

Building VMNetX with qemu and libvirt

github.com/cmusatyalab/vmnetx

Benjamin Gilbert
bgilbert@cs.cmu.edu

June 3, 2013

What is VMNetX?

- Virtual Machine Network Execution
- Tool for executing remote VMs locally with low startup overhead
 - Disk chunks demand-paged from server
- Designed for preservation and execution of old software
 - Collaboration with University Libraries
- GPLv2
- Part of the Olive project, olivearchive.org

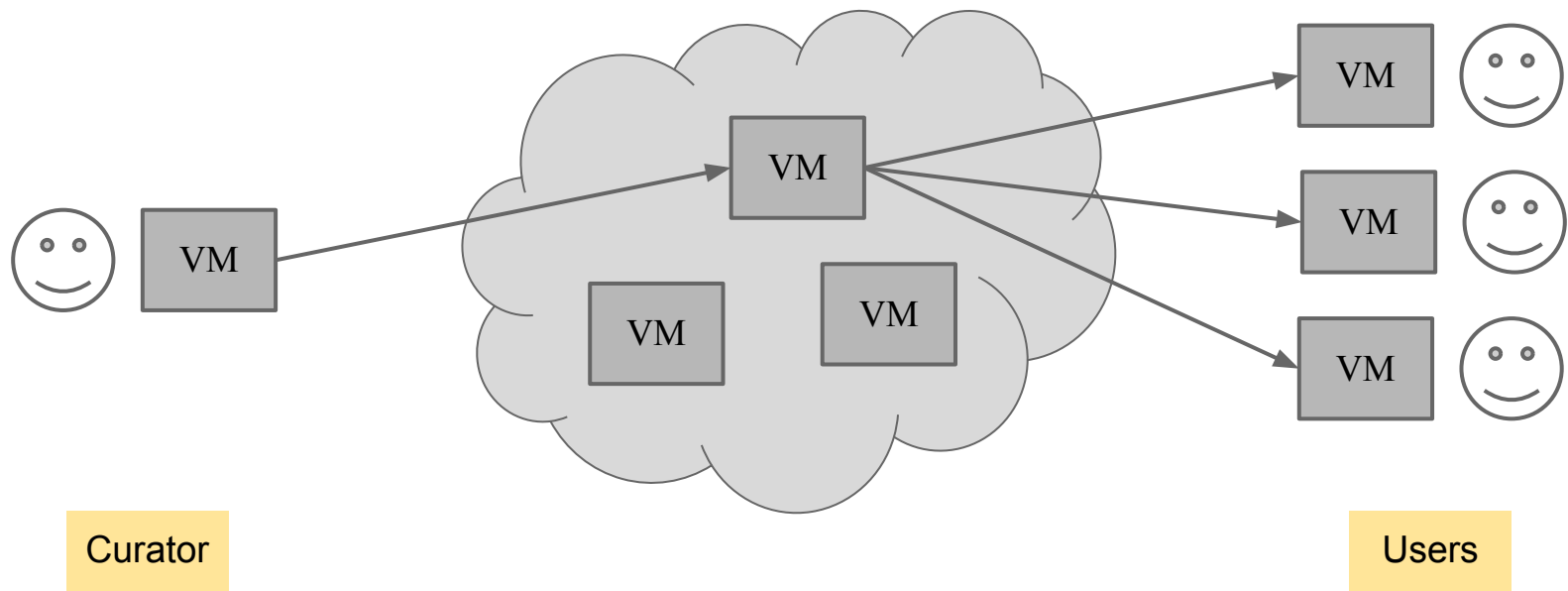
Demo

Building VMNetX

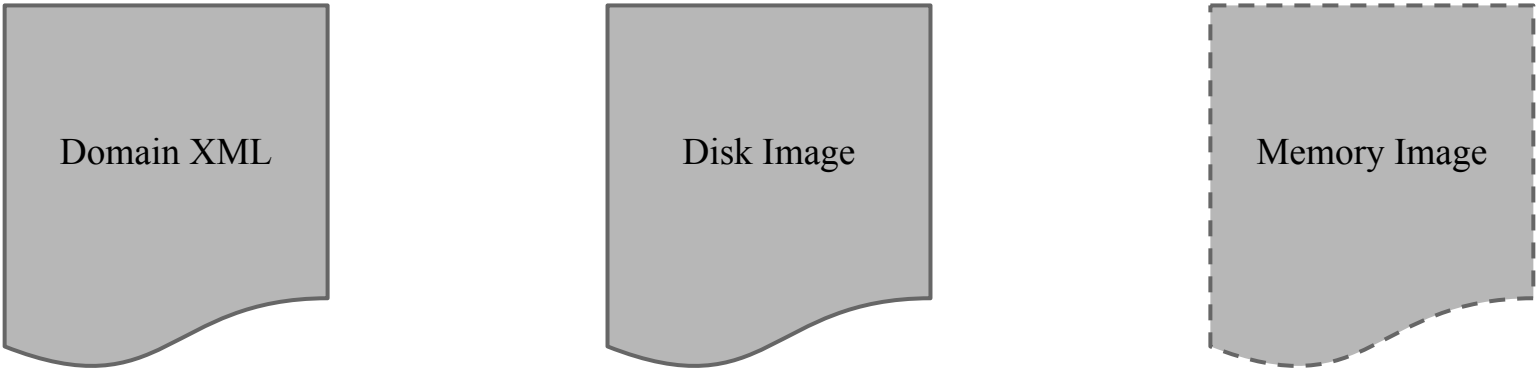
- System architecture
- libvirt & qemu in practice
- Unsolved problems

Virtual Machine Lifecycle

