Kam kráčí Linux: kontejnery, cloud, automatizace

Petr Krčmář

25. května 2017

Prezentace už teď na webu

https://www.petrkrcmar.cz

Linux

- více než 25 let vývoje
- 22 milionů řádek kódu
- téměř 14 tisíc vývojářů
- více než 1300 firem
- každý den přibude 4600 řádek kódu
- Intel, Red Hat, Linaro, Samsung, SUSE...
- ...IBM, Renesas, Google, AMD, Texas Instruments a ARM
- dohromady 44 % kódu

Linux

- více než 25 let vývoje
- 22 milionů řádek kódu
- téměř 14 tisíc vývojářů
- více než 1300 firem
- každý den přibude 4600 řádek kódu
- Intel, Red Hat, Linaro, Samsung, SUSE...
- ...IBM, Renesas, Google, AMD, Texas Instruments a ARM
- dohromady 44 % kódu
- ... a to je jen jádro

Distribuce

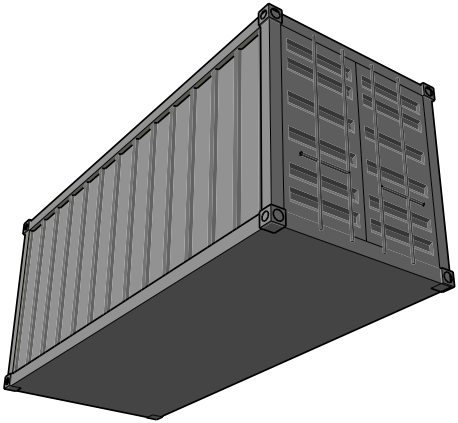
- obsahují desítky tisíc balíčků se softwarem
- komerční vs. nekomerční
- možnost získat podporu velké firmy
- komerční
  - Red Hat (RHEL)
  - SUSE (SLES)
  - Oracle (Oracle Linux)
- nekomerční
  - Debian (Ubuntu)
  - CentOS
  - openSUSE
- a stovky dalších pro různé účely

Linux je všude

- v 10M Alexa 66,7 % Linux/Unix, 33,4 % Windows (W3Techs)
- Ubuntu 35,8 %, Debian 31,9 %, CentOS 20,6 %
  RHEL 3,3 %, Gentoo 2,7 %, Fedora 0,9 %
- v TOP 500 je 498 počítačů s Linuxem (na dvou IBM AIX)
- 1M webů: Apache 41 %, Nginx 29 %, IIS 10 % (Netcraft)

Současnost

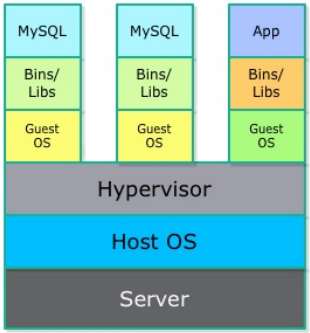
kontejnery, cloud, automatizace

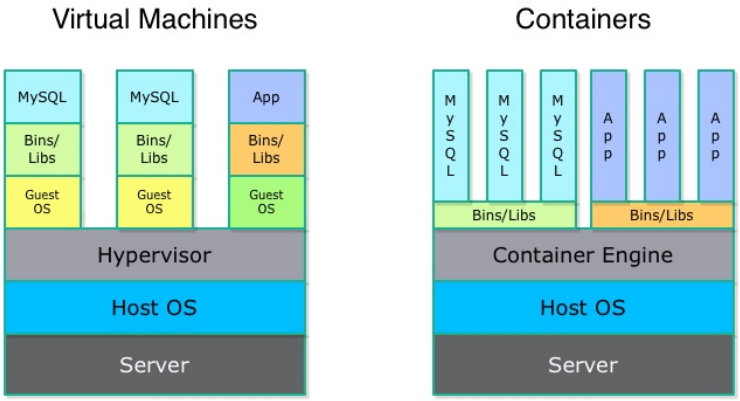


Co jsou kontejnery

- „virtualizace“ na úrovni operačního systému
- nejde o virtualizaci, jen o oddělení procesů
- možnost spustit více uživatelských prostorů
- systém virtualizuje prostředky
- jedno společné jádro, mnoho oddělených systémů
- (chroot), jails, zones, OpenVZ, Linux-VServer…
- omezení: uživatel, souborový systém, paměť, kvóta, síť…
- možnost spustit celou distribuci (VPS) nebo jen aplikaci
- často nás nezajímá systém, chceme aplikaci

