Napredno korištenje operacijskog sustava Linux

6. Virtualizacija

Marin Petričević, Dominik Barbarić
Nositelj: doc.dr.sc. Stjepan Groš

Sveučilište u Zagrebu
Fakultet elektrotehnike i računarstva

07.04.2017

Sadržaj

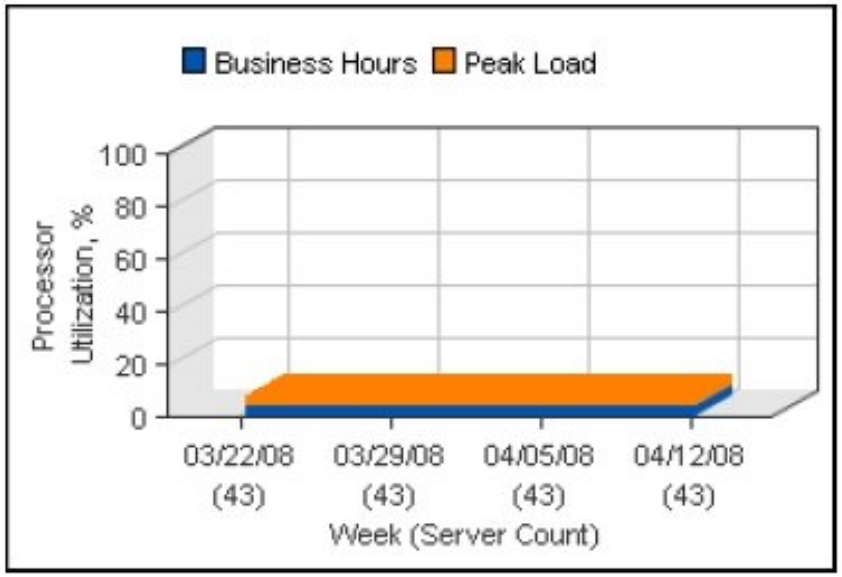
Virtualizacija

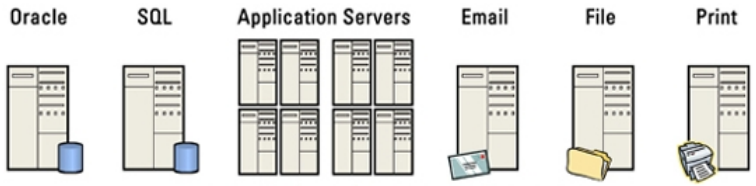
- Emulacija više fizickih računala na jednom
- Virtualizacija dijeli resurse fizičkog računala na više emuliranih računala
- Virtualna računala su izolirana međusobno i od fizičkog računala

- Standardizacija produkcijske / development okoline
- Prenosivost okoline
- Lakši deployment