- Archive curator builds and uploads a VM
 - Using virt-manager for now
- Many users run it
 - We do not save their changes



What is a VM?



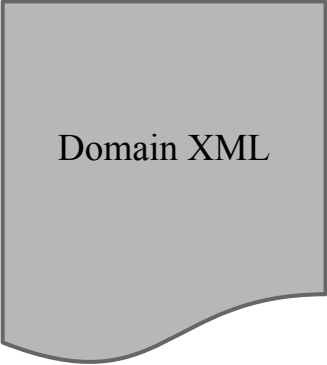
Domain XML

The diagram consists of three gray, rounded-bottom rectangular boxes arranged horizontally. The first box on the left is labeled 'Domain XML' and has a solid black border. The middle box is labeled 'Disk Image' and also has a solid black border. The third box on the right is labeled 'Memory Image' and has a dashed black border. All three boxes have a wavy bottom edge.

Disk Image

Memory Image

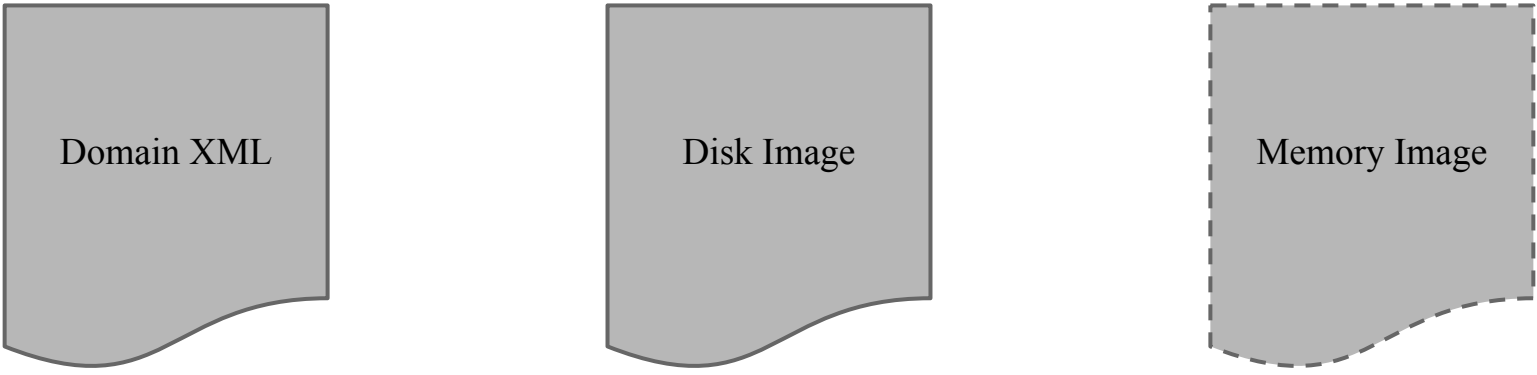
What is a VM?



