

Introduction to KVM: the Linux Kernel-based Virtual Machine





- Linux Kernel based Hypervisor
- Provides a backend for QEMU
- Managed through libvirt

Speaker notes

KVM provides the hypervisor, but without the other components it doesn't do anything. So we're really talking about a virtualization stack that features KVM.

Introduction to Virtualization and Containerization

Speaker notes

The first time I gave this talk, expecting that the audience all wanted to know how to do KVM, I discovered that a lot of the audience didn't understand Virtualization and Containerization or the differences very well.

What Is Virtualization?

Virtualization

Is the Emulation of an entire system entirely within another system.

Complete with its own FileSystem and Process Tree.

- Virtualization can run different Operating Systems than the Host, while Containerization can only run similar Operating Systems.
- KVM-QEMU-libvirt can run not only BSD, but Windows and just about any OS.
- A Container Image built on RedHat will run on Debian, but it can't jump to BSD.
- QEMU also supports alien hardware emulation.

Speaker notes

While QEMU loses the advantage of KVM in doing so, it can run Windows on ARM as a guest on your Ryzen gaming PC.

What is Containerization?

Containerization

is the isolation of a set of components and dependencies so that an application may be encapsulated.

The Container can run on another host without the need to install any additional dependencies.

**Key Differences Between
Containers
and
Virtual Machines**

- Virtualization is focused at the system level and emulates all aspects of a complete system.
- Containerization is focused on single applications or on atomic components of them.
- Containers can be used like a light weight Virtual Machine. This creates a lot of use case overlap and leads to confusion for those who don't understand the two technologies.

Container processes run on the host

```
$> ps -A | grep node
No results!
$> docker run -d --publish=8080:8080 0ba8043ea6028de76f55e4839
$> ps -A | grep node
    1659 ?          00:00:00 node # Now there is a Node process
$> docker exec -it 1cde4e07f659 ps -A
PID TTY          TIME CMD
    1 ?           00:00:00 node
   12 pts/0      00:00:00 ps
$> docker container stop 1cde4e07f659
$> ps -A | grep node
No results! Stopping the Container stopped node!
```

Speaker notes

To demonstrate I had no nodejs processes running on my docker host VM. I started a container running a nodejs process. and show that the node process is running as pid 1 in the container and is mapped to a different pid on the host. Once I stop the container the host no longer has a node process running.

VM processes are isolated

Running `ps` on your `kvm` host will get a result like this:

```
$> ps -A | grep qemu
2463 ?          17:07:48 qemu-system-x86
...
17379 ?         01:57:50 qemu-system-x86
$> ps -A | grep kvm
2467 ?          00:00:00 kvm-pit/2463
...
17386 ?         00:00:00 kvm-pit/17379
```

Speaker notes

For each VM `qemu` and `kvm` spawned a process, the `kvm` process names reference the `qemu` pid. Each VM has a full process tree, but none of those processes are visible on the host.

- LXC is a container system that works like a lightweight Virtual Machine.
- Docker can also be made to function as a lightweight Virtual Machine.
- Kubernetes is an “Orchestrator” — a tool for managing Containers in large environments. It relies on Docker but appears to be developing the ability to operate without it.

Speaker notes

A few years ago I used LXC in a situation where the light weight virtual machine scenario was appropriate and chose it for that reason. Docker offers a lot more and you should not consider LXC as a viable alternative, since it has no advantage, only a lot less features and a much smaller community. In retrospect LXC was the wrong decision.

Micro Services: the Philosophy of Containerization

- Every component should be isolated to the finest detail and containerized at that level.
- Containers can be grouped together to provide a complete application.
- An example WordPress Deployment would have a MariaDB Container, a PHP-FPM Container, and a Web Server container.

- If you build a single WordPress container with Apache-ModPHP and MariaDB it may be convenient but isn't benefitting from being a container.
- The container can only be shipped to Linux Hosts on x86_64 architectures using the same Containerization.
- Virtual Machine Disk Images can be converted to run on different Virtualization Platforms.
- The VMLite Container will use less resources than a full VM.