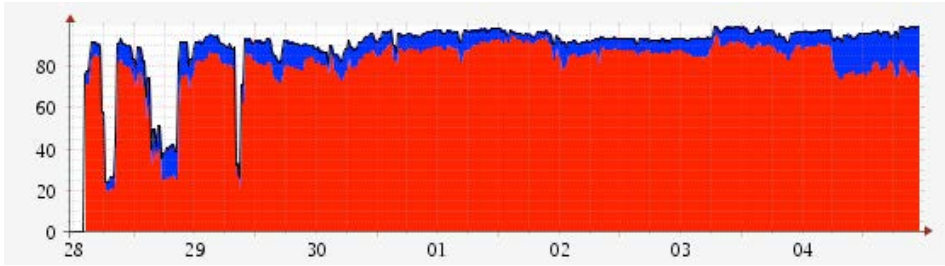
Virtualizacija

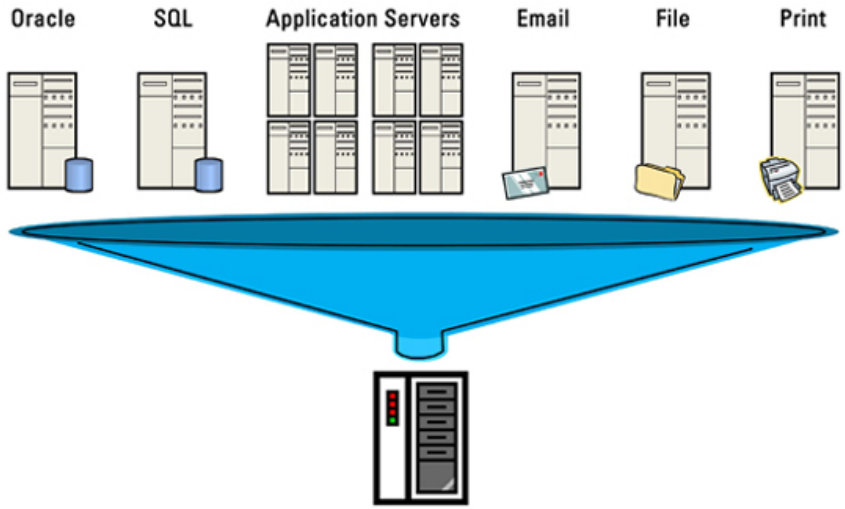
Bez virtualizacije





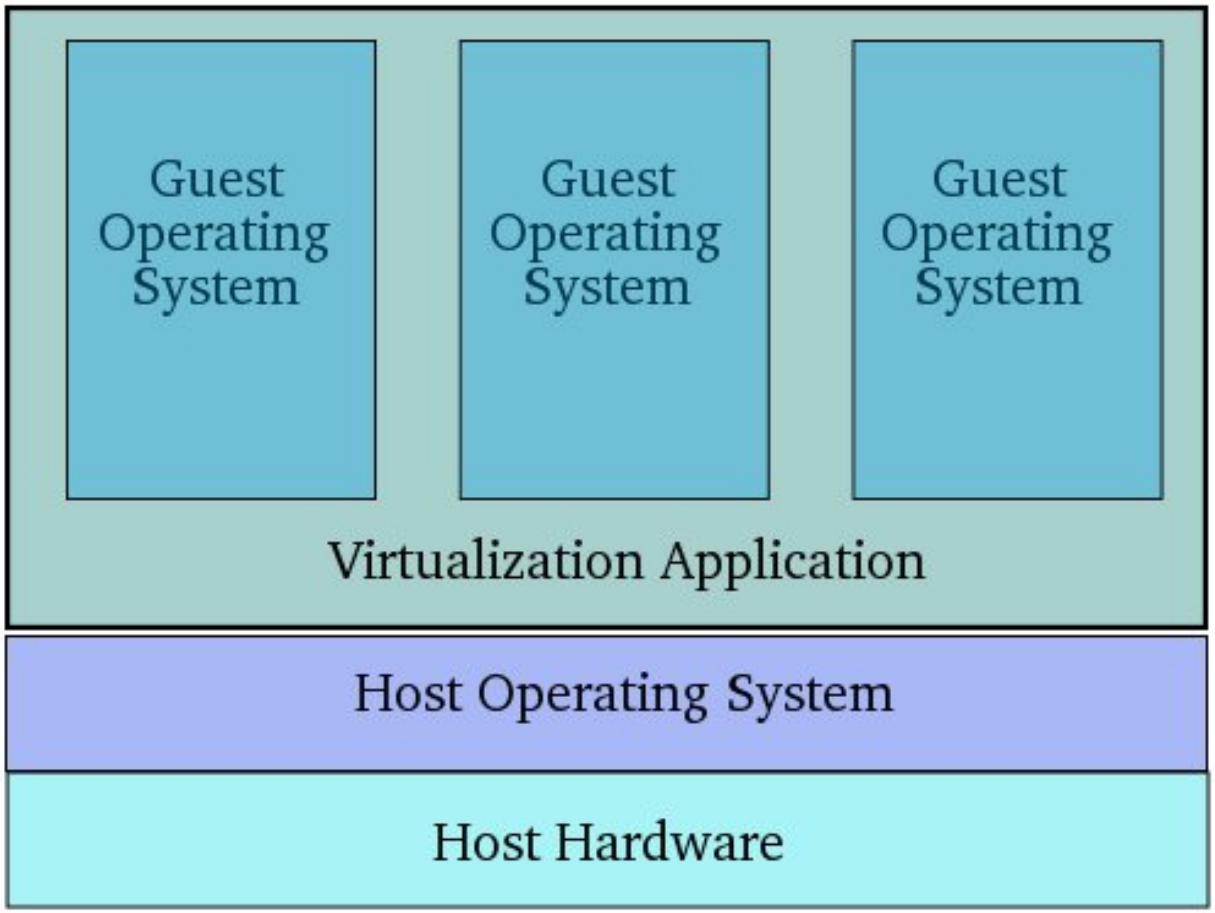
Sa virtualizacijom





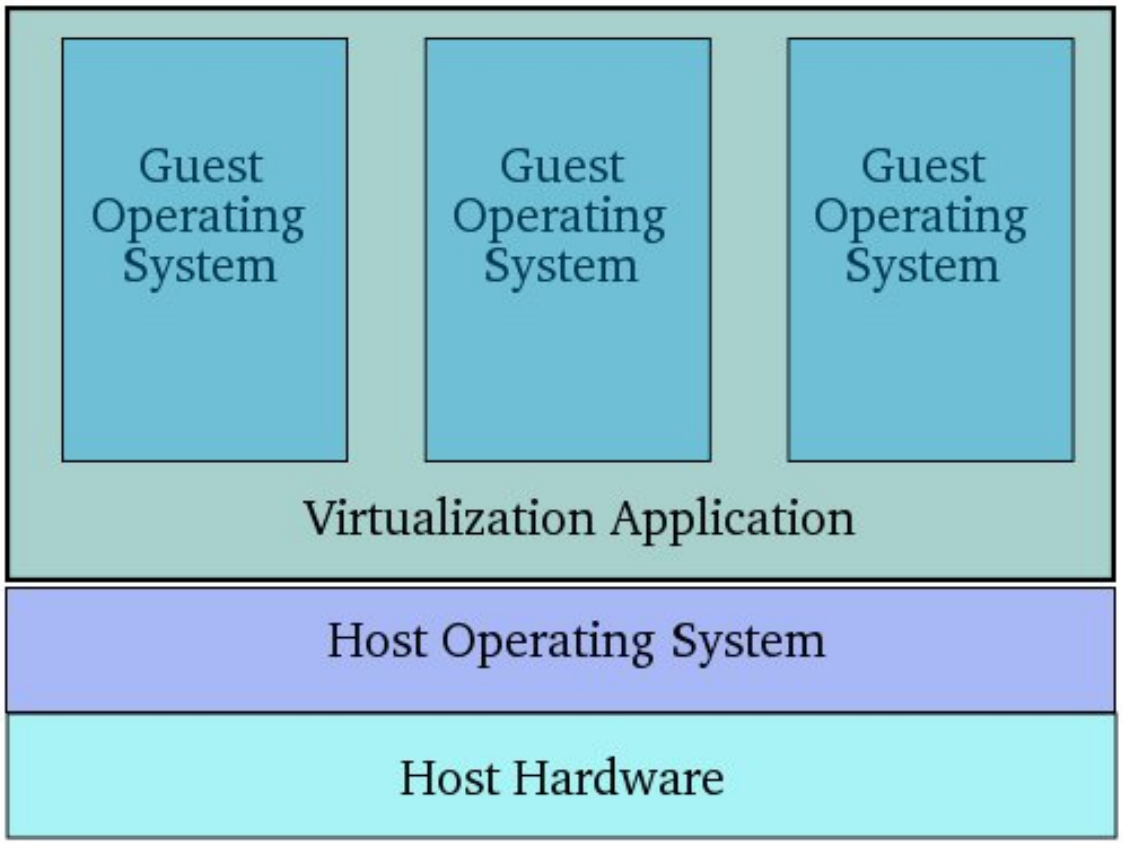
Guest OS virtualizacija

- Virtualizaciju obavlja aplikacija unutar koje se pokreće cijeli operacijski sustav virtualnog računala