Domain XML

```
<domain type="kvm">
  <name>machine</name>
  <uuid>a7434757-631b-496d-a1ba-638014c74cc4</uuid>
  <memory>65536</memory>
  <currentMemory>65536</currentMemory>
  <vcpu>1</vcpu>
  <os>
    <type arch="i686" machine="pc">hvm</type>
    <boot dev="hd"/>
  </os>
  <features>
    <pae/>
  </features>
  <clock offset="utc"/>
  <devices>
    <emulator>/usr/libexec/qemu-kvm</emulator>
    <disk type="file" device="disk">
      <driver name="qemu" type="raw"/>
      <source file="/disk.img"/>
      <target dev="hda" bus="ide"/>
      <address type="drive" controller="0" bus="0" unit="0"/>
    </disk>
    <controller type="ide" index="0">
      <address type="pci" domain="0x0000" bus="0x00" slot="0x01" function="0x1"/>
    </controller>
    <interface type="user">
      <mac address="52:54:00:03:a0:11"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x03" function="0x0"/>
    </interface>
    <input type="mouse" bus="ps2"/>
    <graphics type="vnc" autoport="yes"/>
    <video>
      <model type="vga" vram="9216" heads="1"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x02" function="0x0"/>
    </video>
  </devices>
</domain>
```

What is a VM?