Speaker notes

Although it is possible to convert an LXC image to Docker, you will never get it to a BSD Jail.

Since utilities exist to convert VM disks between formats, you could copy your wordpress VM from KVM on Linux to BHyve on BSD, and also export it to your ui-designer using VMware Fusion on a MAC.

Which is Better?

Containers and Virtualization serve different needs.

- Containers solve deployment issues and permit elastic scaling.
- Virtual Machines are fully isolated from their host.
- Virtual Machine images are larger and require more resources to run.
- Virtual Machines can provide a full desktop environment.
- Virtual Machines can run different OSes than the host and even emulate different architectures.

Speaker notes

It is a common practice to create a containerized environment in a virtualized environment.

Virtualization for a Lab

For your various projects you need to:

- Try a different Linux distribution than the one you're running.
- Test that your code will run on BSD as well as Linux.
- Have multiple hosts because you're learning how to build a Kubernetes Cluster.

A virtualized LAB is what you need.

Speaker notes

And maybe you have an ancient game you still enjoy playing, but it needs Windows 98.

The Hypervisor

- The fundamental functions of a Virtualization platform are carried out by the Hypervisor.
- The Hypervisor creates the guest instances.
- The Hypervisor allocates host machine resources to the guests.

KVM vs Other Virtualization Platforms.

VMWare

- VMWare has the best tooling, feature set, and ecosystem.
- VMWare is expensive
- KVM is free

Speaker notes

The tooling and support help keep VMWare dominant in the Enterprise Market. Very Large Enterprises are willing to foot more internal development and support through OS Vendors such as RedHat and Canonical when the return is elimination of a substantial VMWare bill.

For your lab or a small organization needing an inexpensive virtualization platform KVM can provide all of the needed function.

VirtualBox

- KVM outperforms VirtualBox
- Widely regarded as the least stable virtualization platform.
- VirtualBox works on Linux, Windows and Mac.
- A lot of development tools like Vagrant and Minikube default to VirtualBox.

Speaker notes

While both of these examples do support KVM, there is more effort involved to use it. In the case of Minikube an extra command line switch. In the case of Vagrant, a much smaller stock of available images. Often times if you're following a tutorial the prepared images will be VirtualBox only.

Others

- BSD has BHyve.
- Xen. Early VPS providers were using Xen. Nearly all of the former Xen providers have switched to KVM.
- Hyper V requires running a Microsoft OS. Generally cheaper than VMWare. Failure of Host OS (Windows Server) will also take out VMs. Patching host OS will also cause downtime or require moving VMs.

The Typical KVM Stack

- Bridged Networking
- QEMU Emulator
- Libvirt (API)
- KVM Hypervisor
- Virsh (Command Line Tool)
- Virtual Machine Manager (GUI Tool)

Speaker notes

And finally we get to talk about KVM-QEMU and Libvirt.



Full System Emulation in Software.

- QEMU can emulate a different architecture in software.

Virtualization

- Run virtual machines at near native performance using either KVM or Xen hypervisor.

Speaker notes

QEMU also has a user mode emulation that allows running supported OSes without a full image. Unlike Containers this will permit running BSD on a Linux Host. This functionality is not relevant to this presentation.

First Released in 2003.



Toolkit to manage Virtualization platforms.

API accessible from C, Python, Perl, Java and more.

Supports KVM, QEMU, Xen, VMWare ESX, LXC, BHyve and more.

Runs on Linux, FreeBSD, Windows and OS-X



- Near Bare Metal Performance for most tasks.
- Benchmarks typically show about a 5% performance loss on AMD and Intel CPUs.
- Some benchmarks show a greater performance hit.



- Although KVM is available on ARM the performance analysis I looked at showed a significant performance hit.
- KVM is considered both stable and performant by those who use it. As proof: Amazon Web Services, Google Compute and most of the independent VPS hosting companies use it.
- Video Rendering Performance is poor. Depending on workload it may not be suitable for Virtual Desktop Infrastructure and any computational workload depending on GPU acceleration.

Speaker notes

Linode switched their entire operation from Xen. Companies using KVM for VPS offerings include: Digital Ocean, Vultr, DreamHost, BlueHost and HostGator use KVM. Rackspace bought SliceHost who were a Xen based host, but are now offering VMWare VPS.