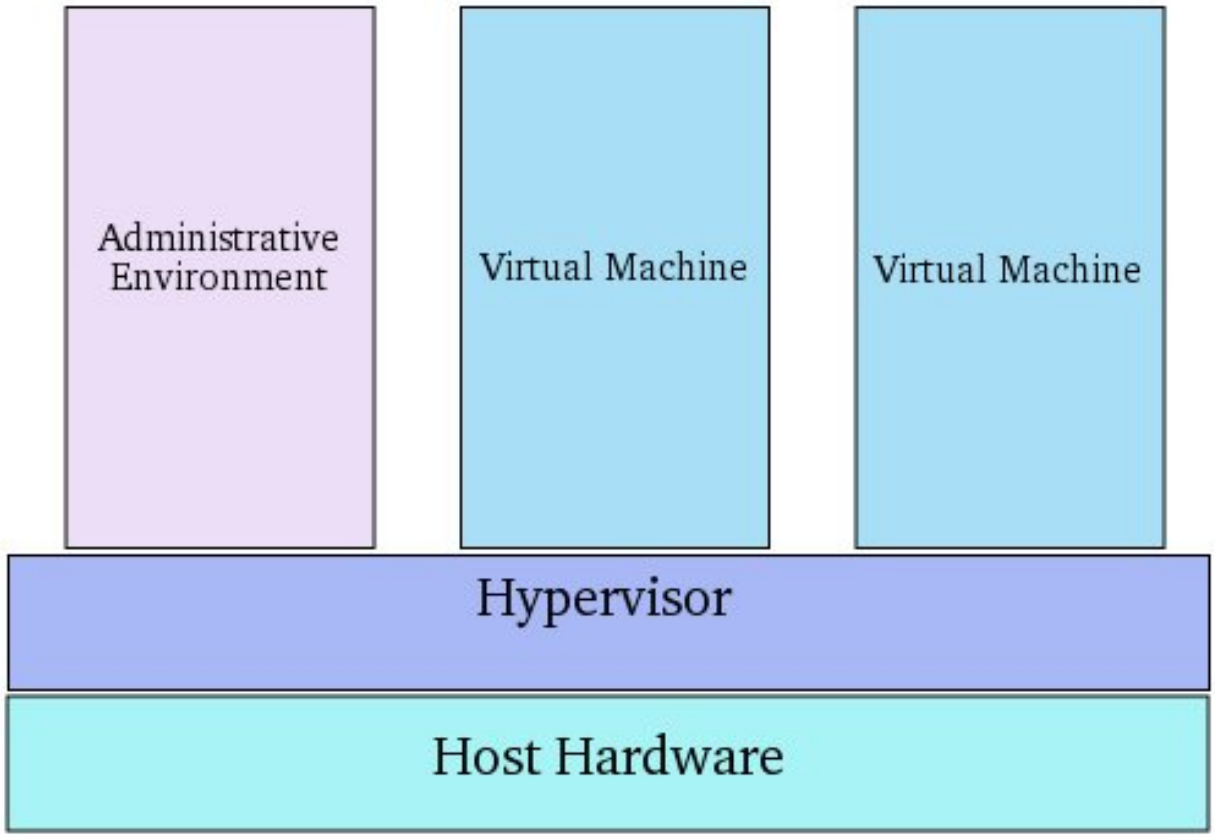
Guest OS virtualizacija

- VirtualBox, VMware
- Dobro: Lakoća korištenja, razni OSovi
- Loše: Performanse



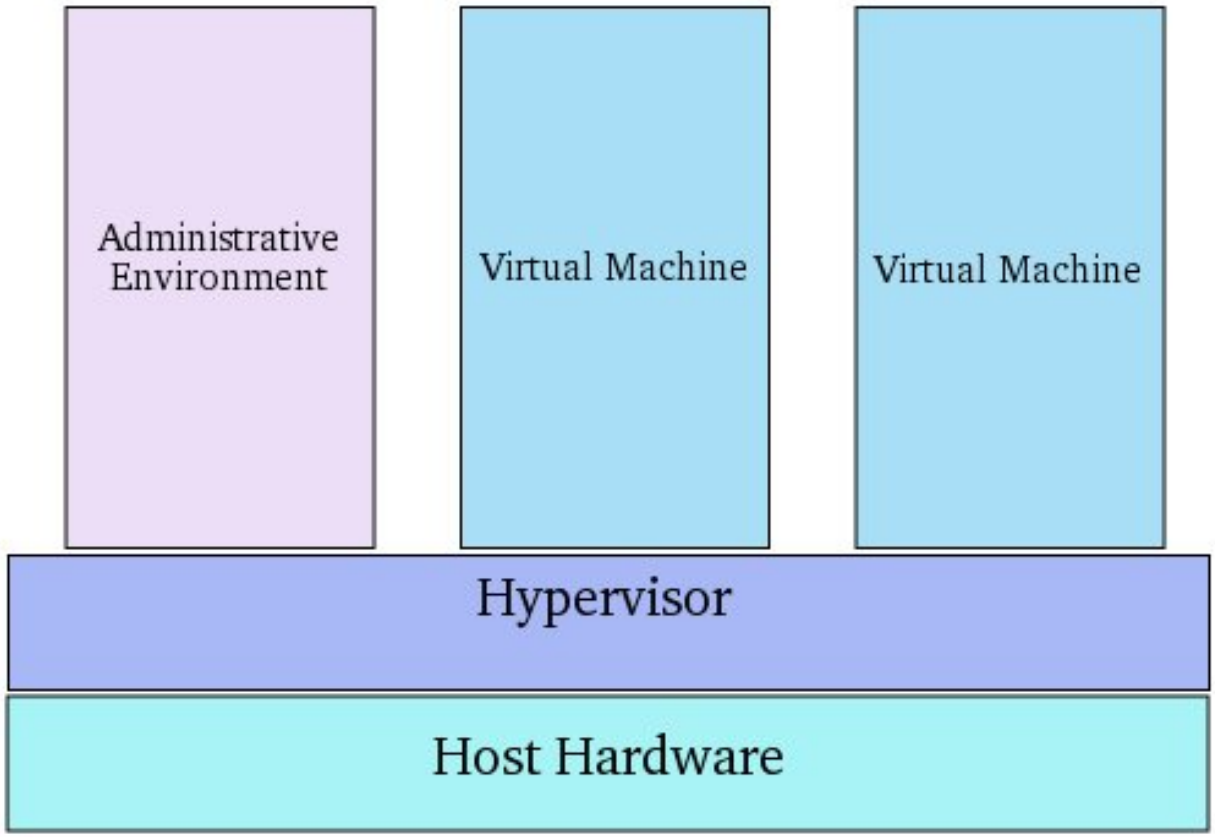
Hypervisor

