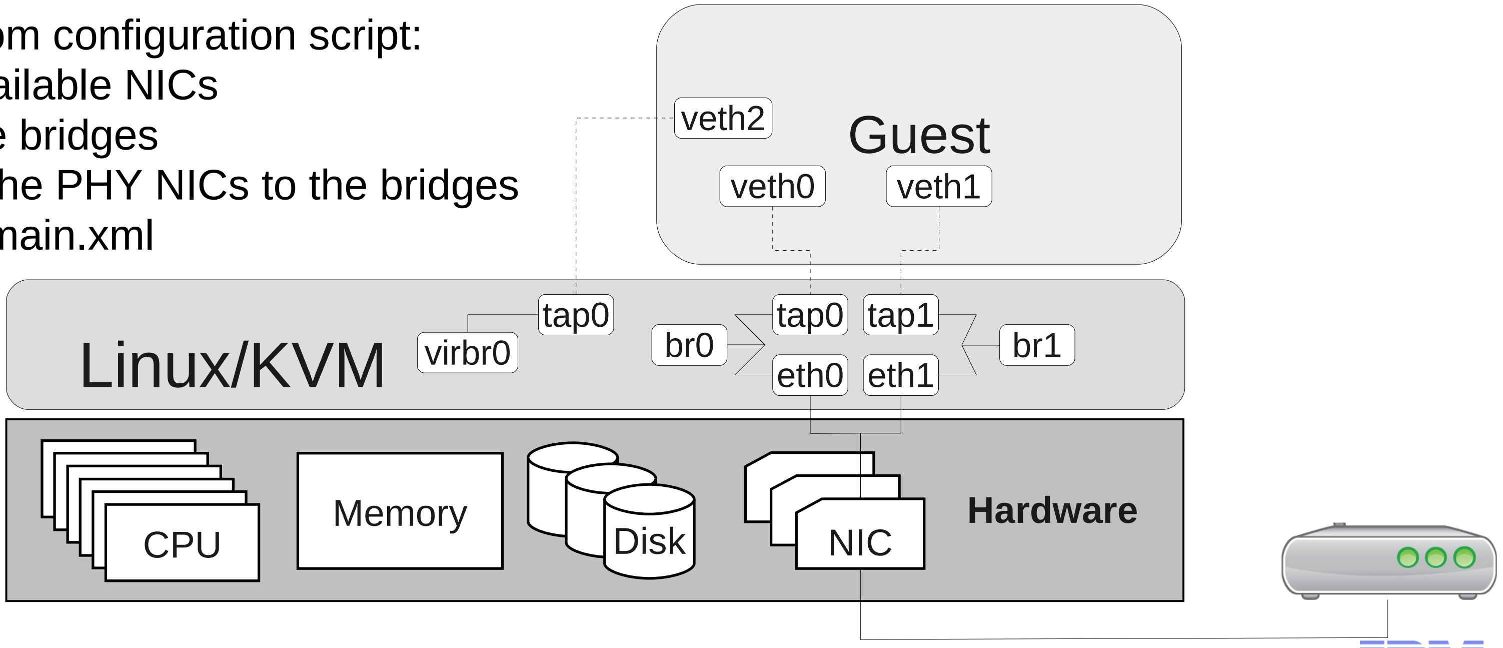
Graphical Console

- The appliance console needed to reflect guest's console
- SDL not working well
- Solution: Needed to reach for a frame-buffer based VNC client



