ADVANCED CLOUD COMPUTING

CONTAINERS

DEEP DIVE, PART 2

CONTAINER DEVICE INTERFACE (CDI)

SPECIFIC DEVICES (EG. GPU)

specification, for container-runtimes, to support third-party devices.

- abstract notion of a device as a resource,
- devices are specified by a Fully qualified name (FQN) `vendor.com/class=unique_name`

SCOPE: enabling containers to be device aware. NO resource management.

WHY?

Theory: passing a device to a container = exposing a device node

Practice: is more than that...

- exposing more than one device node,
- mounting files from the runtime namespace,
- hiding procfs entries,
- Performing compatibility checks
  (Can this container run on this device?).
- Performing runtime-specific operations
  (e.g: VM vs Linux container-based runtimes).
- Performing device-specific operations
  (e.g: scrubbing the memory of a GPU).

WHY?

To converge in from the currenty situation of vendors multiple plugins, for different runtimes or even directly contribute vendor-specificcode in the runtime.

HOW DOES IT WORKS?

- JSON or YAML in `/etc/cdi` and `/var/run/cdi` (depends on runtime configuration)
- FQN's should be passed to the runtime using the container engine options

Usually there are tools to fill these folders

```
nvidia-ctk cdi generate --output=/etc/cdi/nvidia.yaml
```

RUNNING AN LLM

```
podman run --device nvidia.com/gpu=all \
   -v ~/AI/ollama:/root/.ollama \
   -p 11434:11434 \
   --security-opt=label=disable \
   --name ollama ollama/ollama
```

Usage

```
podman exec -it ollama ollama run gemma:7b
```

RUNNING AN LLM

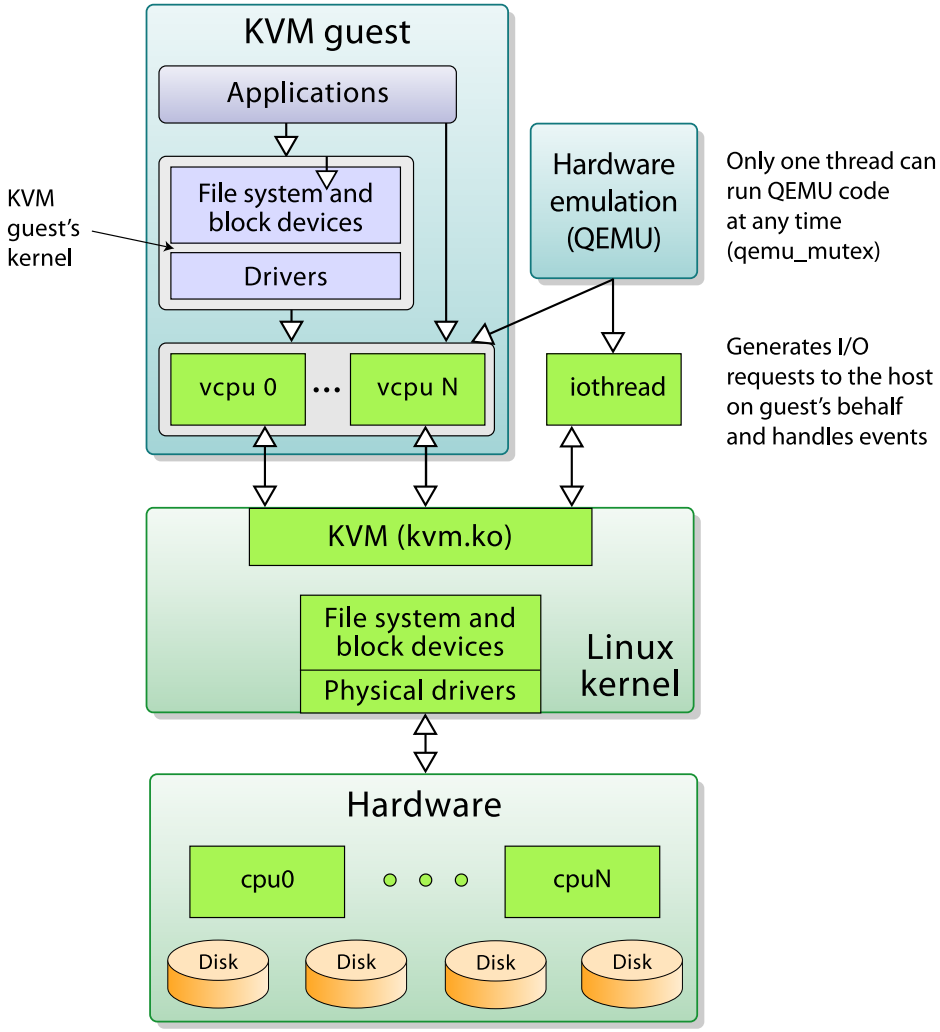
```
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 560.35.03              Driver Version: 560.35.03      CUDA Version: 12.6     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |                  MIG M. |
|=========================================+========================+======================|
|   0  NVIDIA GeForce MX250           Off |   00000000:3C:00.0 Off |                  N/A |
| N/A   71C    P0             N/A / ERR!  |    1369MiB /   2048MiB |      0%      Default |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+

+-----------------------------------------------------------------------------------------+
| Processes:                                                                              |
|  GPU   GI   CI        PID   Type   Process name                              GPU Memory |
|        ID   ID                                                               Usage      |
|=========================================================================================|
|    0   N/A  N/A   1775090      C   ...unners/cuda_v12/ollama_llama_server       1366MiB |
+-----------------------------------------------------------------------------------------+
```