- Operacijski sustav namijenjen virtualizaciji



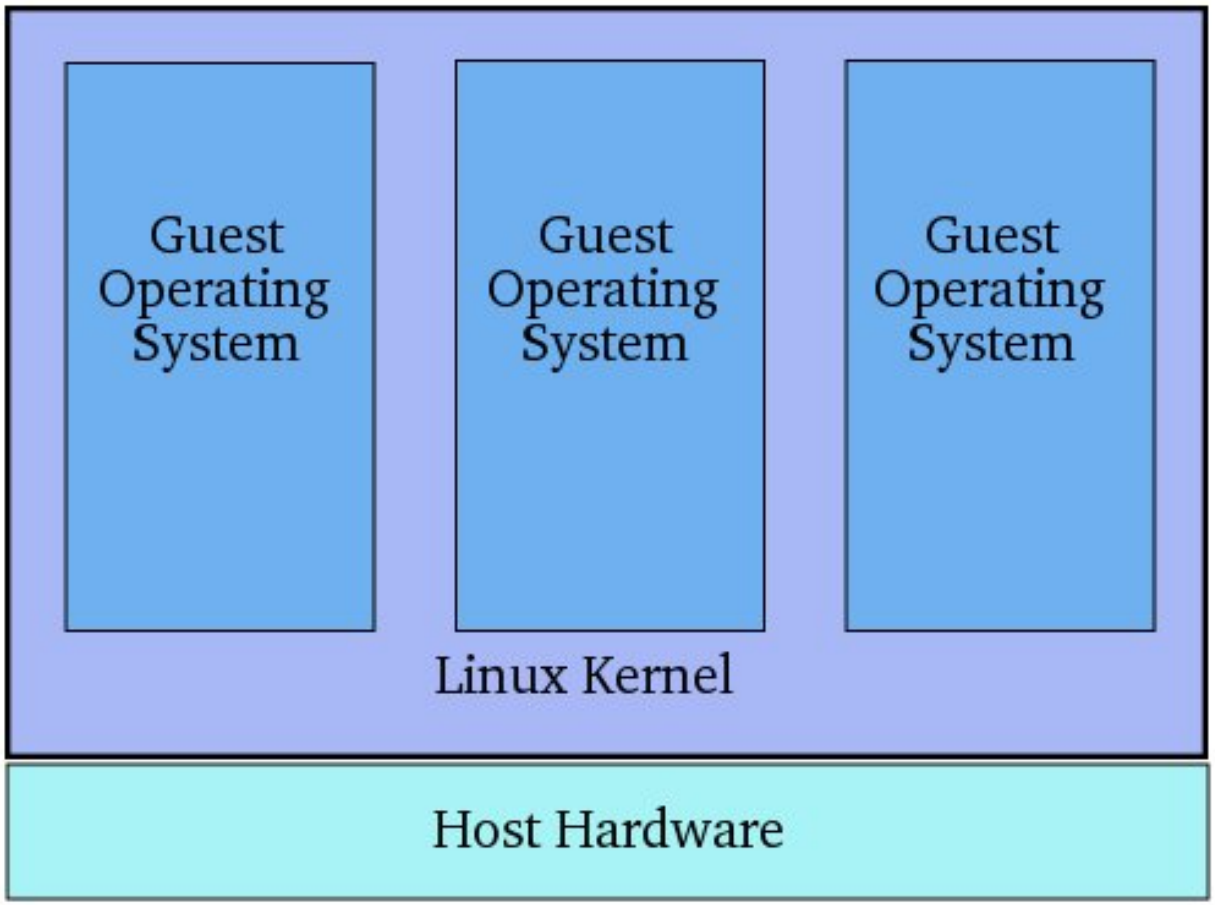
Hypervisor

- Xen, HyperV, VMware, ...
- Dobro: Odlične performanse, dobra izolacija, proizvoljni OSovi
- Loše: Kompliciranije