The workaround for Video Performance is to install graphics cards. For most use cases it probably makes more sense to either dedicate boxes for Windows or pay for VMWare licensing.



**Bui
Idi
ng**

a Home Lab



You
may
use a
multi-

core desktop or dedicate a box that you will manage from your workstation. My current home server is on Debian Stretch and my workstation is LinuxMint (Bionic Base). For going through the steps I installed Cosmic on a spare box.



This
part

of the presentation will walk you through getting a virtualization environment running on a recent (early 2019) Debian family distribution.

Most commands will need to be run as root, however, I will not place sudo before each command.

Before You Begin

Check your BIOS and make sure that your processor has the Virtualization Extensions enabled.

If you're on Ubuntu install 'cpu-checker' then run '(sudo) kvm-ok'. If it fails or if you're on another platform you'll need to get into your BIOS. Typically the setting is AMD-V, Intel-VT, or Intel Virtualization, make sure it is enabled. You may have to poke through a lot of screens and menus to find it.

Bridged Networking

In most cases we'll want our virtual machines to be visible to other hosts, and we'll want to share the host's ethernet adapter. If your hosts don't need to be seen you can use NAT which should work without any extra steps.

To switch to Bridged Networking you'll need to configure networking by hand.

Bridged Networking

A bridge is a way to connect two Ethernet segments together in a protocol independent way. Packets are forwarded based on Ethernet address, rather than IP address (like a router). Since forwarding is done at Layer 2, all protocols can go transparently through a bridge. (<https://wiki.linuxfoundation.org/networking/bridge>)

Speaker notes

For Desktops Ubuntu has defaulted to Network Manager, for servers recent Ubuntu has replaced the traditional `/etc/network/interfaces` with NetPlan. To configure Bridging we'll need to switch to the traditional interfaces file.

Furthermore Ubuntu and other distributions are using `systemd-resolved` instead of traditional `resolv.conf`. You'll need to either switch back or configure name resolution the new way.

Name Resolution

The New Way

If your system uses systemd-resolved:

Edit */etc/systemd/resolved.conf*

```
[Resolve]
DNS=192.168.1.5
# set cloudflare and google as backups to local
FallbackDNS=1.1.1.1 8.8.8.8
Domains=private my.domain
```

The Old way

On Ubuntu install the package *resolvconf*

Delete the symlink stub */etc/resolv.conf* and create a new */etc/resolv.conf*.

Then run `systemctl disable systemd-resolved`

Then run `systemctl mask systemd-resolved`

Speaker notes

With a static IP configuration, your system won't be getting DNS information from DHCP anymore.

If you configure nameserver information in your interfaces file, it will be picked up

I did not provide a legacy resolv.conf example

Switch to Interfaces and Install Bridging

- Make notes on the ip configuration you want.
- The command `ip address` will show the current settings. More importantly it will tell you what the system calls your interface, because of the (un)predictable names feature

```
apt install ifupdown bridge-utils  
systemctl disable NetworkManager  
systemctl mask NetworkManager
```

Speaker notes

So called predictable names uses the computers interpretation of the slot a card is in to assign the device name consistently across boots. MAC address makes more sense but many NICs support changing their MAC, however, rearranging the cards in a chassis would also break predictable names.

The mask command prevents services that want NetworkManager from being able to start it. This isn't critical because Network Manager respects the configuration of interfaces over its own, but the gui may show other settings. With NM disabled the gui will show the network as not running.

`/etc/network/interfaces`

```
auto lo
iface lo inet loopback

allow-hotplug eth0
auto eth0
iface eth0 inet manual
auto br0
iface br0 inet static
    address 192.168.1.6
    network 192.168.1.0
    netmask 255.255.255.0
    broadcast 192.168.1.255
    gateway 192.168.1.1
# nameservice is configured in systemd-resolved
# dns-nameservers 192.168.1.5 8.8.8.8
# dns-search example.com
bridge_ports eth0
bridge_stp off
bridge_fd 0
bridge_maxwait 0
```

Install Virtualization

on servers

```
qemu-kvm libvirt-bin libvirt-daemon-system virtinst libvirt-clients qemu-utils ovmf
```

on workstations

```
libvirt-clients virt-viewer spice-client-gtk virt-manager qemu-utils remmina remmina-plugin-spice
```

then

```
adduser $USER libvirt
```

Speaker notes

The ovmf package installs EFI support. The gtk and remmina spice packages enable vinaigre and remmina support for spice.