KUBERNETS

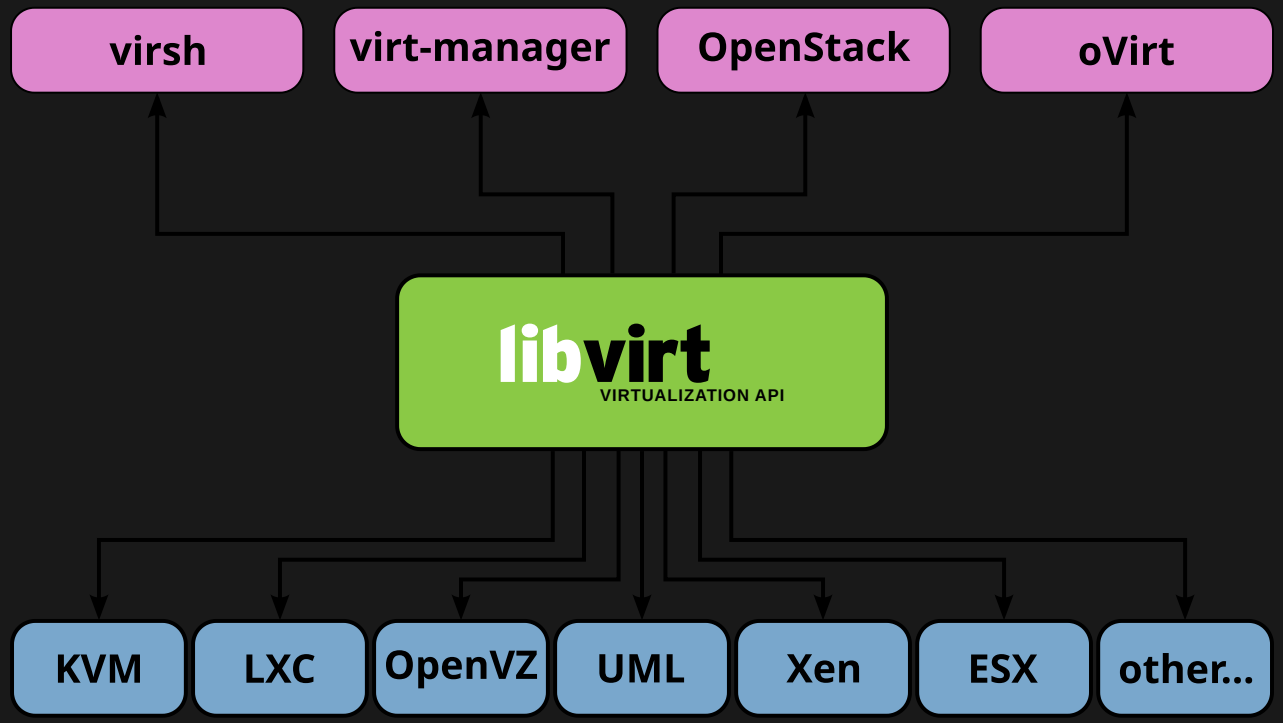
DEPLOYMENT

IN A REALISTIC ENVIRONMENT

QUEMU/KVM


KVM guest
Applications
File system and block devices
Drivers
vcpu 0
vcpu N
KVM guest's kernel
Hardware emulation (QEMU)
Only one thread can run QEMU code at any time (qemu_mutex)
iothread
Generates I/O requests to the host on guest's behalf and handles events
KVM (kvm.ko)
File system and block devices
Physical drivers
Linux kernel
Hardware
cpu0
cpuN
Disk
Disk
Disk
Disk