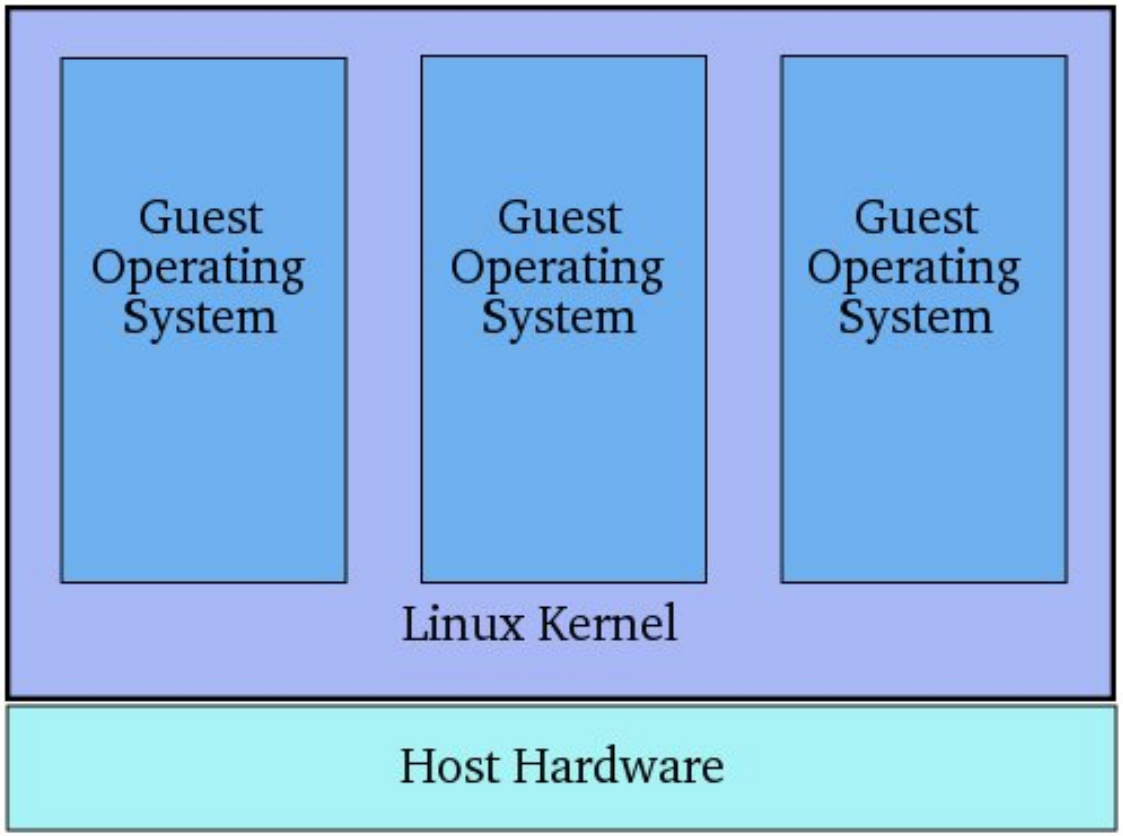
Kernel virtualizacija

- Kernel ima podršku za virtualizaciju
- Vrsta hypervisora



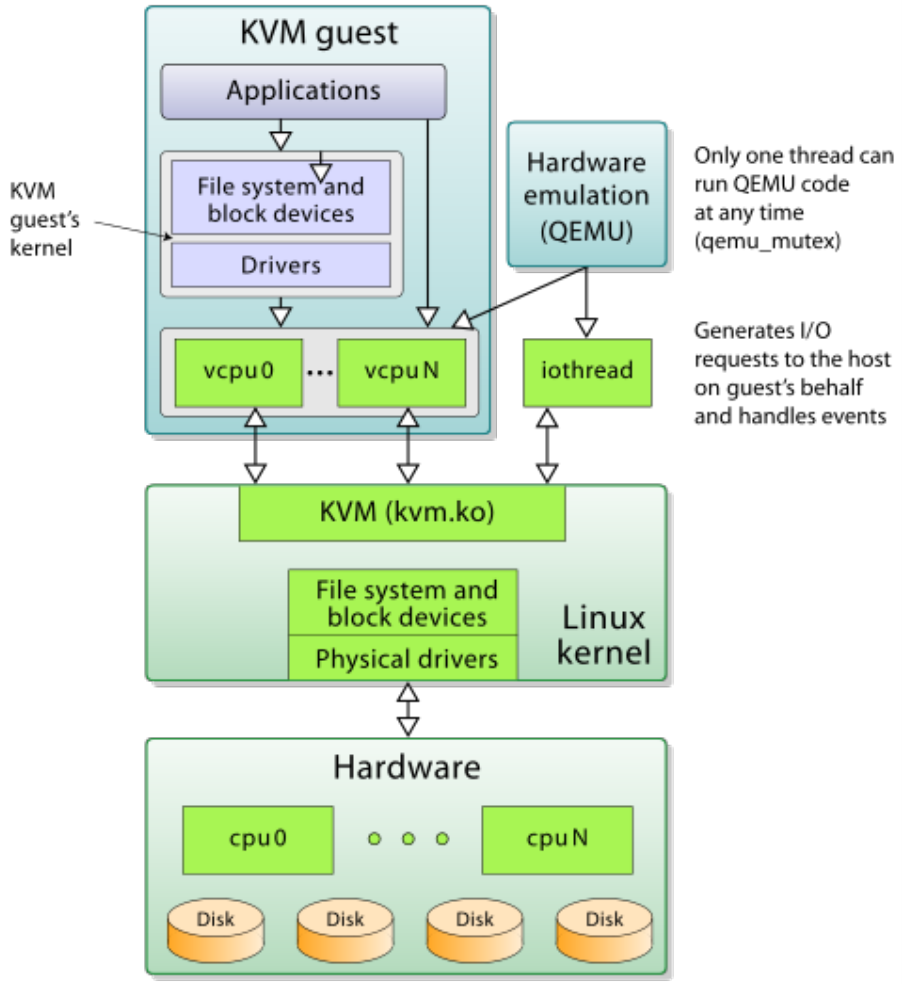
Kernel virtualizacija

- KVM - Kernel-based Virtual Machine
- Dobro: Odlične performanse