Domain XML

Disk Image

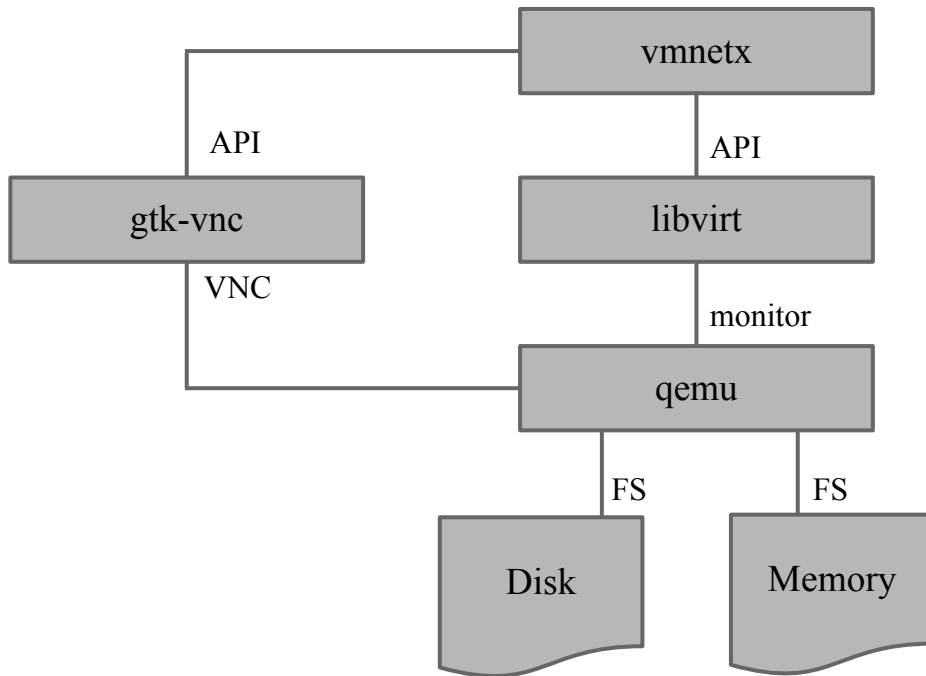
Memory Image

VM Package

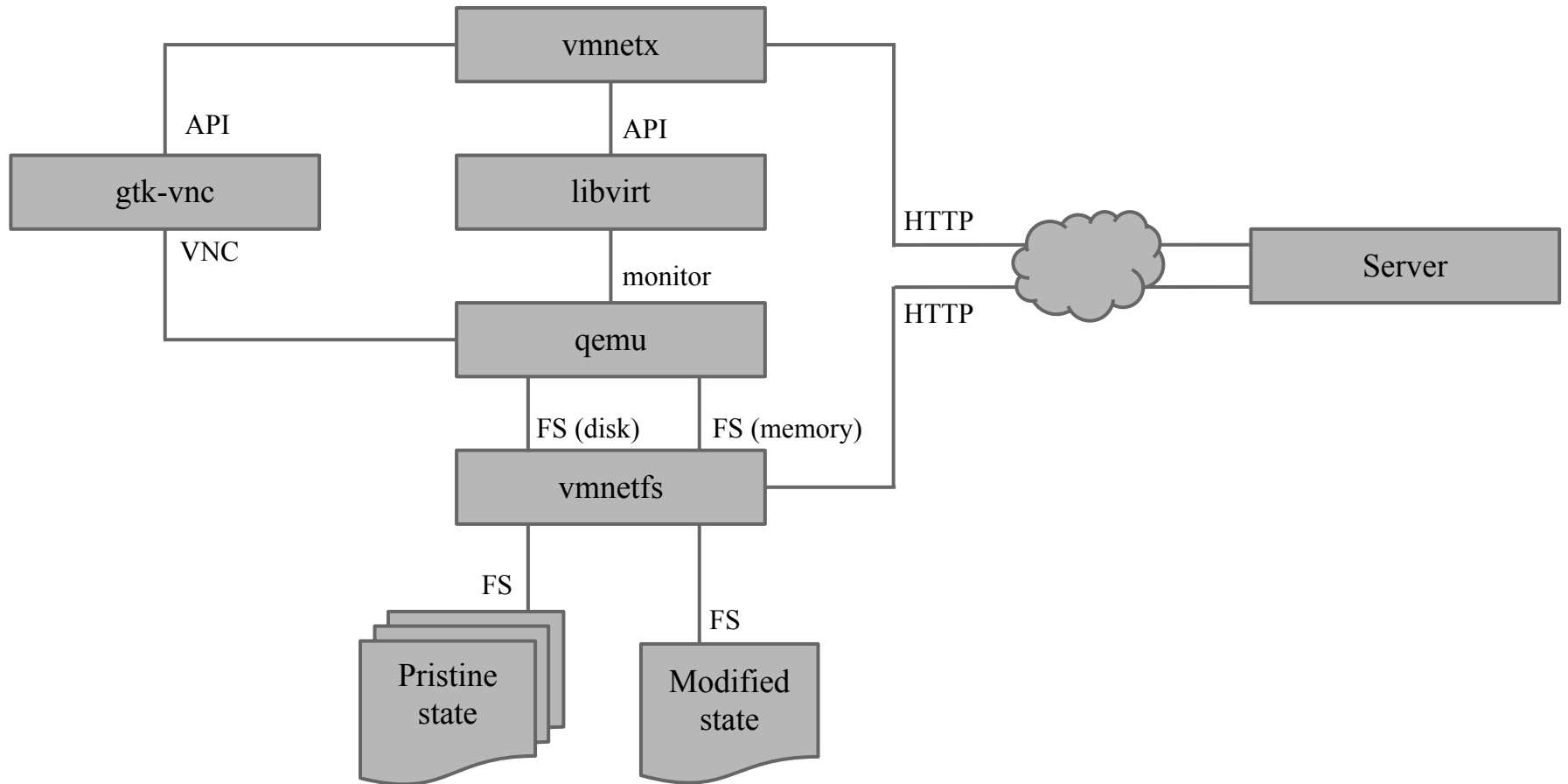
- Single ZIP file with uncompressed members
- HTTP Range requests to read ZIP header and members
- Throw it on a webserver and we'll launch it!



VMNetX Architecture



VMNetX Architecture



Just a piece of Linux software!

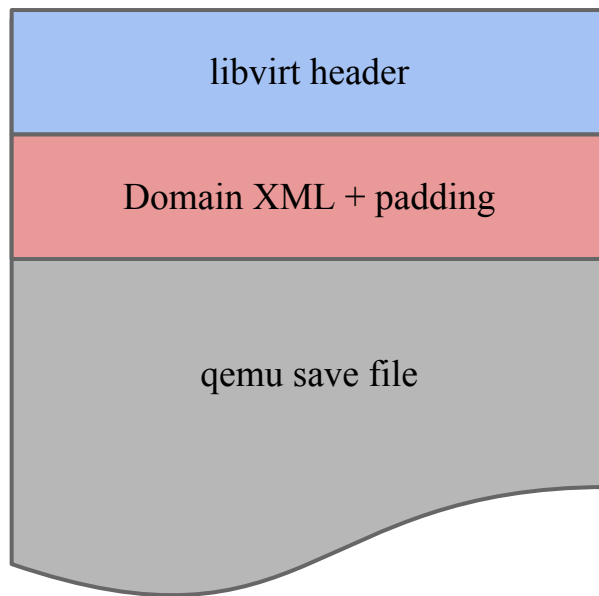
- Intended for installation by end users
- Should be a well-behaved Linux package
- We must use the software packaged by the user's Linux distribution
 - Unmodified KVM, qemu, libvirt, gtk-vnc
 - Software versions dictated by Linux distributions we want to support