VIRSH


virsh
virt-manager
OpenStack
oVirt
libvirt
VIRTUALIZATION API
KVM
LXC
OpenVZ
UML
Xen
ESX
other...

VIRSH

```
<network>
    <name>kub-devel</name>
    <forward mode="nat">
        <nat>
            <port start="1024" end="65535">
        </port></nat>
    </forward>
    <ip address="192.168.133.1" netmask="255.255.255.0">
        <dhcp>
            <range start="192.168.133.2" end="192.168.133.29">
        </range></dhcp>
    </ip>
</network>
```

VAGRANT

tool that simplifies the process of setting up and managing virtual development environments

VAGRANT

KEY CONCEPTS:

- Install a Provider (libvirt)
- Create a Vagrantfile (ruby)
- Start/destroy the VM
- SSH into the VM
- Provision the VM

VAGRANTFILE

```
servers = [
  { :hostname => "k01", :ip => "192.168.133.80" },
  # { :hostname => "k02", :ip => "192.168.133.81" },
]

Vagrant.configure("2") do |config|
  config.vm.box = "fedora/41-cloud-base"

  config.vm.provider :libvirt do |lv|
    lv.qemu_use_session = false
    lv.memory = 2048
    lv.cpus = 2
  end

  servers.each do |conf|
```

VIRTUAL ORFEO

Name	Last commit	Last update
.reuse	started using reuse	1 year ago
LICENSES	fix licences	1 year ago
manifests	rm submodule	1 month ago
playbooks	remove submodule dependency	1 month ago
scripts	Add license	10 months ago
vagrantfiles	more powerfull ipa vm	1 month ago
.gitignore	remove personal git ignore	7 months ago
.gitmodules	rm submodule	1 month ago
README.md	updated command to checkout correct ...	4 months ago
requirements.txt	missing requirement	7 months ago

Gitlab

Repo

SYSTEM SETUP

```
# Load modules
modprobe overlay
modprobe br_netfilter

# make load permanent
cat << EOF | tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF

# change kernel parameters
cat << EOF |  tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-iptables = 1
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward = 1
```