- Loše: Kompliciranije, samo Linux



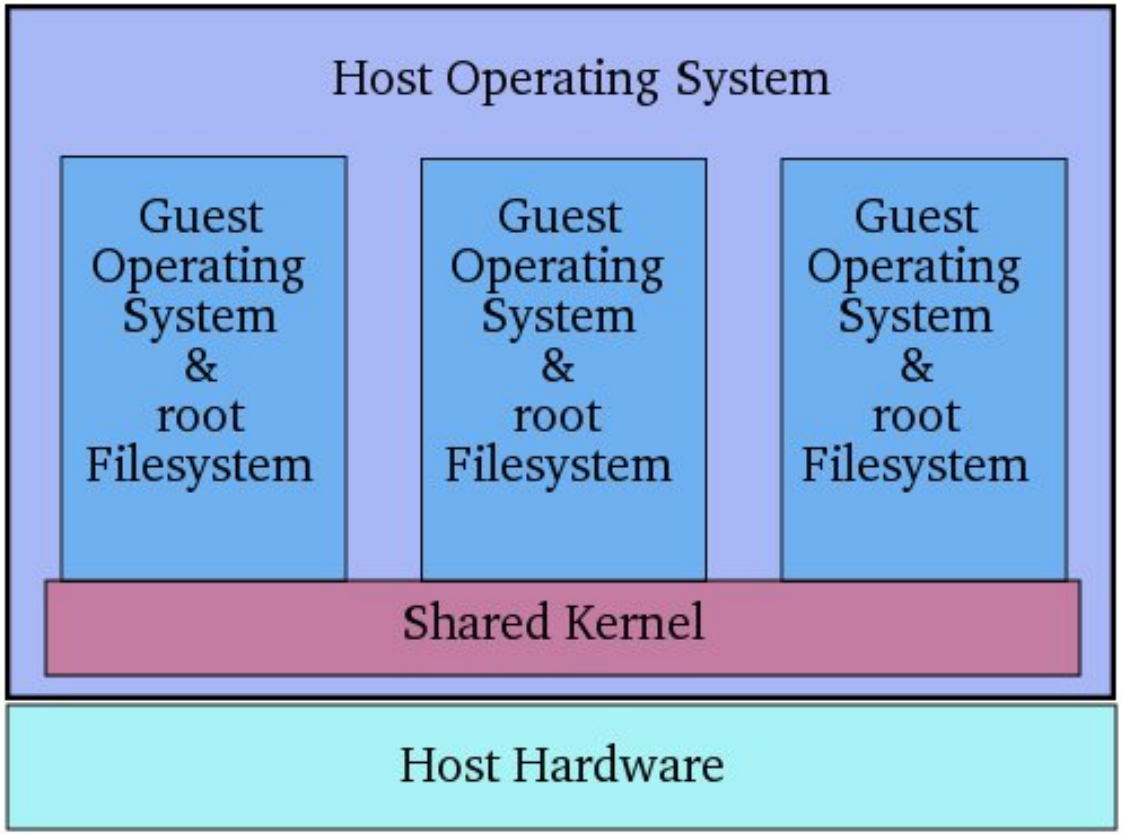
QUEMU

- Emulira različite arhitekture neovisno o arhitekturi hosta
- Npr. pokretanje ARM verzije linuxa na x64 procesoru
- Userspace virtualizator za KVM i Xen