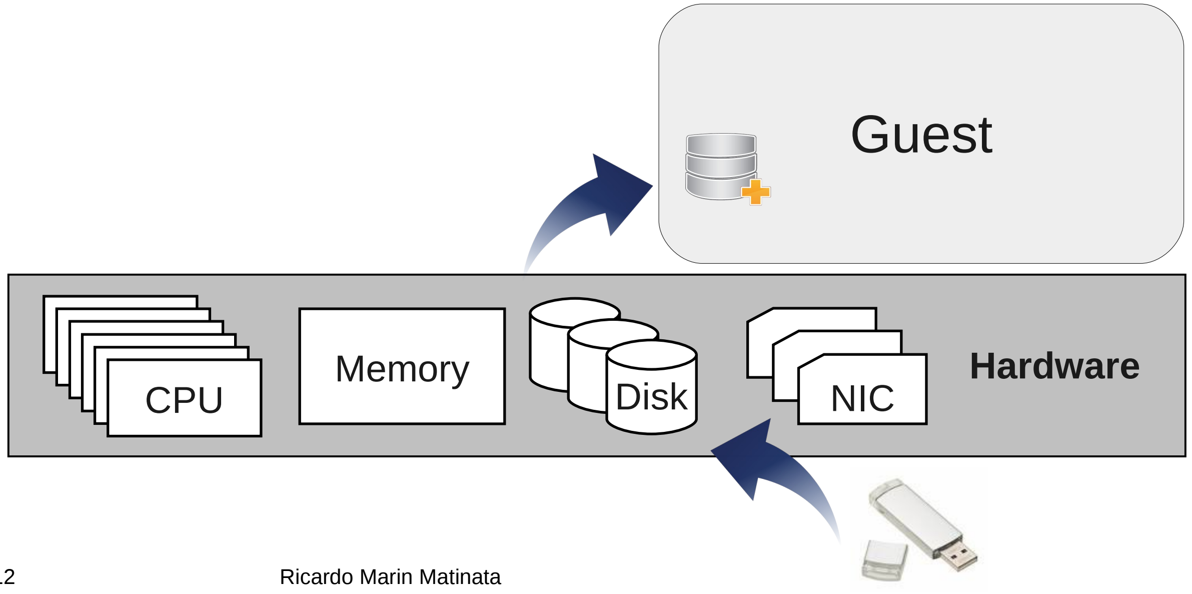
Networking

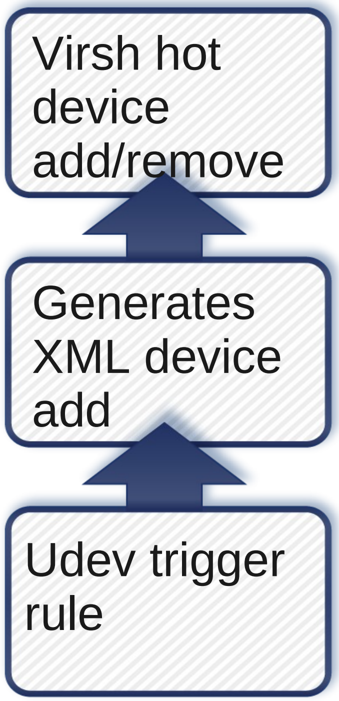
- Only the guest should have access to external network
  - All NICs added to bridges
  - All PHY NICs stripped down from IP configuration
  - All vNICs added accordingly
- Private network between host/guest
- React to H/W changes
- Solution: Custom configuration script:
  - Detects available NICs
  - Creates the bridges
  - Associate the PHY NICs to the bridges
  - Update domain.xml



USB Storage

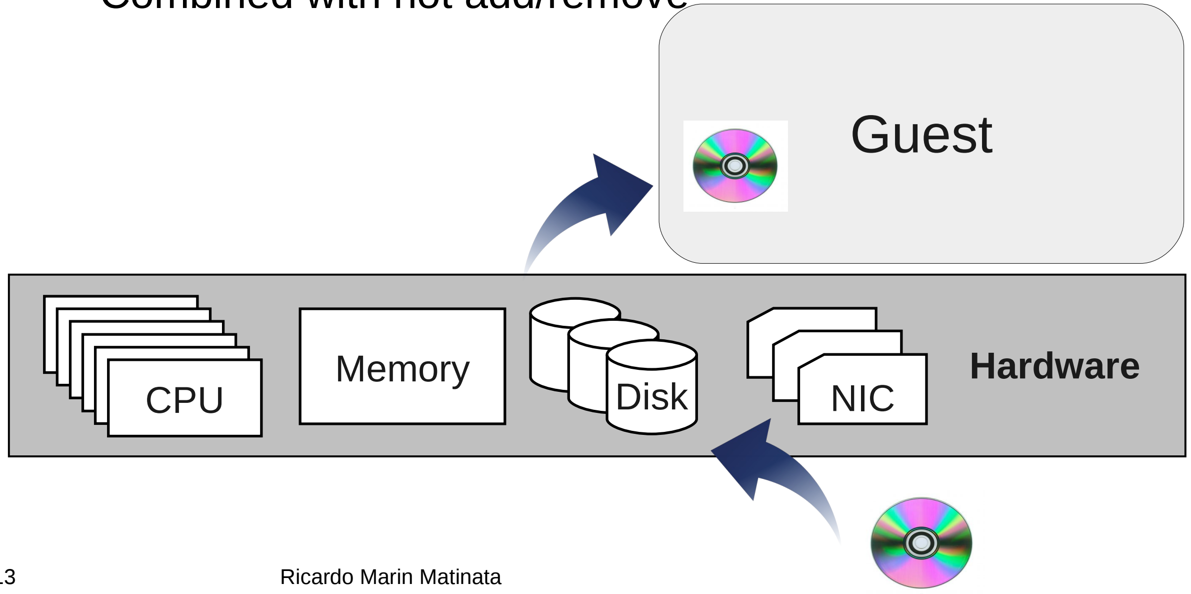
- Guest needs direct access to any plugged USB storage
- No reliable USB pass-through available
- Disk hot add/remove presented some challenges, including libvirt
- Solution: Passing through USB devices as virtio disks
  - Combination of udev trigger and hot disk add/remove
  - Requires creating an XML snippet and passing on to libvirt

