

Politechnika Warszawska

W Y D Z I A Ł E L E K T R Y C Z N Y



Instytut Elektrotechniki Teoretycznej
i Systemów Informacyjno-Pomiarowych
Zakład Elektrotechniki Teoretycznej
i Informatyki Stosowanej

Praca dyplomowa inżynierska

na kierunku Informatyka
w specjalności Inżynieria oprogramowania

Implementacja i testy wydajności środowiska Kubernetes na
maszynach bezdyskowych

Krzysztof Nazarewski

nr albumu 123456

promotor
mgr inż. Andrzej Toboła

WARSZAWA 2018

Implementacja i testy wydajności środowiska Kubernetes na maszynach bezdyskowych

Streszczenie

Celem tej pracy inżynierskiej jest przybliżenie czytelnikowi zagadnień związanych z uruchamianiem systemu Kubernetes na maszynach bezdyskowych.

Zacznę od wyjaśnienia pojęcia systemu bezdyskowego oraz sposobu jego funkcjonowania na przykładzie sieci uczelnianej wzorującego się na niej przygotowanie przeze mnie lokalnego środowiska.

Następnie opiszę problem izolacji i przydzielania zasobów systemowych na przykładzie wirtualnych maszyn, chroot, control group i kontenerów.

W głównej części dokumentu przedstawię pojęcie orkiestrami kontenerami, w jaki sposób odnosi się do wcześniej postawionych problemów. Opiszę alternatywy Kubernetes, jego architekturę oraz sposoby uruchamiania. Na koniec spróbuję uruchomić Kubernetes na maszynach bezdyskowych, problemy z tym związane oraz przedstawię wyniki.

Słowa kluczowe: praca dyplomowa, LaTeX, jakość

Implementing and testing Kubernetes running on diskless machines

Abstract

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi ac dolor scelerisque, malesuada ex vel, feugiat augue. Suspendisse dictum, elit efficitur vestibulum eleifend, mi neque accumsan velit, at ultricies ex lectus et urna. Pellentesque vel lorem turpis. Donec blandit arcu lacus, vitae dapibus tellus tempus et. Etiam orci libero, mollis in dapibus tempor, rutrum eget magna. Nullam congue libero non velit suscipit, vel cursus elit commodo. Praesent mollis augue quis lorem laoreet, condimentum scelerisque ex pharetra. Sed est ex, gravida a porta in, tristique ac nunc. Nunc at varius sem, sit amet consectetur velit.

Keywords: thesis, LaTeX, quality

WARSZAWA, 1 lutego 1234

POLITECHNIKA WARSZAWSKA
WYDZIAŁ ELEKTRYCZNY

OŚWIADCZENIE

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa inżynierska pt. Implementacja i testy wydajności środowiska Kubernetes na maszynach bezdyskowych:

- została napisana przeze mnie samodzielnie,
- nie narusza niczyich praw autorskich,
- nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam, że przedłożona do obrony praca dyplomowa nie była wcześniej podstawą postępowania związanego z uzyskaniem dyplomu lub tytułu zawodowego w uczelni wyższej. Jestem świadom, że praca zawiera również rezultaty stanowiące własności intelektualne Politechniki Warszawskiej, które nie mogą być udostępniane innym osobom i instytucjom bez zgody Władz Wydziału Elektrycznego.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Krzysztof Nazarewski.....

Spis treści

1	Test	1
2	Wstęp	2
2.1	System bezdyskowy	2
3	Systemy orkiestracji kontenerami	3
3.1	Fleet	3
3.2	Docker Swarm	3
3.3	Kubernetes	3
3.4	Mesos	3
3.5	Rancher	4
4	Proces uruchamiania maszyn bezdyskowych	5
5	Przegląd systemów operacyjnych	6
5.1	Konfigurator cloud-init	6
5.1.1	Dostępne implementacje	6
5.2	CoreOS	7
5.2.1	Konfiguracja	7
5.3	RancherOS	8
5.3.1	Konfiguracja	8
5.4	Project Atomic	8
5.4.1	Konfiguracja	8
5.5	Alpine Linux ¹	9
5.5.1	Konfiguracja	9
5.6	ClearLinux	9
5.6.1	Linki	9
	Bibliografia	10

¹<https://alpinelinux.org/>

Podziękowania

Dziękujemy bardzo serdecznie wszystkim, a w szczególności Rodzinom i Unii Europejskiej...