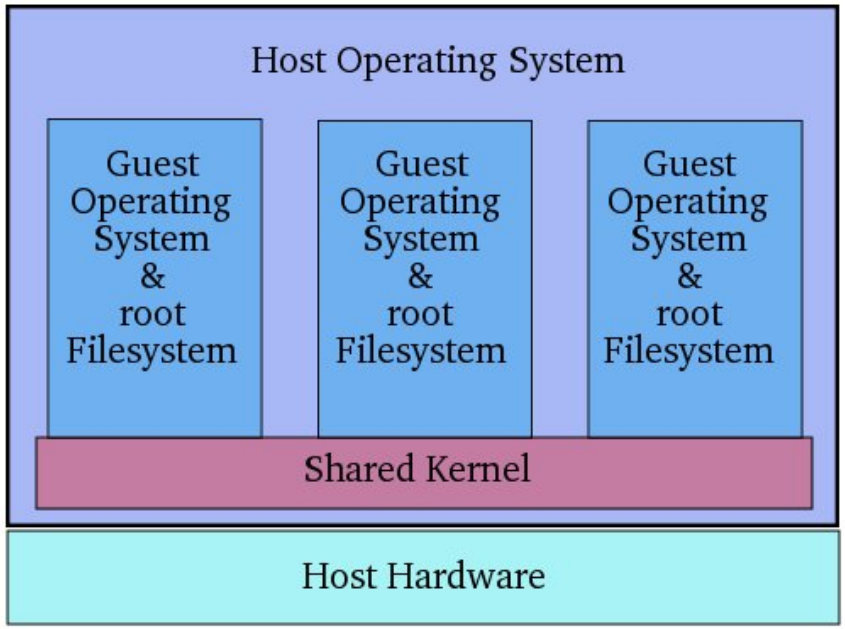
Shared kernel virtualizacija

- Virtualna računala dijele zajednički (Linux / UNIX) kernel
- Često ne koriste naziv "virtual machine" nego "container"



Shared kernel virtualizacija

- Na Linuxu: lxc, docker, runC...
- Drugi UNIX: FreeBSD jails, Solaris zones
- Dobro: lakoća korištenja, odlične performanse!
- Loše: Samo OSovi s kompatibilnim kernelima, nešto lošija izolacija



cgroups

- Control groups
- Izolacija na razini procesa - dijeljeni kernel.

- Grupiranje procesa i razdjela resursa
  - Limitiranje resursa - CPU, RAM, IO, itd.
  - Prioriteti grupa procesa
  - Pokretanje i zaustavljanje grupa procesa
- *Namespace isolation*
  - *Skrivanje* resursa koji nisu dodijeljeni procesu - PIDovi, mreža, file system, IPC, korisnici
  - Skrivanjem svih resursa na host računalu se nekom procesu ili grupi procesa može dati privid da imaju računalo samo za sebe

cgroups

```
# cgcreate -a user -g memory,cpu:groupname
$ ls -l /sys/fs/cgroup/memory/groupname
total 0
-rwxrwxr-x 1 user root 0 Sep 25 00:39 cgroup.event_control
-rwxrwxr-x 1 user root 0 Sep 25 00:39 cgroup.procs
-rwxrwxr-x 1 user root 0 Sep 25 00:39 cpu.rt_period_us
-rwxrwxr-x 1 user root 0 Sep 25 00:39 cpu.rt_runtime_us
-rwxrwxr-x 1 user root 0 Sep 25 00:39 cpu.shares
-rwxrwxr-x 1 user root 0 Sep 25 00:39 notify_on_release
-rwxrwxr-x 1 user root 0 Sep 25 00:39 tasks
$ cgexec   -g memory,cpu:groupname/foo bash
```

LXC

- Linux containers
- Shared kernel virtualizacija
- Implementira cgroups

- Virtualna računala se kreiraju pomoću template skripti `/usr/share/lxc/templates`
- Defaultno instalacija u `/var/lib/lxc/NazivVM`

```
# lxc-create -n archVM -t /usr/share/lxc/templates/lxc-archlinux
# lxc-start -n archVM
# lxc-attach -n archVM
# lxc-stop -n archVM
```

Docker

- Virtualizacija na razini procesa / aplikacije
- Aplikaciji izgleda kao da ima OS sama za sebe
- Dijeli kernel sa host OSom i drugim containerima

- Docker je više od tehnike virtualizacije, docker je ekosustav za izgradnju, deployment i upravljanje containerima