Enable EFI

Even though we installed OVMF, it is still disabled in the QEMU configuration.

Edit */etc/libvirt/qemu.conf* to uncomment these lines:

```
nvram = [  
    "/usr/share/OVMF/OVMF_CODE.fd:/usr/share/OVMF/OVMF_VARS.fd",  
    "/usr/share/OVMF/OVMF_CODE.secboot.fd:/usr/share/OVMF/OVMF_VARS.fd",  
    "/usr/share/AAVMF/AAVMF_CODE.fd:/usr/share/AAVMF/AAVMF_VARS.fd",  
    "/usr/share/AAVMF/AAVMF32_CODE.fd:/usr/share/AAVMF/AAVMF32_VARS.fd"  
]
```

Restart the host.

Define first storage pool

```
virsh pool-define-as main --type dir --target /vm
virsh pool-list (nothing)
virsh pool-start main
virsh pool-autostart main
```

```
virsh # pool-list
```

Name	State	Autostart

main	active	yes

Use Command Line to Create a Guest

```
virt-install \  
--name=debian980 \  
--ram=512 \  
--disk path=/vm/debian980.qcow2,size=8 \  
--vcpus=1 \  
--os-type=linux \  
--network bridge=br0 \  
--console pty,target_type=serial \  
--nographics \  
--location /vm/debian-9.8.0-amd64-netinst.iso \  
--extra-args=console=ttyS0;
```

When you finish the installation `virsh destroy debian980`.

Speaker notes

`destroy` is the shutdown command and `undefine` does what you expect `destroy` to do.

View the Configuration

`virsh edit debian980` to see the configuration. You can also find the config file at `/etc/libvirt/qemu/debian980.xml`.

Add the following between existing entries

```
<graphics type='spice' port='6001' autoport='no' listen='127.0.0.1'>  
<listen type='address' address='127.0.0.1' />  
</graphics>
```

Save the file then start it with `virsh start debian980`.

Graphics Support

- QEMU supports two virtual displays. VNC and Spice. Spice is the better choice but sometimes VNC works and Spice doesn't.
- The build I did for this presentation on Cosmic was the first time I ever had Spice work in Virtual Machine Manager on Ubuntu.

Graphics Support

- Spice offers better performance and supports TLS encryption. For a simple setup like this you'll probably just use SSH for both encryption and access control, by having spice/vnc only listen on localhost and tunneling any remote connections.
- I have a more detailed article at <http://techinfo.brainbuz.org/2019/02/07/kvm-virtualization-vmm-and-spice-on-ubuntu/>.

Launch Virtual Machine Manager GUI

Permissions Problem?

If when you try to run `virsh` or `vmm` as a user (that you added to the `libvirt` group) it fails with a permissions problem the likely cause is that `/var/run/libvirt` and `/var/run/libvirt/libvirt-sock` and possibly other files are not read write by the `libvirt` group.

View Host Details


Right click the host and select Details.

Speaker notes

The second screen shows the storage tab.

Create A New Guest

New VM

 **Create a new virtual machine**
Step 1 of 5

Connection: QEMU/KVM

Choose how you would like to install the operating system

- Local install media (ISO image or CDROM)
- Network Install (HTTP, FTP, or NFS)
- Network Boot (PXE)
- Import existing disk image

New VM

Create a new virtual machine Step 2 of 5

Locate your install media

Use CDROM or DVD

No media detected (/dev/sr0) ▾

Use ISO image:

/vm/Fedora-Workstation-Live-x86_64-29-1.2.iso ▾

Automatically detect operating system based on install media

OS type: Linux

Version: Fedora 28

New VM



Create a new virtual machine

Step 4 of 5

- Enable storage for this virtual machine
- Create a disk image for the virtual machine
- Select or create custom storage