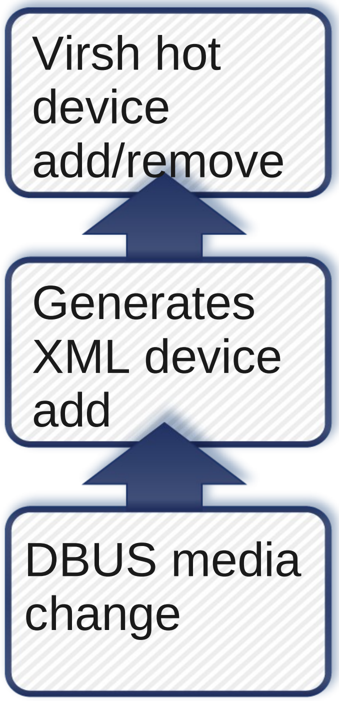




CD/DVD Access

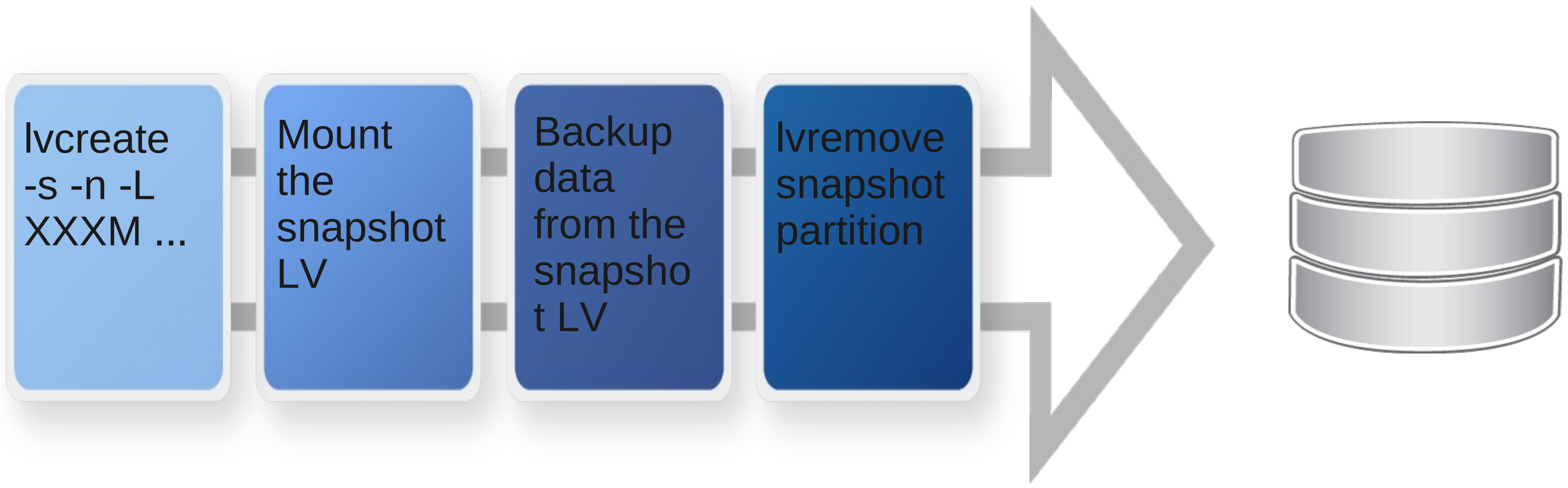
- Owned by the guest, R/W support
- No ATAPI emulation support
- Bug in qemu passing thorugh CDROM to the guest: Couldn't handle correctly media changes.
- Solution; DBUS monitor
  – Had to reach for DBUS events (insert/eject)
  – Combined with hot add/remove





Backup/Restore

- Periodically take guest's image snapshots
- No reliable solution available
- Solution: LVM Snapshots


lvcreate -s -n -L XXXM ...
Mount the snapshot LV
Backup data from the snapshot LV
lvremove snapshot partition

Other Details

- Updates
  - Coalesce the guest, update VM blob and bring back up
- Hardware Attributes
  - Entitlement and Service
  - Guest inherent host's
- Serviceability
  - SOS Reports
  - Guest dumps

Summary

- Having a full fledged OS like Linux, as host, facilitated the overall implementation
- At certain times, the existing interfaces were clearly too low level
  - Need to touch many different subsystems
  - Clearly, the implementation of policies could have make it easier to setup the environment
    - Networking
    - Memory
    - CPUs
- Some of the problems found seemed to be related to the scope of testing.
- Troubleshooting was too hard
  - Need better logging and debug levels

Moving forward...

- From a product development perspective, there needs to be more of a SDK mindset.
- Objective and to the point APIs
  - Node level, KVM specific actions
- Extensive documentation
- Extensive logging and troubleshooting