Virtual Machines



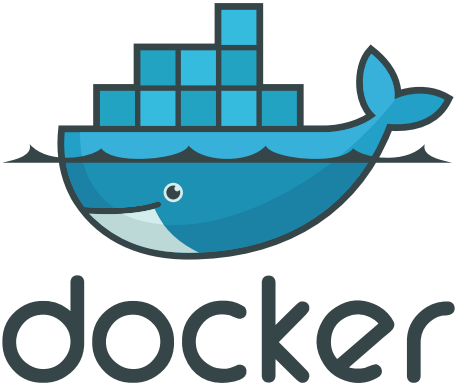

Virtual Machines
MySQL
Bins/ Libs
Guest OS
MySQL
Bins/ Libs
Guest OS
App
Bins/ Libs
Guest OS
Hypervisor
Host OS
Server
Containers
MySQL
MySQL
MySQL
App
App
App
Bins/Libs
Bins/Libs
Container Engine
Host OS
Server

Základ v Linuxu

- linuxové kontejnery staví na Namespaces a Cgroups
- dovolují procesům přepisovat údaje
- proces vidí jiný (virtuální) než skutečný stav
- celkem sedm různých namespaces
  - mnt – připojené svazky
  - pid – process ID
  - net – IP adresy, routovací tabulky, sockety…
  - ipc – meziprocesová komunikace, oddělení sdílené paměti
  - uts – hostname a doména pro různé procesy
  - user – identifikace a oddělení uživatelů
  - cgroup – omezení pohledu na strom
- zbytek zajišťují cgroups (omezení prostředků skupině)

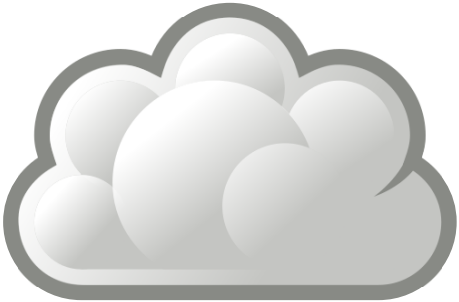
Využití = kontejnery

- použití těchto vlastností = kontejnery
- různé nástroje pro správu
- různé implementace z uživatelského hlediska
- jsou to stále stejné linuxové kontejnery
- obalené různým rozhraním pro různé využití
- oddělení celých systémů nebo aplikací
- nízkoúrovňové vs. vysokoúrovňové
- snapshoty, migrace, uspávání, tvorba šablon...
- až po řízení celého datacentra


docker

Kudy na kontejnery

- Docker – provoz aplikací (PaaS), možnost vrstvení
- LXC – provoz celých systémů, nízkoúrovňová správa
- LXD – nadstavba, škálování, síť, storage, API, živá migrace
- dlouho běžící aktualizovatelná prostředí (LX*)
- malé neměnné aplikační kontejnery (Docker)
- libvirt – integrace hypervizorů a správy
- možnost další integrace na vyšší úrovni



Linux běží v cloudu

- nejrůznější poskytovatelé
- DigitalOcean, Amazon EC2, Microsoft Azure
- tisíce lokálních, včetně vpsFree.cz
- snížení nákladů, netřeba vlastní infrastrukturu
- možnost masivního škálování (vertikálně i horizontálně)
- všechny služby je možné virtualizovat – server, sítě, storage...
- podpora mnoha distribucí, stačí si vybrat

Linux tvoří cloud

- pro potřeby vlastní infrastruktury
- základní technologie: KVM, Xen, LXC
- široká podpora storage řešení
- tradičně velmi silný v síťování
- možnost neomezeného škálování
- potřeba zastřešujícího řešení


openstack®

OpenStack jako vlastní cloud

- softwarová platforma pro stavbu IaaS
- řídí veškeré pochody v datacentru
- výpočetní kapacita, storage i sítě
- využívá řadu existujících technologií
- vytváří jednotné, snadno nasaditelné prostředí
- ovládání přes web, cli nebo REST API