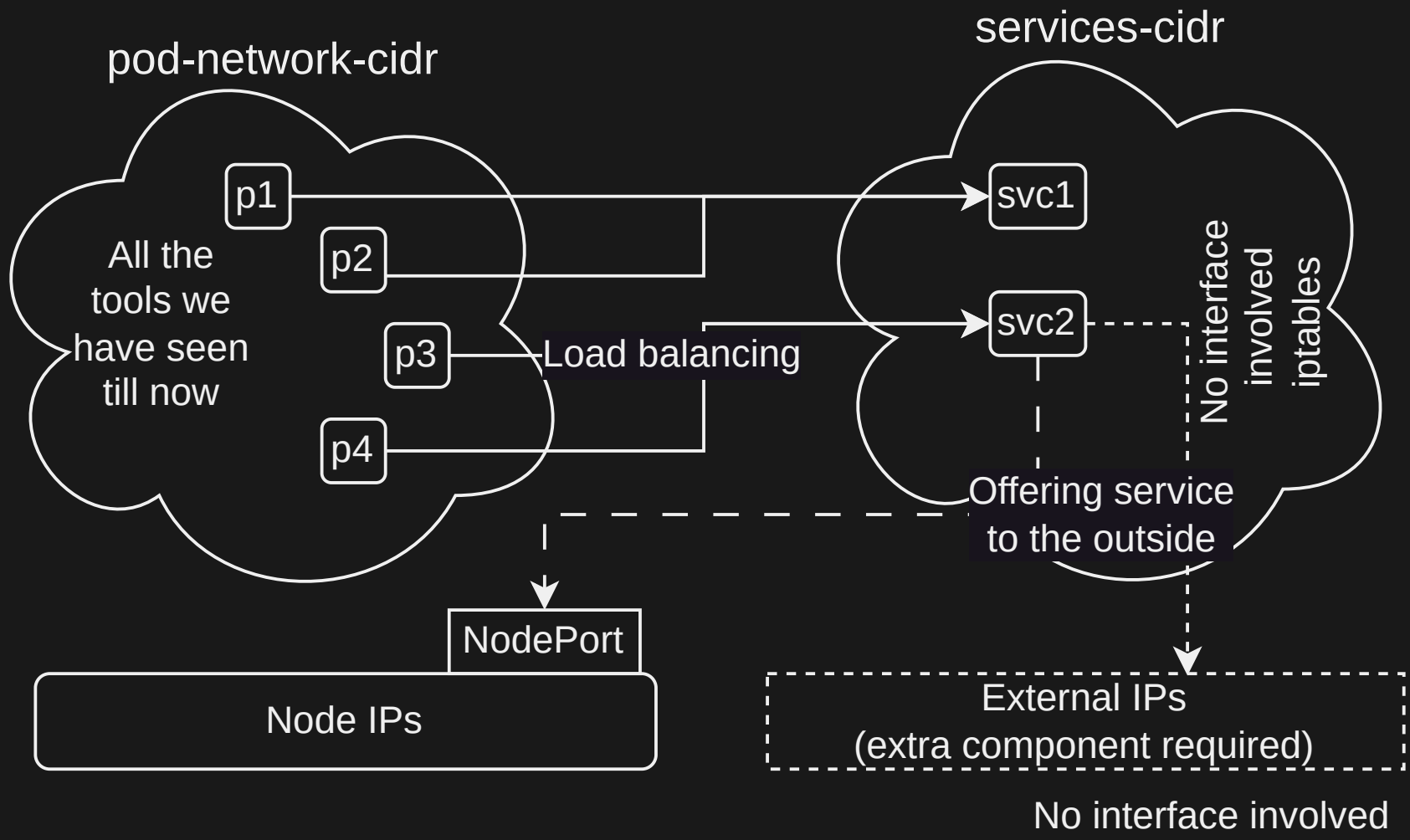
installation

```
# attention to exclude!!
cat << EOF | tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/re
EOF
```

KUBERNETES INSTALLATION

```
# utils
dnf install iproute-tc wget vim bash-completion bat -y
# CRI-o
dnf install crio -y
# real kube
dnf install -y kubelet kubeadm kubectl
```

NETWORKING, AGAIN



KUBERNETES INSTALLATION

```
sed -i 's/10.85.0.0\/16/10.17.0.0\/16/' /etc/cni/net.d/100-cri
systemctl enable --now crio
systemctl enable --now kubelet

kubeadm init --pod-network-cidr=10.17.0.0/16
# --services-cidr=10.96.0.0/12 /default
# --control-plane-endpoint 192.168.132.80 /needed for HA
```

MINIKUBE

- cluster running on your local machine
- Provides a hands-on environment for learning
- Ideal for DevOps

KUBERNETES CLIENTS

- kubectl `alias k=kubectl`, official
  tools: kubectx, kubens
- k9s, cli, non official, way better
- OpenLens, nice if you don't like CLI tools

best practice: remote control you cluster from your PC

k version

```
[root@k01 ~]# kubectl version
Client Version: v1.31.3
Kustomize Version: v5.4.2
Server Version: v1.31.3
```

client can be ± 3 minor version from server

k get nodes

```
[root@k01 ~]# k get nodes
NAME   STATUS   ROLES           AGE   VERSION
k01    Ready    control-plane   40m   v1.31.3
```