Zdolny Student i Pracowity Kolega

Rozdział 1

Test

The seminal work (Pizza i in. 2000)

Rozdział 2

Wstęp

2.1 System bezdyskowy

Rozdział 3

Systemy orkiestracji kontenerami

3.1 Fleet

Fleet¹ jest nakładką na systemd² realizująca rozproszony system inicjacji systemów.

3.2 Docker Swarm

Docker Swarm³ jest rozwiązaniem orkiestracji kontenerami od twórców samego Docker'a. Proste w konfiguracji, nie oferuje tak dużych możliwości jak niżej wymienione.

3.3 Kubernetes

Kubernetes⁴ jest jednym z najpopularniejszych narzędzi orkiestracji kontenerami. Stworzone przez Google i bazowane na wewnętrznym systemie Borg.

3.4 Mesos

Apache Mesos⁵ zaawansowane narzędzie orkiestracji kontenerami.

¹<https://coreos.com/fleet/docs/latest/launching-containers-fleet.html>

²<https://www.freedesktop.org/wiki/Software/systemd/>

³<https://docs.docker.com/engine/swarm/>

⁴<https://kubernetes.io/>

⁵<http://mesos.apache.org/>

3.5 Rancher

Rancher⁶ jest platformą zarządzania kontenerami umożliwiającą między innymi zarządzanie klastrem Kubernetes. Od wersji 2.0 twórcy skupiają się na zarządzaniu Kubernetes porzucając inne silniki

⁶<https://rancher.com/>

Rozdział 4

Proces uruchamiania maszyn bezdyskowych

Na uruchamianie maszyn bezdyskowych w protokole PXE składają się 3 podstawowe elementy: 1. serwer DHCP, np. isc-dhcp-server lub dnsmasq 2. firmware wspierające PXE, np. iPXE 3. serwer plików (np. TFTP, HTTP, NFS)

Rozdział 5

Przegląd systemów operacyjnych

Ze względu na obszerność i niejednoznaczność tematu cloud-init rozdział rozpoczne od jego omówienia.

5.1 Konfigurator cloud-init

cloud-init¹ jest standardem oraz implementacją konfiguratora kompatybilnego z wieloma systemami operacyjnymi przeznaczonymi do działania w chmurze.

Standard polega na dostarczeniu pliku konfiguracyjnego w formacie YAML² w trakcie lub tuż po inicjalizacji systemu operacyjnego.

5.1.1 Dostępne implementacje

cloud-init

Referencyjny cloud-init zaimplementowany jest w Pythonie, co częściowo tłumaczy duży rozmiar obrazów przeznaczonych dla chmury. Po najmniejszych obrazach Pythona dla Dockera³ (python:alpine - 89MB i python2:alpine - 72 MB) wnioskuję, że nie istnieje mniejsza dystrybucja Pythona.

```
docker pull python:2-alpine > /dev/null
docker pull python:alpine > /dev/null
docker images | grep alpine
```

Wywiad z developerem cloud-init⁴

¹<https://cloud-init.io/>

²<http://yaml.org/>

³https://hub.docker.com/_/python/

⁴<https://www.podcastinit.com/cloud-init-with-scott-moser-episode-126>

coreos-cloudinit

coreos-cloudinit⁵ jest częściową implementacją standardu w języku Go przez twórców CoreOS. Niestety rok temu przestał być rozwijany⁶ i wychodzi z użytku.

RancherOS + coreos-cloudinit

rancher cloud-init⁷ jest spadkobiercą⁸ coreos-cloudinit rozwijanym przez zespół RancherOS, na jego potrzeby.

clr-cloud-init

clr-cloud-init⁹ jest wewnętrzną implementacją standardu dla systemu ClearLinux. Powstała z chęci optymalizacji standardu pod ClearLinux oraz pozbycia się zależności referencyjnej implementacji od Python'a.

5.2 CoreOS

CoreOS¹⁰ jest pierwszą dystrybucją linuxa dedykowaną zarządzaniu kontenerami. Zawiera dużo narzędzi dedykowanych klastrowaniu i obsłudze kontenerów, w związku z tym zajmuje 342 MB.

5.2.1 Konfiguracja

Konfiguracja przez Container Linux Config¹¹ transpilowany do Ignition¹². Transpiler konwertuje ogólną konfigurację na przygotowaną pod konkretne chmury (AWS, GCE, Azure itp.). Dyskwalifikującą wadą tego typu konfiguracji jest brak wsparcia transpilatora dla systemów z rodziny BSD.