Hlavní komponenty OpenStacku

- Nova – centrální řízení, hardware i různé druhy virtualizace (KVM, Xen, VMWare, Hyper-V, LXC)
- Neutron – řízení sítě, routování, SDN, OpenFlow, IDS, VPN…
- Cinder – storage, Linux, Ceph, IBM Storage…
- Keystone – správa identit, různé možnosti přihlašování (LDAP)
- Glance – správa obrazů disků (templates)
- Swift – objektová storage, škálovatelná redundantní storage
- Trove – databáze
- Heat – orchestrace
- a mnoho dalšího

Typické možnosti nasazení

- poskytovatel cloudu pro jednotlivé zákazníky
- nasazení ve své síti (Red Hat, Canonical, IBM, Oracle, SUSE…)
- poskytování cloudu pro jednoho zákazníka (včetně hardware)
- mix – hardware má zákazník, OpenCloud softwarová služba

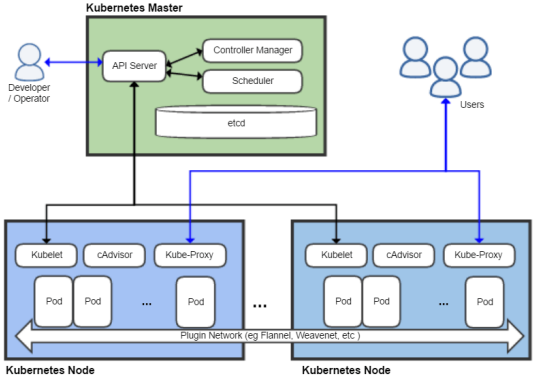
kubernetes

Velké řízení kontejnerů

- nástroj pro správu a orchestraci kontejnerů
- velmi silný v ohromném škálování
- nasazení, škálování, dostupnost, správa
- podporuje celou škálu kontejnerů včetně Dockeru
- původně vyvinul Google (K8s)
- ostrá verze 1.0 v roce 2015
- dnes na něm běží celý Google
- otevřená, předána do rukou komunity
- stále velmi aktivní vývoj
- Red Hat používá v OpenShift

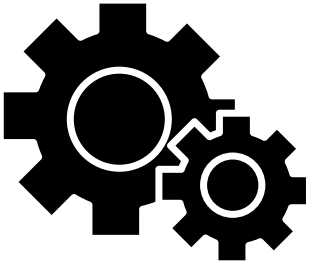
Hlavní komponenty Kubernetes

- Control plane – srdce clusteru, má další komponenty
- etcd – objektová DB ukládající stav clusteru
- API server – zpřístupňuje API pomocí JSON
- Scheduler – plánuje běh podů v závislosti na zdrojích
- Node – virtuál nebo fyzický stroj pro běh aplikací
- Kubelet – proces starající se o node
- Kube-proxy – abstrakce sítě a ballancing
- Pod – kontejner s běžící aplikací


Kubernetes Master
API Server
Controller Manager
Scheduler
etcd
Developer / Operator
Users
Kubelet
cAdvisor
Kube-Proxy
Pod
Pod
...
Pod
...
Kubelet
cAdvisor
Kube-Proxy
Pod
Pod
...
Pod
Plugin Network (eg Flannel, Weavenet, etc.)
Kubernetes Node
Kubernetes Node

Existuje mnoho dalších

- Apache Mesos – abstrakce prostředků napříč datacentrem
- Docker Swarm – správa kontejnerového clusteru
- Amazon ECS – EC2 Container Service
- Azure Container Service (ACS)
- Google Container Engine – postaveno na Kubernetes
- Diego, Kontena, ManageIQ, Kmachine, Wercker…



Automatizace

- zjednodušuje správu serverů
- předchází chybám v konfiguraci
- standardizuje nasazení
- dovoluje ohromné škálování
- umožňuje provádět velké změny na celé farmě
- snadné doplnění nové služby/hardware

Ansible

- automatizace bez agentů na serverech
- vše probíhá přes běžné SSH
- moduly napsané v Python 2 (závislost)
- minimalistický, velmi rychle se učí
- předpisy (playbooky) napsané v jazyce YAML
- podporuje i řadu cloudových služeb
- umí spravovat Windows (PowerShell), Cisco iOS...