Building VMNetX

- System architecture
- libvirt & qemu in practice
- Unsolved problems

Compression

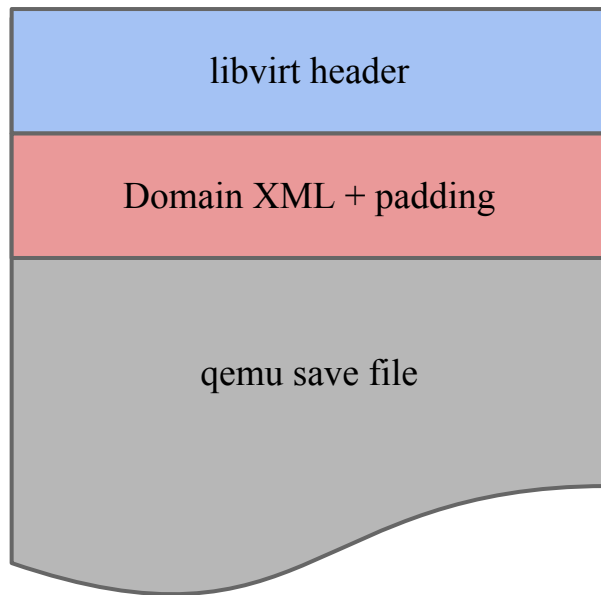
- Goal: reduce data transfer
- Disk: qcow2 compression (zlib)
- Memory: libvirt wrapper



```
struct _virQEMUSaveHeader {  
    char magic[sizeof(QEMU_SAVE_MAGIC)-1];  
    uint32_t version;  
    uint32_t xml_len;  
    uint32_t was_running;  
    uint32_t compressed;  
    uint32_t unused[15];  
};
```

Compression: Memory Image

- virt-manager doesn't compress memory images
- Recompress it ourselves!
 - It's just xz



```
struct _virQEMUSaveHeader {  
    char magic[sizeof(QEMU_SAVE_MAGIC)-1];  
    uint32_t version;  
    uint32_t xml_len;  
    uint32_t was_running;  
    uint32_t compressed;  
    uint32_t unused[15];  
};
```

Domain XML: Compatibility

- We need to support older versions of libvirt
- Validate new VMs against schema from older version
 - New syntax
 - Hardware support
- Manually fix invalid constructs
 - ...for now

Domain XML: Filtering

- We must allow VM to ship with custom domain XML
 - Different guest OSes may want different virtual hardware
- But, domain XML can configure VM to:
 - Attach to host hardware
 - Access host files
 - Bridge to host network interfaces
 - Bypass confinement
- Validate XML against restrictive schema

Domain XML: Rewriting at Runtime

- Rewrites:

- Machine name

```
<name>vmnetx-oregon-trail-15Z9ga</name>
```

- Must be unique

- Disk image path

```
<source file="/var/tmp/vmnetfs-8Xzt1a/disk/image"/>
```

- UUID

```
<uuid>a7434757-631b-496d-a1ba-638014c74cc4</uuid>
```

- VNC settings

```
<graphics type="vnc" socket="/tmp/vmnetx-p5ZGxc/sock"/>
```

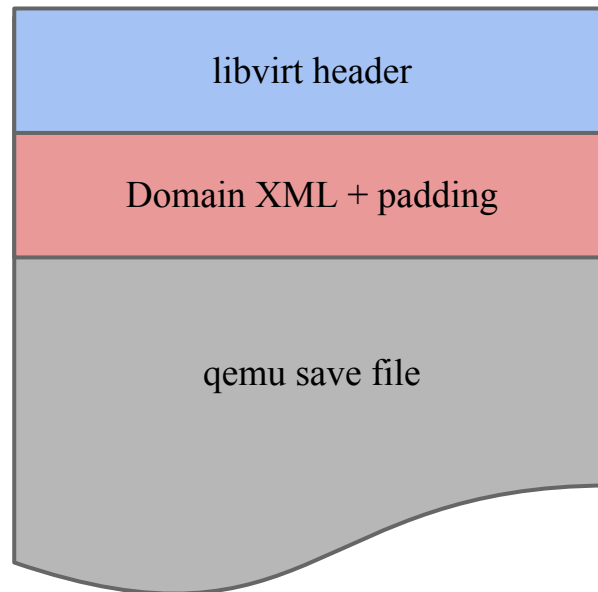
- Emulator path

```
<emulator>/usr/libexec/qemu-kvm</emulator>
```

- Varies from distro to distro
- Match type, architecture, virtual hardware version