Poprzednikiem Ignition jest coreos-cloudinit.

⁵<https://github.com/coreos/coreos-cloudinit>

⁶<https://github.com/coreos/coreos-cloudinit/commit/3460ca4414fd91de66cd581d997bf453fd895b67>

⁷<http://rancher.com/docs/os/latest/en/configuration/>

⁸<https://github.com/rancher/os/commit/e2ed97648ad63455743ebc16080a82ee47f8bb0c>

⁹<https://clearlinux.org/blogs/announcing-clr-cloud-init>

¹⁰<https://coreos.com/>

¹¹<https://coreos.com/os/docs/latest/provisioning.html>

¹²<https://coreos.com/ignition/docs/latest/>

5.3 RancherOS

RancherOS¹³ jest systemem operacyjnym, w którym tradycyjny system inicjalizacji został zastąpiony trzema poziomami Dockera¹⁴: - `bootstrap_docker` działający w initramie, czyli przygotowujący system, - `system-docker` zastępuje tradycyjnego inita, zarządza wszystkimi programami systemowymi, - `docker` standardowy docker, interakcja z nim nie może uszkodzić działającego systemu,

Jego głównymi zaletami są mały rozmiar (45 MB) oraz prostota konfiguracji.

5.3.1 Konfiguracja

RancherOS jest konfigurowany przez własną wersję `coreos-cloudinit`.

Znaczną przewagą nad oryginałem jest możliwość sekwencyjnego uruchamiania dowolnej ilości plików konfiguracyjnych.

Przydatne jest wyświetlenie kompletnej konfiguracji komendą `ros config export --full`¹⁵.

5.4 Project Atomic

Project Atomic¹⁶ jest grupą podobnie skonfigurowanych systemów operacyjnych dedykowaną środowiskom cloud i kontenerom.

Dystrybucje Project Atomic nazywają się Atomic Host, dostępne są następujące warianty: - Red Hat Atomic Host¹⁷ - CentOS Atomic Host¹⁸ - Fedora Atomic Host¹⁹

5.4.1 Konfiguracja

Atomic Host są konfigurowane systemem `cloud-init`²⁰,

¹³<https://rancher.com/rancher-os/>

¹⁴<http://rancher.com/docs/os/latest/en/configuration/docker/>

¹⁵<https://forums.rancher.com/t/good-cloud-config-reference/5238/3>

¹⁶<https://www.projectatomic.io/>

¹⁷<https://www.redhat.com/en/resources/enterprise-linux-atomic-host-datasheet>

¹⁸<https://wiki.centos.org/SpecialInterestGroup/Atomic/Download/>

¹⁹<https://getfedora.org/atomic/download/>

²⁰<https://cloud-init.io/>

5.5 Alpine Linux²¹

Minimalna dystrybucja Linuxa bazowana na musl-libc i busybox.

Niestety nie bootuje się w trybie diskless ze względu na błąd, którego twórcy nie umieją naprawić.

5.5.1 Konfiguracja

Alpine Backup²² - spakowane pliki wypakowywane w sekwencji bootu
Alpine Configuration Framework²³

5.6 ClearLinux

ClearLinux²⁴

- “bundle” zamiast pakietów systemowych aktualizowane z całym systemem,
- skoncentrowany na wydajności na procesorach Intel,
- skąpa i trudna w nawigacji dokumentacja systemu,
- lokalizacja wszystkich modyfikacji w /var i /etc (prosty reset),
- instalacja samego docker’a + serwera ssh zajmuje 700 MB w RAMie, co dyskwalifikuje system jako baza uczelnianego Kubernetesa

5.6.1 Linki

- <https://www.infoworld.com/article/3159658/linux/6-key-points-about-intels-hot-new-linux-distro.html>

²¹<https://alpinelinux.org/>

²²https://wiki.alpinelinux.org/wiki/Alpine_local_backup

²³http://wiki.alpinelinux.org/wiki/Alpine_Configuration_Framework_Design

²⁴<https://clearlinux.org/>

Bibliografia

Pizza, Mariagrazia, Vincenzo Scarlato, Vega Massignani, Marzia Monica Giuliani, Beatrice Arico, Maurizio Comanducci, Gary T Jennings, i in. 2000. „Identification of vaccine candidates against serogroup B meningococcus by whole-genome sequencing”. *Science* 287 (5459). American Association for the Advancement of Science:1816–20.