Chef

- automatická instalace a provisioning serverů
- napsán v Ruby a Erlang
- podporuje řadu cloudových platforem
- architektura klient/server (vyžaduje agenta)
- server má recept, klienti dodají ingredience
- server pak rozhodné, co se má instalovat a konfigurovat
- umožňuje hlídat závislosti
- podporuje Linux, BSD, Windows, AIX a další platformy

Puppet

- napsán v Ruby
- podporuje Linux i Windows
- má vlastní deklarativní jazyk
- architektura klient/server
- agenti komunikují se serverem a stahují změny
- poté reportují stav serveru
- nabízí abstrakci zdrojů – nezáleží na distribuci
- obstarává celý životní cyklus od instalace přes správu

CFEngine

- konfigurační management
- pro velká firemní řešení (40 000 serverů LinkedIn)
- správa serverů, desktopu, telefonů, tabletů…
- také síťových prvků a průmyslových zařízení
- dobře portovatelný (Linux, Windows, BSD, AIX, Solaris…)
- vlastní deklarativní jazyk
- minimalistický agent sledující stav systému

A mnoho dalších...

- Bcfg2
- cdist
- Foreman
- lSconf
- Pulp
- Rex
- Rudder
- Salt
- Spacewalk
- Synctool
- ...

Vagrant

- správa přenosných vývojových prostředí
- napsán v Ruby, ovládat lze z libovolného jazyka
- spojuje provisionovací a virtualizační nástroje
- Puppet, Chef, VirtualBox, Docker, VMware, AWS…
- vývojář nemusí řešit ani jedno
- vagrantfile popisuje kroky nutné k sestavení stroje
- soubor .box obsahuje
- výsledkem je jednotné replikovatelné vývojové prostředí

nftables

- nový firewall, konkurent iptables
- napsán stejným týmem lidí
- jednoduchý virtuální stroj zpracovávající bytecode
- obecnější principy, 10× méně jaderného kódu
- jedna mocná utilita nft
- obsluhuje vše: IPv4, IPv6, ARP, bridge filtering
- atomické operace, možnost změn, rychlé operace
- snadné rozšíření pravidel v userspace
- zabudované tabulky (jako IPset)
- plánovaná vrstva kompatibility

Reprodukovatelná kompilace

- jak zjistit, že binárka je z konkrétního zdrojáku?
- zlý vývojář, kompromitované build prostředí (CIA, Apple)
- spousta překážek: cesty, časové značky, lokalizace, platforma...
- nutné upravit balíčky, plus vytvořit stabilní prostředí
- nejdále je Debian (94 % balíků pro AMD64)
- automatické testovací nástroje a spousta ruční práce
- NetBSD od letošního roku reprodukovatelné na AMD64 a SPARC64
- dále pracují OpenWrt/LEDE, FreeBSD, Arch Linux, Coreboot
- v plánu Fedora, F-Droid, GNU Guix a další

Sandboxing

- izolace procesů pomocí virtualizace/kontejnerů
- zajištění bezpečnosti při kompromitaci procesu
- dříve chroot (nebezpečné), dnes virt/kont
- Qubes OS – virtuální prostředí pomocí Xen
- RancherOS – Docker jako init system, vše v kontejneru
- Container Linux (původně CoreOS Linux) – Rocket + Gentoo

Bezpečnost kontejnerů

- trápí zejména Docker (ale nejen ten)
- neaktualizované prostředí v kontejnerech
- chyba v OpenSSL = vše znovu sestavit
- výsledek: mnoho kontejnerů je děravých
- hlavně problém veřejných repozitářů
- špatně vytvořené, úmyslně děravé nebo zastaralé
- MongoDB, OpenSSL, glibc stack overflow, SQL injections…
- kontejner jako hrozba pro jádro (DirtyCow)
- možnost úniku z kontejneru – kompromitace systému
- DDoS, ovlivnění ostatních kontejnerů, ztráta dat
- bezpečnostní nástroje teprve vznikají

Otázky?

Petr Krčmář
petr.krcmar@iinfo.cz