k get nodes k01

```
Name:               k01
Roles:              control-plane
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/os=linux
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=k01
                    kubernetes.io/os=linux
                    node-role.kubernetes.io/control-plane=
                    node.kubernetes.io/exclude-from-external-load-balancers=
Annotations:        kubeadm.alpha.kubernetes.io/cri-socket: unix:///var/run/crio/crio.sock
                    node.alpha.kubernetes.io/ttl: 0
                    volumes.kubernetes.io/controller-managed-attach-detach: true
CreationTimestamp:  Fri, 06 Dec 2024 00:08:04 +0000
Taints:             node-role.kubernetes.io/control-plane:NoSchedule
Unschedulable:      false
Lease:
  HolderIdentity:  k01
  AcquireTime:     unset
  RenewTime:       Fri, 06 Dec 2024 00:48:49 +0000
Conditions:
  Type             Status  LastHeartbeatTime                 LastTransitionTime                Re
  ----             ------  -----------------                 ------------------                --
  MemoryPressure   False   Fri, 06 Dec 2024 00:43:59 +0000   Fri, 06 Dec 2024 00:08:02 +0000   Ku
  DiskPressure     False   Fri, 06 Dec 2024 00:43:59 +0000   Fri, 06 Dec 2024 00:08:02 +0000   Ku
```

LABELS, ANNOTATIONS AND TAINTS

- labels: k, v are responsible for node info, can be used to schedule pods
- annotations: ~ 3rd party labels
- taints: influenced by labels and conditions can have effect on pods (eg. no schedule)

TAINTS

- NoSchedule: No schedule but no eviction take place
- Noexecute: Pods are evicted (unless they tollerate the taint)
- PreferNoSchedule: To avoid scheduling on current node (eg. node under pressure)

`kubelet` can add/remove label, this reflects on taints

k get all -A

```
NAMESPACE     NAME                                 READY   STATUS    RESTARTS   AGE
kube-system   pod/coredns-7c65d6cfc9-f482q         1/1     Running   0          53m
kube-system   pod/coredns-7c65d6cfc9-xndxh         1/1     Running   0          53m
kube-system   pod/etcd-k01                         1/1     Running   0          53m
kube-system   pod/kube-apiserver-k01               1/1     Running   0          53m
kube-system   pod/kube-controller-manager-k01      1/1     Running   0          53m
kube-system   pod/kube-proxy-428rs                 1/1     Running   0          53m
kube-system   pod/kube-scheduler-k01               1/1     Running   0          53m

NAMESPACE     NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)
default       service/kubernetes   ClusterIP   10.96.0.1    none        443/TCP
kube-system   service/kube-dns     ClusterIP   10.96.0.10   none        53/UDP,53/TCP,9153/TCP

NAMESPACE     NAME                        DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NO
kube-system   daemonset.apps/kube-proxy   1         1         1       1            1           ku

NAMESPACE     NAME                      READY   UP-TO-DATE   AVAILABLE   AGE
kube-system   deployment.apps/coredns   2/2     2            2           53m

NAMESPACE     NAME                                 DESIRED   CURRENT   READY   AGE
kube-system   replicaset.apps/coredns-7c65d6cfc9   2         2         2       53m
```

COMPONETS REFRESH:

- kube-proxy: **daemonset** responsible for routing services
- core-dns: **deployment** internal dns
- etcd: db for the cluster state
- api server: api endpoint
- control manager: keeps track or the cluster status
- scheduler: node selection

NAMESPACES

they are like a folders to organize cluster object and compartimentalize resources (multitenancy)

- `-n namespace`: select namespace
- `-A`: all namespaces

CONTEXT

Context of your interaction with the API, cluster + namespaces

```
k config set-context my-context --namespace=my-namespace
k config use-context my-context
```

DEFINITIONS

Resource: is an endpoint in the Kubernetes API that stores a collection of API objects of a certain kind.

A resource is part of a declarative API, used by Kubernetes client libraries and CLIs, or kubectl. It can lead to "custom resource", an extension of the Kubernetes API.

DEFINITIONS

Object: is a persistent entities in the Kubernetes system.

A Kubernetes object is a "record of intent"

you create the object, the Kubernetes system will constantly work to ensure that object exists. By creating an object, you're effectively telling the Kubernetes system what you want your cluster's workload to look like; this is your cluster's desired state.

KUBECTL VIEW OBJECT COMMAND STRUCTURE

kubectl get {{ resource-name }} {{ obj-name }}

```
k get pods
k get pods {{ pod-name }} -o yaml
```

k describe {{ resource-name }} {{ resource-obj }} more detailed information

k explain {{ resource-name }} doc (very useful)

CREATING, UPDATING, AND DESTROYING OBJ

```
k apply -f x.yaml
k delete -f x.yaml
```