Domain XML: Rewriting at Runtime

- Must also update memory image
 - ...directly: API won't allow unsafe overrides

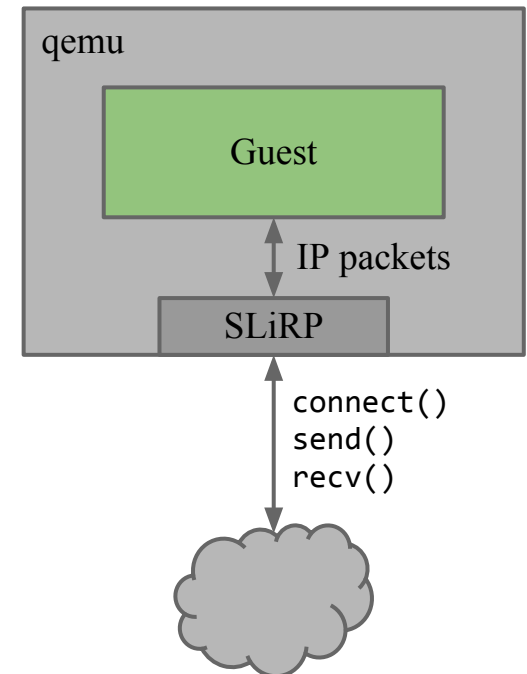


Virtual Hardware: CPU

- Ensure guests see the same CPUID on all machines
- We don't require configuring this, but we encourage it

Virtual Hardware: Networking

- We run qemu completely unprivileged
 - `qemu:///session`
- We don't want to require special network configuration in the host
- Solution: SLiRP

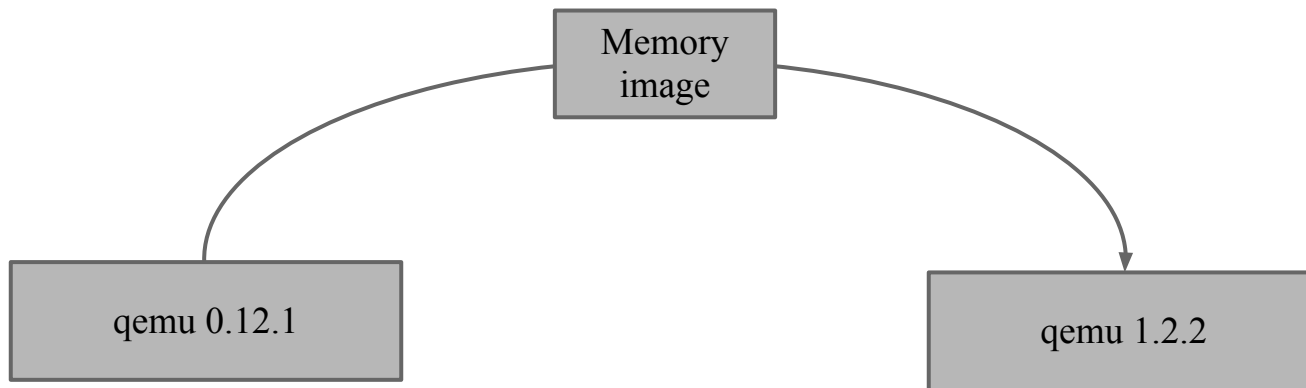


Building VMNetX

- System architecture
- libvirt & qemu in practice
- Unsolved problems

Memory Image Compatibility

- Multiple failure modes
 - Load memory image, then crash
 - Load memory image, then spin forever
- Detect and reboot



Guest OS Support

- KVM is not well-tested with obscure OSes
 - We want to support old Windows versions, DOS, BeOS, OS/2, OPENSTEP, ...
- We rely on upstream support

Guest OS Support: Windows 3.1

- Some host hardware delivers mouse events at 40 Hz, some at 80 Hz
- Windows 3.1 chokes if its mouse event queue overflows
 - goo.gl/3mjbh
- Limit Windows 3.1 guests to 40 Hz by intercepting Gtk `motion-notify-event`
 - Add custom metadata to domain XML

libvirt, FUSE, and SELinux

- libvirt runs qemu confined
- SELinux grants access based on file labels stored in filesystem
- FUSE doesn't support per-file labels
- Fedora provides SELinux boolean, but we can't ask the user to set it
- Solution: policy module reimplementing the boolean

Future Work

- Better VM creation tools
- Replace VNC with SPICE
 - Better graphics performance
 - Audio support
 - ...but not available everywhere
- Removable media
 - e.g., encyclopedias on CD
- Save/load local changes to a snapshot file
- Thin-client execution in OpenStack

Thanks!

github.com/cmusatyalab/vmnetx

Questions?

github.com/cmusatyalab/vmnetx