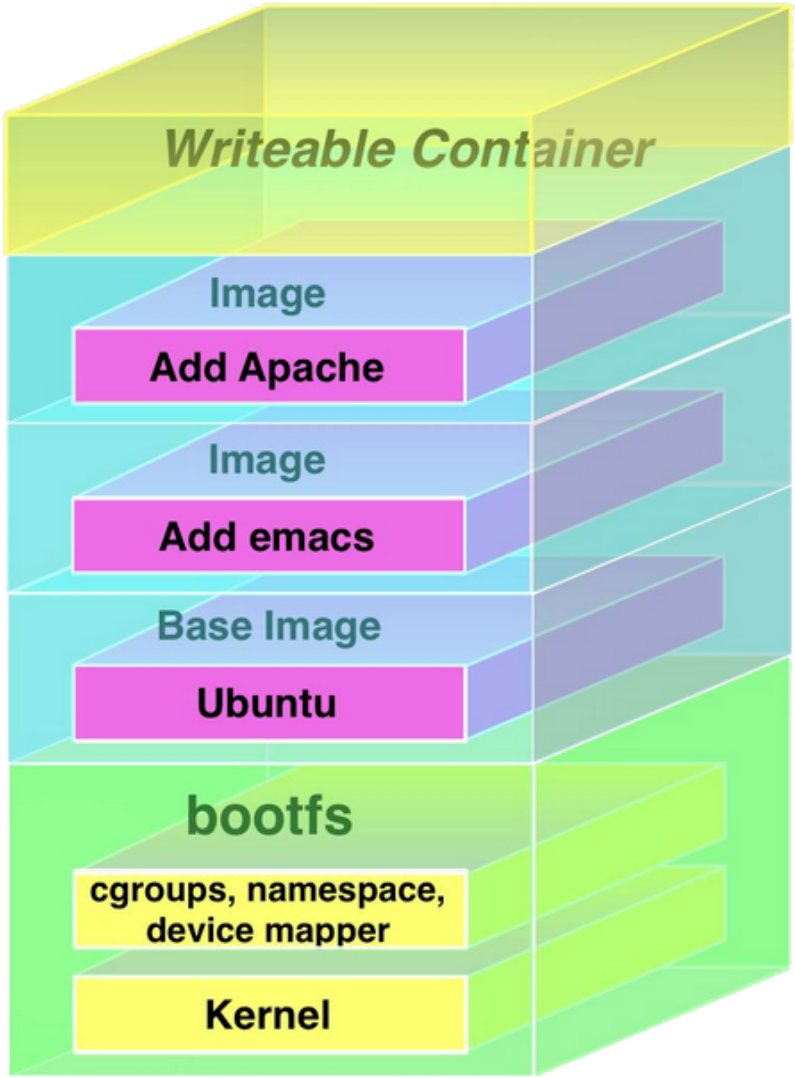
- Docker je izolirana platforma koja sadrži sve potrebno da se pokrene neka specifična aplikacija (dependencies)

```
$ docker run hello-world

$ docker run -it ubuntu /bin/bash
```

Docker images

- Srž Dockera je image sustav na copy-on-write file sustavu
- Hijerarhijska struktura promjena nad osnovnim datotečnim sustavom
- Deklarira se tekstualnom datotekom
- Bazini imagei dostupni na dockerhubu (hub.docker.com)



Dockerfile

```
FROM ruby:2.3

RUN apt-get update \
    && apt-get install -y --no-install-recommends \
        postgresql-client \
        nodejs \
    && rm -rf /var/lib/apt/lists/*

WORKDIR /usr/src/app
COPY Gemfile* ./
RUN bundle install
COPY . .
RUN bundle exec rake assets:precompile

EXPOSE 3000
CMD "./start.sh"
```

Docker naredbe

- Pregled pokrenutih containera

```
$ docker ps
```

- Izgradnja Docker imagea iz foldera gdje se nalazi Dockerfile i postavljanje taga

```
$ docker build -t "mojcontainer:2.0" .
```

- Pokretanje tog imagea

```
$ docker run -p 3000:3000 \
             -v ./data:/app/data \
             mojcontainer:2.0 bash start.sh
```

docker-compose

- Način za pokrenuti sustav containera
- Tekstualna datoteka koja deklarativno opisuje ovisnosti između više containera
- Opisuje i sve postavke containera koje možemo upisati u command line

- Iz foldera s *docker-compose.yml* svi se containeri pokreću naredbom

```
$ docker-compose up
```

docker-compose.yml

```
services:
  db:
    image: postgres
    volumes:
      - ./db/data:/var/lib/postgresql/data
  web:
    build: .
    volumes:
      - .:/usr/src/app
    links:
      - db
    env_file: .env
    ports:
      - "3000:3000"
    restart: always
```

Docker at scale

- Docker olakšava održavanje velikih sustava s redundancijom
- Cluster management - Desetci, stotine, tisuće servera s višestrukim instancama iste aplikacije

- Najpopularniji alati:
- Docker swarm, Kubernetes, Mesos
- Izvan opsega NKOSLa

Literatura

http://www.virtuatopia.com/index.php/An_Overview_of_Virtualization_Techniques

https://wiki.archlinux.org/index.php/Cgroups
https://www.kernel.org/doc/Documentation/cgroups/cgroups.txt

https://wiki.archlinux.org/index.php/Linux_Containers

https://wiki.archlinux.org/index.php/QEMU

http://www.linux-kvm.org/page/Main_Page

https://docs.docker.com/