20.0 - + GiB

7.9 GiB available in the default location

Manage... /vm/fedora28.qcow2

Cancel

Back

Forward

New VM



Create a new virtual machine

Step 5 of 5

Ready to begin the installation

Name:

OS: Fedora 28

Install: Local CDROM/ISO

Memory: 2048 MiB

CPUs: 1

Storage: /vm/fedora28.qcow2

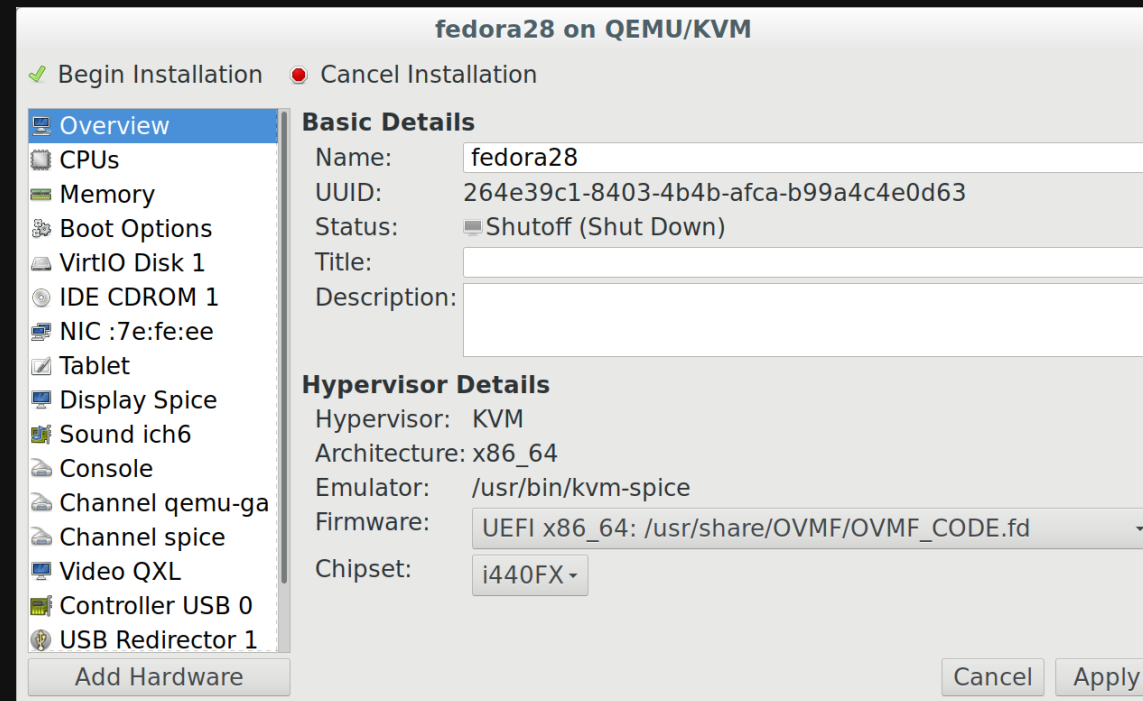
Customize configuration before install

▾ Network selection

Cancel

Back

Finish



Manually Select UEFI firmware.

Apply and Begin Installation.

Links

- <https://www.qemu.org/>
- <https://libvirt.org/>
- https://www.linux-kvm.org/page/Main_Page
- <https://help.ubuntu.com/lts/serverguide/libvirt.html.en>
- <https://forum.level1techs.com/t/how-fast-is-kvm-host-vs-virtual-machine-performance/110192>
- <https://help.ubuntu.com/community/KVM/Networking>
- <http://techinfo.brainbuz.org/2019/02/07/kvm-virtualization-vmm-and-spice-on-ubuntu/>

© 2018, 2019 John Karr

<http://techinfo.brainbuz.org>

<http://techinfo.brainbuz.org/slides>


```
Reveal.initialize({ // ... width: "100%", height: "100%", margin: 0, minScale: 1, maxScale: 1 });
```